A General Framework for Learning-Augmented Online Allocation

Ilan Reuven Cohen

Faculty of Engineering, Bar-Ilan University, Ramat Gan, Israel.

Debmalya Panigrahi

Department of Computer Science, Duke University, Durham, NC, USA.

Abstract

Online allocation is a broad class of problems where items arriving online have to be allocated to agents who have a fixed utility/cost for each assigned item so to maximize/minimize some objective. This framework captures a broad range of fundamental problems such as the Santa Claus problem (maximizing minimum utility), Nash welfare maximization (maximizing geometric mean of utilities), makespan minimization (minimizing maximum cost), minimization of ℓ_p-norms, and so on. We focus on divisible items (i.e., fractional allocations) in this paper. Even for divisible items, these problems are characterized by strong super-constant lower bounds in the classical worst-case online model.

In this paper, we study online allocations in the *learning-augmented* setting, i.e., where the algorithm has access to some additional (machine-learned) information about the problem instance. We introduce a *general* algorithmic framework for learning-augmented online allocation that produces nearly optimal solutions for this broad range of maximization and minimization objectives using only a single learned parameter for every agent. As corollaries of our general framework, we improve prior results of Lattanzi et al. (SODA 2020) and Li and Xian (ICML 2021) for learning-augmented makespan minimization, and obtain the first learning-augmented nearly-optimal algorithms for the other objectives such as Santa Claus, Nash welfare, ℓ_p-minimization, etc. We also give tight bounds on the resilience of our algorithms to errors in the learned parameters, and study the learnability of these parameters.

2012 ACM Subject Classification Theory of computation → Scheduling algorithms

Keywords and phrases Algorithm with predictions, Scheduling algorithms, Online algorithms

Digital Object Identifier 10.4230/LIPIcs.ICALP.2023.50

Category Track A: Algorithms, Complexity and Games

Funding *Ilan Reuven Cohen*: This research was supported by the Israel Science Foundation grant No. 1737/21.

Debmalya Panigrahi: This research was supported in part by NSF grants CCF-1750140 (CAREER) and CCF-1955703.

1 Introduction

Recent research has focused on obtaining learning-augmented algorithms for many online problems to overcome pessimistic lower bounds in competitive analysis. In this paper, we consider the *online allocation* framework in the learning-augmented setting. In this framework, a set of (divisible) items have to be allocated online among a set of agents, where each agent has a non-negative utility/cost for each item. This framework captures a broad range of classic problems depending on the objective one seeks to optimize. In load balancing (also called *makespan minimization*), the goal is to *minimize the maximum* (MinMax) cost of any agent. A more general goal is to minimize the ℓ_p-norm of the cost vector defined on the agents, for some $p \geq 1$. Both makespan minimization (which is ℓ_∞-minimization) and ℓ_p-minimization are classic problems in scheduling theory and have been extensively studied in competitive analysis. In a different vein, the online allocation framework also applies to

50th International Colloquium on Automata, Languages, and Programming (ICALP 2023).
Editors: Kousha Etessami, Uriel Feige, and Gabriele Puppis; Article No. 50; pp. 50:1–50:20
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

maximization problems, where the allocation of an item obtains some utility for the receiving agent. This includes the famous Santa Claus problem, where the goal is to *maximize the minimum* (MaxMin) utility of any agent, or the maximization of *Nash welfare* which is defined as the geometric mean of the agents' utilities. These maximization objectives have also been been extensively studied, particularly because of their connection to *fairness* in allocations.

Learning-Augmented Online Allocation. In this paper, we consider the online allocation framework in the *learning-augmented* setting. Typically, online allocation problems are characterized by strong super-constant lower bounds in competitive analysis, e.g., $\Omega(\log m)$ for load balancing [7], $\Omega(p)$ for ℓ_p-minimization [4] and $\Omega(m)$ for both Santa Claus (folklore) and Nash welfare [9]. A natural question, then, is whether some additional (machine-learned) information about the problem instance (we call these *learned parameters*) can help overcome these lower bounds and obtain a near-optimal solution. In this paper, we answer this question in the affirmative. In particular, we give a simple, unified framework for obtaining near-optimal (fractional) allocations *using a single learned parameter for every agent.* Our result holds for both maximization and minimization problems, and applies to all objective functions that satisfy two mild technical conditions that we define below. Indeed, the most interesting aspect of our techniques and results is this generality: prior work for online allocation problems, both in *competitive analysis* and *beyond worst-case algorithms*, has typically been specific to the objective at hand, and the techniques for maximization and minimization objectives bear no similarity. In contrast, our techniques surprisingly handles not only a broad range of objectives but applies both to maximization and minimization problems simultaneously. We hope that the generality of our methods will cast a new light on what is one of the most important classes of problems in combinatorial optimization.

Before proceeding further, we define the two technical conditions that the objective function of the online allocation problem needs to satisfy for our results to apply. Let $f : \mathbb{R}^m_{>0} \to \mathbb{R}_{>0}$ be the objective function defined on the vector of costs/utilities of the agents. Then, the conditions are:

- *Monotonicity:* f is said to be *monotone* if the following holds: for any $\ell, \ell' \in \mathbb{R}^m_{>0}$ such that $\ell_i \geq \ell'_i$ for all $i \in [m]$, we have $f(\ell) \geq f(\ell')$.
- *Homogeneity:* f is said to be *homogeneous* if the following holds: for any $\ell, \ell' \in \mathbb{R}^m_{>0}$ such that $\ell'_i = \alpha \cdot \ell_i$ for all $i \in [m]$, then we have $f(\ell') = \alpha \cdot f(\ell)$.

We say an objective function is *well-behaved* if it is both monotone and homogeneous. All online allocation objectives studied previously that we are aware of are well-behaved, including the examples given above.

1.1 Our Results

We now state our main result below:

▶ **Theorem 1** (Informal). *Fix any $\epsilon > 0$. For any online allocation problem with a well-behaved objective, there is an algorithm that achieves a competitive ratio of $1 - \epsilon$ for maximization problems or $1 + \epsilon$ for minimization problems using a single learned parameter for every agent.*

We remark that the role of ϵ in the above theorem is to ensure that the learned parameter vector is of bounded precision.

Comparison to Prior Work. Lattanzi *et al.* [17] were the first to consider online allocation in a learning-augmented setting. They considered a special case of the load balancing problem called restricted assignment, and showed the surprising result that a single (learned) parameter for each agent is sufficient to bypass the lower bound and obtain a nearly optimal (fractional) allocation. This result was further generalized by Li and Xian [20] to the full generality of the load balancing problem, but instead of a single parameter, they now required two parameters for every agent. At a high level, their algorithm first uses one set of parameters to restrict the set of agents who can receive an item, and then solves the resulting restricted assignment problem using the second set of parameters. A a corollary of Theorem 1, we improve this result by obtaining a near-optimal solution using a single learned parameter for every agent. In both these papers, as well as in our paper, the (fractional) allocation uses *proportional allocation*. In the setting of online optimization, proportional allocations were used earlier by Agrawal *et al.* [1] for the (weighted) b-matching problem. As in our paper, they also gave an iterative algorithm for computing the parameters of the allocation. However, because the two problems are structurally very different (e.g., matching is a packing problem while our allocation problems are covering problems), the iterative algorithm in the Agrawal *et al.* paper is different from ours. To the best of our knowledge, our results for the other problems, namely Santa Claus, Nash welfare maximization, ℓ_p-norm minimization, and other objectives that can be defined in the online allocation framework are the first results in learning-augmented algorithms for these problems.

We now state our additional results.

Resilience to Prediction Error. A key desiderata of learning-augmented online algorithms is resilience to errors in the learned parameters. In other words, one desires that the competitive ratio of the algorithm should gracefully degrade when the learned parameters used in the algorithm deviate from their optimal values. For well-behaved objectives for both minimization and maximization problems, we give an error-resilient algorithm whose competitive ratio degrades gracefully with prediction error:

▶ **Theorem 2** (Informal). *For any online allocation problem with a well-behaved objective, there is an (learning-augmented) algorithm that achieves a competitive ratio of $O(\alpha)$ when the learned parameter input to the algorithm is within a multiplicative factor of α of the optimal learned parameter for every agent. This holds for both minimization and maximization objectives.*

The above theorem is asymptotically tight for the MaxMin objective. But, interestingly, for the MinMax objective we can do better:

▶ **Theorem 3** (Informal). *For the load balancing problem (*MinMax *objective), there is an (learning-augmented) algorithm that achieves a competitive ratio of $O(\log\alpha)$ when the learned parameter input to the algorithm is within a multiplicative factor of α of the optimal learned parameter for every agent. Moreover, the dependence $O(\log\alpha)$ in the above statement is asymptotically tight.*

An analogous statement was previously known only in the special case of restricted assignment [17].

▶ Remark 4. We use a multiplicative measure of error α similar to [17]. For both MinMax and MaxMin objectives, we may assume w.l.o.g. that $\alpha \le m$. This is because by standard techniques, it is possible to achieve $O(\min(\alpha, m))$ and $O(\log\min(\alpha, m))$ competitiveness for the MaxMin and MinMax objectives respectively. We also show that our bounds are asymptotically tight as a function of α, in addition to matching existing lower bounds for the two problems as a function of m.

Learnability of Parameters. We also study the learnability of the parameters used in our algorithm. Following [20] and [18], we adopt the PAC framework. We assume that each item is drawn independently (but not necessarily identically) from a distribution, and show a bound on the sample complexity of approximately learning the parameter vector under this setting. For the MaxMin and MinMax objectives, we show the following:

▶ **Theorem 5** (Informal). *Fix any $\epsilon > 0$. For the online allocation problem with* MaxMin *or* MinMax *objectives, the sample complexity of learning a parameter vector that gives a $1-\epsilon$ (for* MaxMin*) or $1+\epsilon$ (for* MinMax*) approximation is $O(\frac{m}{\log m} \cdot \log \frac{m}{\epsilon})$.*

We note that a similar result was previously known for the MinMax objective (Li and Xian [20]). We also generalize this result to all well-behaved objectives subject to a technical condition of *superadditivity* for maximization or *subadditivity* for minimization. All the objectives described earlier in the introduction satisfy these conditions.

1.2 Our Techniques

Our learning-augmented online algorithms for both minimization and maximization objectives follow from a single, unified algorithmic framework that we develop in this paper. This is quite surprising because in the worst-case setting, the online algorithms for the different objectives do not share any similarity (indeed have different competitive ratios), particularly between maximization and minimization problems. First, let us first consider the MinMax and MaxMin objectives. To use common terminology across these problems, let us call the cost/utility of an item j to an agent i the *weight* of item j for agent i and denote it $p_{i,j}$. Our common algorithmic framework uses proportional allocation according to the learned parameters of the agents. Let w_i denote the parameter for agent i. Normally, proportional allocation would entail that we allocate a fraction $x_{i,j}$ of item j to agent i where $x_{i,j} = \frac{w_i p_{i,j}}{\sum_{i'} w_{i'} p_{i',j}}$. But, this is clearly not adequate, since it would produce the same allocation for both the MaxMin and MinMax objectives. Specifically, if $p_{i,j}$ is *large* for a pair i, j, then $x_{i,j}$ should be large for the MaxMin objective and small for the MinMax objective respectively. To implement this intuition, we exponentiate the weight $p_{i,j}$ by a fixed value α that depends on the objective (i.e., is different for MaxMin and MinMax) and then allocate using fractions $x_{i,j} = \frac{w_i p_{i,j}^{\alpha}}{\sum_{i'} w_{i'} p_{i',j}^{\alpha}}$. We call this an *exponentiated proportional* allocation (or EP-allocation in short), and call α the *exponentiation constant.*

Let us fix any value of α. It is clear that for both the MinMax and MaxMin objectives, an optimal allocation has *uniform* cumulative fractional weights (called *load*) across all agents. (Note that otherwise, an infinitesimal fraction of an item can be repeatedly moved from the most loaded to the least loaded agent to eventually improve the competitive ratio.) Following this intuition, we define a *canonical allocation* as one that sets learned parameters on the agents in a way that equalizes the loads on all agents. We show that the canonical allocation always exists and is *unique.* Indeed, this is true not only for all EP-allocation algorithms, but for a much broader class of proportional allocation schemes that we called *generalized proportional* allocations (or GP-allocations). In the latter class, we allow any transformation of the weights $p_{i,j}$ before applying proportional allocation. Thus, EP-allocations represent the subclass of GP-allocations where the transformation is exponentiation by the fixed value α. We also give a simple iterative (Sinkhorn-like) algorithm for computing the optimal learned parameters, and establish its convergence properties, for GP-allocations. GP-allocations give an even larger palette of proportional allocation schemes to choose from than EP-allocations, and we hope it will be useful in future work for problem settings that are not covered in this paper (e.g., non-linear utilities).

Finally, we need to set the value of α specifically for the MinMax and MaxMin objectives. Intuitively, it is clear that we need to set α to a large *positive* value for the MaxMin objective and a large *negative* value for the MinMax objective. Indeed, we show that in the limit of $\alpha \to \infty$ and $\alpha \to -\infty$, the canonical allocation defined above recovers optimal allocations for the MaxMin and MinMax objectives respectively. We also show a monotonicity property of the optimal objective (with the value of α) that can be used to set α to a finite value (function of ϵ) and obtain a $1-\epsilon$ (resp., $1+\epsilon$) approximation for the MaxMin (resp., MinMax) objective, for any $\epsilon > 0$.

Now that we have described the EP-allocation scheme for obtaining nearly optimal algorithms for the MinMax and MaxMin objectives, we generalize to all well-behaved objective functions. This is quite simple. The main advantage of the MinMax and MaxMin objectives that is not shared by other objectives is the property that the optimal solution has uniform load across all agents. Now, suppose for a maximization objective, the load of agent i in an optimal solution is s_i (we call this the *scaling parameter* for agent i). For now, suppose these values s_i are also provided offline as a second set of parameters. Then, we can first scale the weights $p_{i,j}$ using these parameters to obtain a new instance $q_{i,j} = \frac{p_{i,j}}{s_i}$. Clearly, the optimal solution for the original instance has uniform load across all agents for the transformed instance. Indeed, by the monotonicity of the maximization objective, this solution for the transformed instance is also optimal for the MaxMin objective. Using the above analysis for the MaxMin objective, we can now claim that there exist learned parameters w_i for $i \in [m]$ such that setting $x_{i,j} = \frac{w_i q_{i,j}^\alpha}{\sum_{i'} w_{i'} q_{i',j}^\alpha}$ gives an optimal solution to the original instance of the problem. Now, note that

$$x_{i,j} = \frac{w_i q_{i,j}^\alpha}{\sum_{i'} w_{i'} q_{i',j}^\alpha} = \frac{(w_i/s_i^\alpha) p_{i,j}^\alpha}{\sum_{i'} (w_{i'}/s_{i'}^\alpha) p_{i',j}^\alpha} = \frac{w'_i p_{i,j}^\alpha}{\sum_{i'} w'_{i'} p_{i',j}^\alpha} \text{ for } w'_i = w_i/s_i^\alpha.$$

It follows that by using learned parameters w'_i in an EP-allocation on the original instance, we can obtain an optimal solution for the original maximization objective. (The case for a minimization objective is identical to the above argument, with the MaxMin objective being replaced by the MinMax objective.) Finally, using the homogeneity of the objective function, we can also set α to a finite value (function of ϵ) and obtain a $1-\epsilon$ (resp., $1+\epsilon$) approximation for the maximization (resp., minimization) objective, for any $\epsilon > 0$.

1.3 Related Work

Learning-augmented online algorithms were pioneered by the work of Lykouris and Vassilvikskii [21] for the caching problem, and has become a very popular research area in the last few years. The basic idea of this framework is to augment an online algorithm with (machine-learned) predictions about the future, which helps overcome pessimistic worst case lower bounds in competitive analysis. Many online allocation problems have been considered in this framework in scheduling [27, 5, 6, 8, 15, 24], online matching [2, 13, 16], ad delivery [22, 19], etc. The reader is referred to the survey by Mitzenmacher and Vassilvitskii [25, 26] for further examples of online learning-augmented algorithms. The papers specifically related to our work are those of Lattanzi et al. [17] and Li and Xian [20] that we described above, and that of Lavastida et al. [18] that focuses on the learnability of the parameters for the same problem. As mentioned earlier, Agrawal *et al.* [1] used the proportional allocation framework earlier for the online (weighted) b-matching problem, and gave an iterative algorithm for computing the parameters of the allocation.

We now give a brief summary of online allocation in the worst-case model. For minimization problems, two classic objectives are makespan (i.e., ℓ_∞ norm) and ℓ_p norm minimization for $p > 1$. The former was studied in several works (e.g., [7, 3]), eventually leading to an asymptotically tight bound of $\Theta(\log m)$. This was later generalized to arbitrary ℓ_p norms, and a tight bound of $\Theta(p)$ was obtained for this case [4, 12]. For maximization objectives, there are $\Omega(m)$ lower bounds for many natural objectives such as MaxMin (see, e.g., [14]) and Nash welfare [9]. Some recent work has focused on overcoming these lower bounds using additional information such as monopolist values for the agents [9, 10]. While this improves the competitive ratio to sub-linear in m, lower bounds continue to rule out near-optimal solutions (or even constant factor approximations) that we seek in this paper.

Organization. For most of the paper, we only consider the MinMax and MaxMin objectives. We establish the notation in Section 2 and give an overview of the results. Then, we prove these results by showing properties of GP-allocations in Section 3 and of EP-allocations in Section 4. Next, we give noise resilient algorithms in Section 5 and discuss learnability of the parameters in Section 6. Finally, in Section 7, we extend our results to all well-behaved objective functions via simple reductions to the MaxMin and MinMax objectives.

2 Preliminaries and Results

2.1 Problem Definition

We have n (divisible) items that arrive online and have to be (fractionally) allocated to m agents. The weight of item $j \in [n]$ for agent $i \in [m]$ is denoted $p_{i,j}$ and is revealed when item j arrives. We denote the *weight matrix*

$$P = \begin{bmatrix} p_{1,1} & \cdots & p_{1,n} \\ \vdots & \ddots & \vdots \\ p_{m,1} & \cdots & p_{m,n} \end{bmatrix} \text{ where all } p_{i,j} > 0 \text{ for all } i \in [m], j \in [n].^1$$

A feasible allocation is given by an *assignment matrix*

$$X = \begin{bmatrix} x_{1,1} & \cdots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{m,1} & \cdots & x_{m,n} \end{bmatrix} \text{ where } x_{i,j} \in [0,1] \text{ for all } i \in [m], j \in [n] \text{ and } \sum_{i=1}^{m} x_{i,j} = 1 \text{ for all } j \in [n].$$

Note that every item has to be fully allocated among all the agents. We use $\mathcal{X}$ to denote the set of feasible solutions. The total weight of an agent i corresponding to an allocation X (we call this the *load* of i) is given by

$$\ell_i(P, X) = \sum_{j \in [n]} x_{i,j} \cdot p_{i,j},$$

and the vector of loads of all the agents is denoted $\boldsymbol{\ell}(P, X)$.

The load balancing problem is now defined as

$$\min_{X \in \mathcal{X}} \Big\{ T : \ell_i(P, X) \le T \text{ for all } i \in [m] \Big\},$$

while the Santa Claus problem is defined as

$$\max_{X \in \mathcal{X}} \Big\{ T : \ell_i(P, X) \ge T \text{ for all } i \in [m] \Big\}.$$

2.2 Exponentiated and Generalized Proportional Allocations

Our algorithmic framework is simple: when allocating item j, we first exponentiate the weights $p_{i,j}$ to $p_{i,j}^{\alpha}$ for some fixed α (called the *exponentiation constant*) that only depends on the objective being optimized. Next, we perform proportional allocation weighted by the learned parameters w_i for agents $i \in [m]$:

$$x_{i,j} = \frac{p_{i,j}^{\alpha} \cdot w_i}{\sum_{i' \in [m]} p_{i',j}^{\alpha} \cdot w_{i'}}.$$

We call this an *exponentiated proportional* allocation or EP-allocation in short.

Our main theorem is the following:

▶ **Theorem 6.** *For the load balancing and Santa Claus problems, there are* EP*-allocations that achieve a competitive ratio of* $1 + \epsilon$ *and* $1 - \epsilon$ *respectively, for any* $\epsilon > 0$.

The Canonical Allocation. In order to define an EP-allocation and establish Theorem 6, we need to specify two things: the vector of learned parameters $\mathbf{w} \in \mathbb{R}^m_{>0}$ and the exponentiation constant α. First, we focus on the learned parameters. For any fixed α and a weight matrix P, we use learned parameters $\mathbf{w} \in \mathbb{R}^m_{>0}$ that result in *equal load* for every agent. We call this the *canonical allocation*. The corresponding learned parameters and the load of every agent are respectively called the *canonical parameters* (denoted $\mathbf{w}^*$) and the *canonical load* (denoted ℓ^*).

Apriori, it is not clear that a canonical allocation should even exist, and even if it does, that it is unique. Interestingly, we show this existence and uniqueness not just from EP-allocations but for the much broader class of proportional allocations where *any* function $f : \mathbb{R}_{>0} \to \mathbb{R}_{>0}$ (called the *transformation function*) can be used to transform the weights rather than just an exponential function. I.e.,

$$x_{i,j} = \frac{f(p_{i,j}) \cdot w_i}{\sum_{i' \in [m]} f(p_{i',j}) \cdot w_{i'}}.$$

We call this a *generalized proportional* allocation or GP-allocation in short.

We show the following theorem for GP-allocations:

▶ **Theorem 7.** *For any weight matrix* $P \in \mathbb{R}^{m \times n}_{>0}$ *and any transformation function* $f : \mathbb{R}_{>0} \to \mathbb{R}_{>0}$*, the canonical load for a* GP*-allocation exists and is unique. Moreover, it is attained by a unique (up to scaling) set of canonical parameters.*

We prove Theorem 7 algorithmically by giving a simple iterative (offline) algorithm that converges to the set of canonical parameters (see Algorithm 1). We will show later that the canonical allocations produced by appropriately setting the value of the exponentiation constant α are respectively optimal (fractional) solutions for the Santa Claus and the load balancing problems. Therefore, an interesting consequence of the iterative convergence of this algorithm to the canonical allocation is that it gives a simple alternative *offline* algorithm for computing an optimal fractional solution for these two problems. To the best of our knowledge, this was not explicitly known before our work.

An interesting direction for future research would be to explore other natural classes of transformation functions, other than the exponential functions considered in this paper. Since Theorem 7 holds for any transformation function, they also admit a canonical allocation, and it is conceivable that such canonical allocations would optimize objective functions other than the MinMax and MaxMin functions considered here. For example, one natural open problem is following: are there a transformation functions whose canonical allocations correspond to maximizing Nash Social Welfare or minimizing p-norms of loads?

Monotonicity and Convergence of EP-allocations. Now that we have defined the learned parameters in Theorem 6 as the corresponding canonical parameters, we are left to define the values of the exponentiation constant α for the MaxMin and MinMax problems respectively. We show two key properties of canonical loads of EP-allocations. First, we show that the canonical load is monotone nondecreasing with the value of α. This immediately suggests that we should choose the largest possible value of α for the MaxMin problem since it is a maximization problem, and the smallest possible value of α for the MinMax problem since it is a minimization problem. Indeed, the second property that we show is that in the limit of $\alpha \to \infty$, the canonical load converges to the optimal objective for the Santa Claus problem (we denote this optimal value $\ell^{\mathbf{SNT}}$) and in the limit of $\alpha \to -\infty$, the canonical load converges to the optimal objective for the load balancing problem (we denote this optimal value $\ell^{\mathbf{MKS}}$).

For a fixed α, let $X(P, \alpha, \mathbf{w})$ denote the assignment matrix and $\boldsymbol{\ell}(P, \alpha, \mathbf{w})$ the load vector for a learned parameter vector $\mathbf{w}$. Let $\boldsymbol{\ell}^*(P, \alpha)$ denote the corresponding canonical load. We show the following properties of canonical EP-allocations:

▶ **Theorem 8.** *For any weight matrix $P \in \mathbb{R}_{>0}^{m\times n}$, the following properties hold for canonical* EP*-allocations:*

- *The monotonicity property: For $\alpha_1, \alpha_2 \in \mathbb{R}$ such that $\alpha_1 \geq \alpha_2$, we have $\ell^*(P, \alpha_1) \geq \ell^*(P, \alpha_2)$.*
- *The convergence property: $\lim_{\alpha\to\infty} \ell^*(P, \alpha) = \ell^{\boldsymbol{SNT}}(P)$ and $\lim_{\alpha\to-\infty} \ell^*(P, \alpha) = \ell^{\boldsymbol{MKS}}(P)$.*

Clearly, Theorem 8 implies Theorem 6 as a corollary when α is set sufficiently large for the Santa Claus problem and sufficiently small for the load balancing problem.

In the rest of the paper, we will prove Theorem 7 and Theorem 8.

3 Canonical Properties of Generalized Proportional Allocations

In this section, we prove Theorem 7. For notational convenience, we define a transformation matrix $G \in \mathbb{R}_{>0}^{m\times n}$ where $G(i, j) = f(p_{i,j})$ for the transformation function f. Using this notation, we denote by $x_{i,j}(G, \mathbf{w})$ the fractional allocation of item j to agent i, and by $\ell_i(P, G, \mathbf{w})$ the load of agent i (we use $\boldsymbol{\ell}(P, G, \mathbf{w})$ to denote the vector of agent loads) under the GP-allocation corresponding to the transformation matrix G and learned parameters $\mathbf{w}$.

We say two sets of learned parameters $\mathbf{w}, \mathbf{w}'$ are *equivalent* (denoted $\mathbf{w} \equiv \mathbf{w}'$) if there exists some constant $c > 0$ such that $w'_i = c \cdot w_i$ for every agent $i \in [m]$. The following is a simple observation from the GP-allocation scheme that two equivalent sets of learned parameters produce the same allocation:

▶ **Observation 9.** *For any $G \in \mathbb{R}_{>0}^{m\times n}$, if $\mathbf{w} \equiv \mathbf{w}' \in \mathbb{R}_{>0}^{m}$, then $x_{i,j}(G, \mathbf{w}) = x_{i,j}(G, \mathbf{w}')$ for all i, j.*

We also note that GP-allocations are monotone in the sense that if one agent's parameter decreases while the rest increase, then the allocation on this agent decreases as well.

▶ **Observation 10.** *Consider any $G \in \mathbb{R}_{>0}^{m\times n}$ and any nonzero vector $\boldsymbol{\epsilon} \in \mathbb{R}_{\geq 0}^{m}$ such that $-w_k < \epsilon_k \leq 0$ for some $k \in [m]$ and $\epsilon_i \geq 0$ for all $i \neq k$. Then, $x_{k,j}(G, \mathbf{w}') < x_{k,j}(G, \mathbf{w})$ for all $j \in [n]$, where $\mathbf{w}' = \mathbf{w} + \boldsymbol{\epsilon}$ and $\mathbf{w}' \neq \mathbf{w}$.*

Our first nontrivial property is that the load vector uniquely determines the learned parameters up to equivalence of the parameters.

▶ **Lemma 11.** *For any* $P, G \in \mathbb{R}_{>0}^{m\times n}$, $\ell_i(P,G,\mathbf{w}) = \ell_i(P,G,\mathbf{w}')$ *for all* $i \in [m]$ *if and only if* $\mathbf{w} \equiv \mathbf{w}'$.

Proof. In one direction, if $\mathbf{w} \equiv \mathbf{w}'$, the loads are identical because the allocations are identical (by Observation 9).

We now show the lemma in the opposite direction. Let $k = \arg\min_i \frac{w_i}{w'_i}$ and $c = \frac{w_k}{w'_k}$. Let us define $\hat{\mathbf{w}} = c \cdot \mathbf{w}'$. Then, $\hat{w}_k = w_k$, and $\hat{w}_{i'} = \left(\min_i \frac{w_i}{w'_i}\right) \cdot w'_{i'} \leq w_{i'}$ for all $i' \neq k$. Now, if $\mathbf{w}$ and $\mathbf{w}'$ are not equivalent, then there exists some $i' \in [m]$ such that $\hat{w}_{i'} < w_{i'}$. Therefore, by Observation 10, $x_{k,j}(G,\hat{\mathbf{w}}) > x_{k,j}(G,\mathbf{w})$ for all $j \in [n]$. But, by Observation 9, $x_{k,j}(G,\hat{\mathbf{w}}) = x_{k,j}(G,\mathbf{w}')$ for all $j \in [n]$. Thus, $x_{k,j}(G,\mathbf{w}') > x_{k,j}(G,\mathbf{w})$ for all $j \in [n]$, which contradicts $\ell_k(P,G,\mathbf{w}') = \ell_k(P,G,\mathbf{w})$. ◀

Similarly, we show that if the canonical load exists (i.e., a load vector where all loads are identical), it must be unique.

▶ **Lemma 12.** *For any* $P, G \in \mathbb{R}_{>0}^{m\times n}$, *if there exist* $\mathbf{w}, \mathbf{w}' \in \mathbb{R}_{>0}^{m}$ *such that* $\ell_i(P,G,\mathbf{w}) = \ell$ *and* $\ell_i(P,G,\mathbf{w}') = \ell'$ *for all* $i \in [m]$, *then* $\ell = \ell'$.

Proof. Assume for the purpose of contradiction that there exist $\mathbf{w}, \mathbf{w}' \in \mathbb{R}_{>0}^{m}$ such that for all $i \in [m]$, $\ell_i(P,G,\mathbf{w}) = \ell$ and $\ell_i(P,G,\mathbf{w}') = \ell'$ but $\ell > \ell'$. Let $k = \arg\min_i \frac{w_i}{w'_i}$ and $c = \frac{w_k}{w'_k}$, and let $\hat{\mathbf{w}} = c \cdot \mathbf{w}'$. We have

$$\ell' = \ell_k(P,G,\mathbf{w}') = \ell_k(P,G,\hat{\mathbf{w}}) \geq \ell_k(P,G,\mathbf{w}) = \ell, \text{which is a contradiction.}$$

Here, the second equality is by Observation 9, and the inequality is by Observation 10, since $\hat{w}_k = w_k$, and $\hat{w}_i \leq w_i$ for $i \in [m]$. ◀

3.1 Convergence of Algorithm 1

The rest of this section focuses on showing the existence of a canonical allocation for GP-allocations. We do so by showing convergence of the following simple iterative algorithm (Algorithm 1):

Algorithm 1 The iterative algorithm showing the existence of a canonical allocation for GP-allocations.

- Initialize: $\mathbf{w}^{(0)} \leftarrow \mathbf{1}^m$

Iteration r:

- Compute $\boldsymbol{\ell}^{(r)}$ as $\ell_i^{(r)} \leftarrow \ell_i(P,G,\mathbf{w}^{(r)})$, for all $i \in [m]$, where $\ell_i(P,G,\mathbf{w}^{(r)})$ is the load of agent i under the GP-allocation with transformation matrix G and learned parameters $\mathbf{w}^{(r)}$.
- Set $\mathbf{w}^{(r+1)}$ as $w_i^{(r+1)} \leftarrow \frac{w_i^{(r)}}{\ell_i^{(r)}} \cdot \gamma^{(r)}$, for all $i \in [m]$.
  Here, $\gamma^{(r)} \in \mathbb{R}_{>0}$ is a scaling factor whose value does not affect the load (by Observation 9). But, by using, e.g., $\gamma^{(r)} = \ell_1^{(r)}$, we can ensure that the algorithm terminates with a single set of learned parameters instead of repeatedly finding equivalent sets of parameters after it has converged.

Note that Algorithm 1 ensures that if the loads of all agents are uniform at any stage, then the iterative process has converged and the algorithm terminates. So, it remains to show that for any $P, G \in \mathbb{R}^{m\times n}_{>0}$, this iterative process reaches a set of parameters $\mathbf{w}^* \in \mathbb{R}^m_{>0}$ such that $\ell_i(P, G, \mathbf{w}^*) = \ell_{i'}(P, G, \mathbf{w}^*)$ for all $i, i' \in [m]$.

Our proof has two parts. The first part shows that the maximum and minimum loads are (weakly) monotone over the course of the iterative process. For this, we focus on a single iteration. For a vector $\boldsymbol{\ell} \in \mathbb{R}^m_{>0}$, let $\ell_{\max} = \max_{i\in[m]} \ell_i$ and $\ell_{\min} = \min_{i\in[m]} \ell_i$ be the maximum and minimum coordinates of $\boldsymbol{\ell}$. We will show that if $\ell^{(r)}_{\max}$ and $\ell^{(r)}_{\min}$ are not equal at the beginning of an iteration, then $\ell^{(r)}_{\max}$ can only decrease (or stay unchanged) and $\ell^{(r)}_{\min}$ can only increase (or stay unchanged) in a single iteration.

▶ **Lemma 13.** *Consider any $P, G \in \mathbb{R}^{m\times n}_{>0}$, $\gamma > 0$. Let $\mathbf{w}, \mathbf{w}', \boldsymbol{\ell}, \boldsymbol{\ell}' \in \mathbb{R}^m_{>0}$ such that $\ell_i = \ell_i(P, G, \mathbf{w})$, $\ell'_i = \ell_i(P, G, \mathbf{w}')$ and $w'_i = \frac{w_i}{\ell_i} \cdot \gamma$ and let $\tilde{p}_i = \sum_j p_{i,j}$. Then, we have $\ell'_i \geq \ell_{\min}/\left(1 - \frac{\ell_i - \ell_{\min}}{\tilde{p}_i}\right)$ and $\ell'_i \leq \ell_{\max}/\left(1 + \frac{\ell_{\max} - \ell_i}{\tilde{p}_i}\right)$*

In the second part, we show that the ratio $\frac{\ell^{(r)}_{\max}}{\ell^{(r)}_{\min}}$ is strictly decreasing after a finite number of iterations. The proof of this stronger property requires the per-iteration weak monotonicity property that we establish in the first part of the proof.

▶ **Lemma 14.** *Let $P, G \in \mathbb{R}^{m\times n}_{>0}$ be given fixed matrices. Fix an iteration r in Algorithm 1 where $\ell^{(r)}_{\max} > \ell^{(r)}_{\min}$. Let $\ell^{(r)}_{\max} \geq (1+\epsilon) \cdot \ell^{(r)}_{\min}$ for some $\epsilon \in (0, 1]$. Then, in the next iteration, we have $\ell^{(r+1)}_{\min} \geq (1 + c\cdot\epsilon) \cdot \ell^{(r)}_{\min}$ for some constant $c > 0$ that only depends on P and G.*

Using Lemma 13 and Lemma 14, we complete the proof of Theorem 7.

Proof of Theorem 7. We are given fixed matrices $P, G \in \mathbb{R}^{m\times n}_{>0}$. Let $\ell^{(r)}_{\max}, \ell^{(r)}_{\min}$ denote the maximum and the minimum load respectively in iteration r of Algorithm 1. Let $c > 0$ be the constant (that depends only on P, G) in Lemma 14.

For a non-negative integer a, let r_a be defined recursively as follows:

$$r_a = r_{a-1} + \left\lceil \frac{\log(1 + 2^{-a+1})}{\log\left(1 + c\cdot 2^{-a}\right)} \right\rceil + 1, \text{ where } r_0 = \left\lceil \frac{\log\left(\ell^{(0)}_{\max}/\ell^{(0)}_{\min}\right)}{\log\left(1+c\right)} \right\rceil + 1.$$

We will show for any a, in any iteration $r \geq r_a$, we have $\ell^{(r)}_{\max}/\ell^{(r)}_{\min} \leq 1 + 2^{-a}$. First, we prove it for $a = 0$. If there exists some $r \leq r_0$ such that $\ell^{(r)}_{\max}/\ell^{(r)}_{\min} \leq 2$, then this also holds for $r \geq r_0$ by Lemma 13. Otherwise, for all $r \leq r_0$ we have $\ell^{(r)}_{\max}/\ell^{(r)}_{\min} > 2$. Then, using Lemma 14 with $\epsilon = 1$, we get $\ell^{(r+1)}_{\min} \geq (1+c)\cdot \ell^{(r)}_{\min}$. Therefore, $\ell^{(r_0)}_{\min} \geq (1+c)^{r_0} \cdot \ell^{(0)}_{\min} > \ell^{(0)}_{\max}$ by our choice of r_0. This contradicts Lemma 13, thereby showing that $\ell^{(r)}_{\max}/\ell^{(r)}_{\min} \leq 2$ for any $r \geq r_0$.

Now, we show the inductive case. Assume the inductive hypothesis that $\ell^{(r_{a-1})}_{\max}/\ell^{(r_{a-1})}_{\min} \leq 1 + 2^{-(a-1)}$. We will prove that $\ell^{(r_a)}_{\max}/\ell^{(r_a)}_{\min} \leq 1 + 2^{-(a)}$. The proof is similar to the base case of $a = 0$. If there exists some $r \leq r_a$ such that $\ell^{(r)}_{\max}/\ell^{(r)}_{\min} \leq 1 + 2^{-a}$, then this inequality also holds for any $r \geq r_a$ by Lemma 13. Otherwise, for all $r \leq r_a$ we have $\ell^{(r)}_{\max}/\ell^{(r)}_{\min} > 1 + 2^{-a}$. Then, for all $r_{a-1} \leq r \leq r_a$, using Lemma 14 with $\epsilon = 2^{-a}$, we have $\ell^{(r+1)}_{\min} \geq (1 + c\cdot 2^{-a})\cdot \ell^{(r)}_{\min}$. Therefore, $\ell^{(r_a)}_{\min} \geq (1 + c\cdot 2^{-a})^{r_a - r_{a-1}} \cdot \ell^{(r_{a-1})}_{\min}$. By our choice of r_a, this implies $\ell^{(r_a)}_{\min} > (1 + 2^{-(a-1)}) \cdot \ell^{(r_{a-1})}_{\min}$. By the induction hypothesis, this implies $\ell^{(r_a)}_{\min} > \ell^{(r_{a-1})}_{\max}$. But, this implies $\ell^{(r_a)}_{\max} > \ell^{(r_{a-1})}_{\max}$, which contradicts Lemma 13. Therefore,

$$\lim_{r\to\infty} \ell^{(r)}_{\max}/\ell^{(r)}_{\min} = 1,$$

and $\ell^*(P, G) = \lim_{r\to\infty} \ell^{(r)}_{\max}$. Moreover, by Lemma 12 this value is uniquely defined and attained by a unique (up to scaling) set of learned parameters.

◀

3.2 Weak Monotonicity of the Maximum and Minimum Loads in Algorithm 1: Proof of Lemma 13

For ease of description, we assume that G and $\mathbf{w}$ are normalized in the following sense:

$$\mathbf{w} = \mathbf{1}^m \text{ and } \sum_j g_{i,j} = 1.$$

This transformation is local to the current iteration, and only for the purpose of this proof. First, we explain why this change of notation is w.l.o.g. Suppose $\hat{G}, \hat{\mathbf{w}}$ represent the actual transformation matrix and learned parameters respectively. Now, we define G as follows:

$$g_{i,j} = \frac{\hat{g}_{i,j} \cdot \hat{w}_i}{\sum_{i' \in [m]} \hat{g}_{i',j} \cdot \hat{w}_{i'}},$$

and our new learned parameters is given by $\mathbf{1}^m$.

Note that the fractional allocation remains unchanged, i.e., $x_{i,j}(\hat{G}, \hat{w}) = x_{i,j}(G, \mathbf{1}^m) = g_{i,j}$, and therefore the loads are also unchanged: $\ell_i = \ell_i(P, \hat{G}, \hat{\mathbf{w}}) = \ell_i(P, G, \mathbf{1}^m) = \sum_{j \in [n]} g_{i,j} \cdot p_{i,j}$. Assume w.l.o.g. (by Observation 9) that $\gamma = \ell_1$, so $\hat{w}'_i = \frac{\hat{w}_i}{\ell_i} \cdot \ell_1$. In the normalized notation, the new parameters are $w'_i = \frac{\ell_1}{\ell_i}$. Again, the allocation is unchanged whether we use the original notation or the normalized one:

$$x_{i,j}(\hat{G}, \hat{\mathbf{w}}') = x_{i,j}(G, \mathbf{w}') = \frac{g_{i,j} \cdot w'_i}{\sum_{i' \in [m]} g_{i',j} \cdot w'_{i'}},$$

and we have, $\ell'_i = \ell_i(P, \hat{G}, \hat{\mathbf{w}'}) = \ell_i(P, G, \mathbf{w}')$.

The case of Two Agents. First, we consider the case of two agents here, i.e., $m = 2$. Later, we will show the reduction from general m to $m = 2$.

We have

$$\ell_1 = \sum_j g_{1,j} \cdot p_{1,j} \qquad \text{and} \qquad \ell_2 = \sum_j g_{2,j} \cdot p_{2,j},$$

and the parameter for the second agent after the update is given by: $w'_2 = \frac{\ell_1}{\ell_2}$ (note that $w'_1 = 1$).

Accordingly, the loads after the update are given by:

$$\ell'_1 = \sum_j p_{1,j} \cdot \frac{g_{1,j}}{g_{1,j} + w'_2 \cdot g_{2,j}} \qquad \text{and} \qquad \ell'_2 = \sum_j p_{2,j} \cdot \frac{w'_2 \cdot g_{2,j}}{g_{1,j} + w'_2 \cdot g_{2,j}}.$$

Assume w.l.o.g that $\ell_1 < \ell_2$. First, note that, from monotonicity (Observation 10) we have:

$$\ell'_2 \leq \ell_2 = {}^{\ell_{\max}}\!\big/\!\left(1+\frac{\ell_{\max}-\ell_2}{\tilde{p}_1}\right).$$

Next, we have to show that

$$\ell'_1 \leq {}^{\ell_{\max}}\!\big/\!\left(1+\frac{\ell_{\max}-\ell_1}{\tilde{p}_1}\right) = {}^{\ell_2}\!\big/\!\left(1+\frac{\ell_2-\ell_1}{\tilde{p}_1}\right). \tag{1}$$

The proof of the lower bound on ℓ'_1 is similar and is omitted for brevity.

We use the following standard inequality:

▶ **Fact 15** (Milne's Inequality [23]). *For any $a, b \in \mathbb{R}^n$, we have*

$$\sum_{j \in [n]} \frac{a_j \cdot b_j}{a_j + b_j} \quad \leq \quad \frac{\sum_{j \in [n]} a_j \cdot \sum_{j \in [n]} b_j}{\sum_{j \in [n]} (a_j + b_j)}.$$

In using this inequality, we set for any $j \in [n]$,

$$a_j = p_{1,j} \text{ and } b_j = p_{1,j} \cdot \left(\frac{f_j}{w'_2} - 1\right) \text{ where } f_j = g_{1,j} + w'_2 \cdot g_{2,j} = g_{1,j} + w'_2 \cdot (1 - g_{1,j}).$$

First, we calculate each term in Milne's inequality separately:

$$\sum_{j\in[n]} \frac{a_j \cdot b_j}{a_j + b_j} = \sum_{j\in[n]} p_{1,j} \cdot \frac{f_j - w'_2}{f_j} = \sum_{j\in[n]} p_{1,j} \cdot \frac{g_{1,j} + w'_2 \cdot g_{2,j} - w'_2}{f_j} = \sum_{j\in[n]} p_{1,j} \cdot \frac{g_{1,j} - w'_2 \cdot (1 - g_{2,j})}{f_j}$$

$$= \sum_{j\in[n]} p_{1,j} \cdot \frac{g_{1,j} - w'_2 \cdot g_{1,j}}{f_j} = \sum_{j\in[n]} p_{1,j} \cdot g_{1,j} \cdot \frac{1 - w'_2}{f_j} = \ell'_1 \cdot (1 - w'_2).$$

$$\sum_{j\in[n]} a_j = \tilde{p}_1.$$

$$\sum_{j\in[n]} b_j = \sum_{j\in[n]} p_{1,j} \cdot g_{1,j} \cdot \left(\frac{1}{w'_2} - 1\right) = \frac{\ell_1}{w'_2} - \ell_1 = \ell_2 - \ell_1 = \ell_2 \cdot (1 - w'_2).$$

Using Fact 15, we get

$$\ell'_1 \cdot (1 - w'_2) \leq \frac{\tilde{p}_1 \cdot \ell_2}{\ell_2 - \ell_1 + \tilde{p}_1} \cdot (1 - w'_2)$$

By our assumption that $\ell_1 < \ell_2$, and therefore $w'_2 < 1$. We now get Equation (1) by rearranging terms. This completes the proof for the lemma for the case of two agents.

4 Monotonicity and Convergence of Exponentiated Proportional Allocations

In this section, we prove the monotonicity and convergence of EP-allocations (Theorem 8).

First, we establish monotonicity of EP-allocations (first part of Theorem 8). We compare two EP-allocations with arbitrary learned parameters but different exponential constants. We show that with a larger exponent, at least one agent's load will be higher, regardless of the parameters used.

▶ **Lemma 16.** *Fix a weight matrix $P \in \mathbb{R}^{m\times n}_{>0}$. Let $\alpha, \alpha' \in \mathbb{R}$ such that $\alpha > \alpha'$. Now, for any two sets of learned parameters $\mathbf{w}_\alpha, \mathbf{w}_{\alpha'} \in \mathbb{R}^m_{>0}$, there exists an agent $k \in [m]$ such that*

$$\ell_k(P, \alpha, \mathbf{w}_\alpha) \geq \ell_k(P, \alpha', \mathbf{w}_{\alpha'}).$$

Proof. Let Δ denote the vector of differences of loads of the machines in the two allocations, namely $\Delta_i = \ell_i(P, \alpha, \mathbf{w}_\alpha) - \ell_i(P, \alpha', \mathbf{w}_{\alpha'})$. Our goal is to show that Δ has at least one nonnegative coordinate.

To show this, we define a vector in the positive orthant $\mathbf{c} \in \mathbb{R}^m_{>0}$ as follows:

$$c_i = \left(\frac{w_{\alpha,i}}{w_{\alpha',i}}\right)^{\frac{1}{\rho}}, \text{ where } \rho = \alpha - \alpha' > 0$$

and show that this vector $\mathbf{c}$ has a nonnegative inner product with the vector Δ. Note that this suffices since the inner product of a vector with all positive coordinates and one with all negative coordinates cannot be nonnegative. In other words, we want to show the following:

$$\sum_{i\in[m]} c_i \cdot (\ell_i(P, \alpha, w_\alpha) - \ell_i(P, \alpha', w_{\alpha'})) \geq 0. \quad (2)$$

Let us denote the fractional allocation of an item j in the two cases by $x_{i,j}$ and $x'_{i,j}$ respectively. Then, Equation (2) can be rewritten as

$$\sum_{i\in[m]} c_i \cdot \sum_{j\in[n]} p_{i,j} \cdot (x_{i,j} - x'_{i,j}) \geq 0.$$

Changing the order of the two summations, we rewrite further as

$$\sum_{j\in[n]} \left(\sum_{i\in[m]} c_i \cdot p_{i,j} \cdot (x_{i,j} - x'_{i,j}) \right) \geq 0.$$

We will prove this inequality separately for each item $j \in [n]$. Namely, we will show that

$$\sum_{i\in[m]} c_i \cdot p_{i,j} \cdot (x_{i,j} - x'_{i,j}) \geq 0, \text{ for every } j \in [n]. \tag{3}$$

Fix an item j. Since the item is fixed, we will drop j from the notation and define $\mathbf{u} \in \mathbb{R}^m$ as

$$u_i = p_i \cdot (x_i - x'_i).$$

So, we need to show that

$$\mathbf{c} \cdot \mathbf{u} \geq 0, \text{ i.e., } \sum_{i\in[m]} c_i \cdot u_i \geq 0. \tag{4}$$

We have

$$\begin{aligned}
\sum_i c_i \cdot u_i &= \sum_i c_i \cdot p_i \cdot \left(\frac{p_i^{\alpha} \cdot w_{\alpha,i}}{\sum_{i'} p_{i'}^{\alpha} \cdot w_{\alpha,i'}} - \frac{p_i^{\alpha'} \cdot w_{\alpha',i}}{\sum_{i'} p_{i'}^{\alpha'} \cdot w_{\alpha',i'}} \right) \\
&= \frac{1}{T} \cdot \sum_i c_i \cdot p_i \cdot \left(p_i^{\alpha} \cdot w_{\alpha,i} \cdot \left(\sum_{i'} p_{i'}^{\alpha'} \cdot w_{\alpha',i'} \right) - p_i^{\alpha'} \cdot w_{\alpha',i} \cdot \left(\sum_{i'} p_{i'}^{\alpha} \cdot w_{\alpha,i'} \right) \right) \\
&\qquad \text{where } T = \left(\sum_{i'} p_{i'}^{\alpha'} \cdot w_{\alpha',i'} \right) \cdot \left(\sum_{i'} p_{i'}^{\alpha} \cdot w_{\alpha,i'} \right).
\end{aligned}$$

Now, on the right hand side of the above equation, we replace α by $\alpha' + \rho$ and $w_{\alpha,i}$ by $w_{\alpha',i} \cdot c_i^{\rho}$ for every $i \in [m]$. This gives us:

$$\begin{aligned}
&\sum_i c_i \cdot u_i = \\
&\frac{1}{T} \sum_i c_i \cdot p_i \left(p_i^{\alpha'} \cdot p_i^{\rho} \cdot w_{\alpha',i} \cdot c_i^{\rho} \left(\sum_{i'} p_{i'}^{\alpha'} \cdot w_{\alpha',i'} \right) - p_i^{\alpha'} \cdot w_{\alpha',i} \left(\sum_{i'} p_{i'}^{\alpha'} \cdot p_{i'}^{\rho} \cdot w_{\alpha',i'} \cdot c_{i'}^{\rho} \right) \right) \\
&= \frac{1}{T} \sum_i b_i \left(a_i \cdot b_i^{\rho} \left(\sum_{i'} a_{i'} \right) - a_i \left(\sum_{i'} a_{i'} \cdot b_{i'}^{\rho} \right) \right), \\
&\qquad \text{where } a_i = w_{\alpha',i} \cdot p_i^{\alpha'} \text{ and } b_i = p_i \cdot c_i.
\end{aligned}$$

Rearranging the summations on the two terms on the right hand side, we get

$$\sum_i c_i \cdot u_i = \frac{1}{T} \cdot \left(\sum_{i'} a_{i'} \right) \cdot \sum_i a_i \cdot b_i^{\rho+1} - \frac{1}{T} \cdot \left(\sum_{i'} a_{i'} \cdot b_{i'}^{\rho} \right) \cdot \sum_i a_i \cdot b_i$$

Now, let $z_i = a_i^{1/2}$, and $y_i = a_i^{1/2} \cdot b_i^{\rho/2+1/2}$, and $\theta = \frac{|\rho-1|}{\rho+1}$. Then, we have

$$\begin{aligned} T \cdot \sum_i c_i \cdot u_i &= \left(\sum_{i'} a_{i'}\right) \cdot \left(\sum_i a_i \cdot b_i^{\rho+1}\right) - \left(\sum_{i'} a_{i'} \cdot b_{i'}^{\rho}\right) \cdot \left(\sum_i a_i \cdot b_i\right) \\ &= \left(\sum_{i'} z_{i'}^2\right) \cdot \left(\sum_i y_i^2\right) - \left(\sum_{i'} z_{i'}^{1+\theta} \cdot y_{i'}^{1-\theta}\right) \cdot \left(\sum_i z_i^{1-\theta} \cdot y_i^{1+\theta}\right). \end{aligned}$$

In the last equation, the first term follows directly from $a_{i'} = z_{i'}^2$ and $a_i \cdot b_i^{\rho+1} = y_i^2$. The second term is more complicated. There are two cases. If $\rho \le 1$, then $a_{i'} \cdot b_{i'}^{\rho} = z_{i'}^{1+\theta} \cdot y_{i'}^{1-\theta}$ and $a_i \cdot b_i = z_i^{1-\theta} \cdot y_i^{1+\theta}$ but if $\rho > 1$, then the roles get reversed and we get $a_{i'} \cdot b_{i'}^{\rho} = z_{i'}^{1-\theta} \cdot y_{i'}^{1+\theta}$ and $a_i \cdot b_i = z_i^{1+\theta} \cdot y_i^{1-\theta}$.

Now, note that $T \ge 0$. So, to establish $\sum_i c_i \cdot u_i \ge 0$, it suffices to show that the right hard side of the equation is nonnegative. We do so by employing Callebaut's inequality which we state below:

▶ **Fact 17** (Callebaut's Inequality [11]). *For any $y, z \in \mathbb{R}^n$ and $\theta \le 1$, we have*

$$\left(\sum_{i'} z_{i'}^2\right) \cdot \left(\sum_i y_i^2\right) \ge \left(\sum_{i'} z_{i'}^{1+\theta} \cdot y_{i'}^{1-\theta}\right) \cdot \left(\sum_i z_i^{1-\theta} \cdot y_i^{1+\theta}\right)$$

Note that we can apply Callebaut's inequality because $\rho \ge 0$ implies that $\theta \le 1$. This completes the proof of the lemma.

◀

▶ **Lemma 18.** *Given any weight matrix $P \in \mathbb{R}_{>0}^{m \times n}$ and any constant $\epsilon > 0$,*

(a) there exists an α (think of α as a sufficiently large negative number) and a corresponding set of parameters $\mathbf{w}_\alpha$ such that $\ell_i(P, \alpha, w_\alpha) \le (1+\epsilon) \cdot \ell^{\boldsymbol{MKS}}(P)$ for all $i \in [m]$.

(b) there exists an α' (think of α' as a sufficiently large positive number) and a corresponding set of parameters $\mathbf{w}_{\alpha'}$ such that $\ell_i(P, \alpha', w_{\alpha'}) \ge (1-\epsilon) \cdot \ell^{\boldsymbol{SNT}}(P)$ for all $i \in [m]$.

Using Lemma 18, we complete the proof of Theorem 8.

Proof of Theorem 8. First by Lemma 11, there exists $\mathbf{w}_\alpha^*$ and $\mathbf{w}_{\alpha'}^*$, such that, for all $i \in [m]$, $\ell_i(P, \alpha, \mathbf{w}_\alpha^*) = \ell^*(P, \alpha)$ and $\ell_i(P, \alpha', \mathbf{w}_{\alpha'}^*) = \ell^*(P, \alpha')$. Now, if $\ell^*(P, \alpha) < \ell^*(P, \alpha')$, it would contradict Lemma 16. And combining Lemma 16 and Lemma 18, we completed the proof the second part of Theorem 8.

◀

5 Noise Resilience: Handling Predictions with Error

In this section, we show the noise resilience of our algorithms, namely that we can handle errors in the learned parameters. First, we will show that for both objectives (MaxMin and MinMax), an η-approximate set of learned parameters yields an online algorithm with a competitive ratio of at least/at most η. Second, for the MinMax objective, we show that it is possible to improve the competitive ratio further in the following sense: using a set of learned parameters with a multiplicative error of η with respect to the optimal parameters, we can obtain a $O(\log \eta)$-competitive algorithm. (This was previously shown by Lattanzi *et al.* [17] but only for the special case of restricted assignment.) We also rule out a similar guarantee for the MaxMin objective, i.e., we show that using η-approximate learned parameters, an algorithm cannot hope to obtain a competitive ratio better than η/c for some constant c.

Finally, we show that noise-resilient bounds can be obtained not just for the MinMax and MaxMin objectives but also for any homogeneous monotone minimization or maximization objective function.

Formally, a weight vector $\mathbf{w}$ is η-approximate with respect to a weight vector to $\mathbf{w}^*$, if for any two agents $i, i' \in [m]$, $\frac{w_{i'}}{w_i} \le \eta \cdot \frac{w^*_{i'}}{w^*_i}$. First, we show a basic noise resilience property that holds for both the MinMax and MaxMin objectives:

▶ **Lemma 19.** *Fix a weight matrix $P \in \mathbb{R}^{m\times n}_{>0}$ and a transformation matrix $G \in \mathbb{R}^{m\times n}_{>0}$. For any two parameter vectors $\mathbf{w}^*, \mathbf{w} \in \mathbb{R}^m_{>0}$, such that $\mathbf{w}$ is η-approximate to $\mathbf{w}^*$, we have that for any agent k:*

$$\frac{\ell_k(P, G, \mathbf{w}^*)}{\eta} \le \ell_k(P, G, \mathbf{w}) \le \eta \cdot \ell_k(P, G, \mathbf{w}^*).$$

Proof. Let $y_{i,j} = x_{i,j}(G, \mathbf{w}^*)$ and $z_{i,j} = x_{i,j}(G, \mathbf{w})$ be the respective fractional allocations under proportional allocation using the transformation matrix G. For an agent i, let $\tau_i = w_i/w^*_i$. Then for any two agents i, k, we have that $1/\eta \le \tau_k/\tau_i \le \eta$. We have, $\frac{y_{i,j}}{z_{i,j}} = \sum_{i'\in[m]} \frac{\tau_{i'}}{\tau_i} \cdot y_{i',j}$. Therefore,

$$\frac{y_{i,j}}{z_{i,j}} = \sum_{i'\in[m]} \frac{\tau_{i'}}{\tau_i} \cdot y_{i',j} \ge \sum_{i'\in[m]} \frac{1}{\eta} \cdot y_{i',j} = \frac{1}{\eta} \cdot \sum_{i'\in[m]} y_{i',j} = \frac{1}{\eta}, \text{ and}$$

$$\frac{y_{i,j}}{z_{i,j}} = \sum_{i'\in[m]} \frac{\tau_{i'}}{\tau_i} \cdot y_{i',j} \le \sum_{i'\in[m]} \eta \cdot y_{i',j} = \eta \cdot \sum_{i'\in[m]} y_{i',j} = \eta.$$

Hence, $y_{i,j}/\eta \le z_{i,j} \le y_{i,j} \cdot \eta$. Finally, the lemma hold by summing over all items. ◀

The next theorem follows immediately by using a proportional allocation according to the parameter vector $\tilde{\mathbf{w}}$:

▶ **Theorem 20.** *Fix any $P, G \in \mathbb{R}^{m\times n}_{>0}$. Let $\mathbf{w}$ be a learned parameter vector that gives a solution of value γ for the MaxMin (resp., MinMax) objective using proportional allocation. Let $\tilde{\mathbf{w}}$ be η-approximate to $\mathbf{w}$ for some $\eta > 1$. Then, there exists an online algorithm that given $\tilde{\mathbf{w}}$ generates a solution with value at least $\Omega(\gamma/\eta)$ (resp., at most $O(\eta\gamma)$).*

In particular, if $\mathbf{w}$ is the *optimal* learned parameter vector in the above theorem and $\tilde{\mathbf{w}}$ is an η-approximation to it, then we obtain a competitive ratio of $\Omega(1/\eta)$.

The rest of this section focuses on the MinMax objective for which we can obtain an improved bound. In the next lemma, we establish an upper bound on the load, using Lemma 19 and monotonicity.

▶ **Lemma 21.** *Fix a weight matrix $P \in \mathbb{R}^{m\times n}_{>0}$ and a transformation matrix $G \in \mathbb{R}^{m\times n}_{>0}$. For any two parameter vectors $\mathbf{w}^*, \mathbf{w} \in \mathbb{R}^m_{>0}$ such that there exists an agent $k \in [m]$ for which $w^*_k/2 \le w_k \le w^*_k$ and for all other agents $i \ne k$, we have $w_i \ge w^*_i/2$, then the following holds: $\ell_k(P, G, \mathbf{w}) \le 2 \cdot \ell_k(P, G, \mathbf{w}^*)$.*

Proof. Define $\mathbf{w}'$ where $w'_k = w^*_k$ (i.e., the maximum in its allowed range) and $w'_i = w^*_i/2$ for all $i \ne k$ (i.e., the minimum in their allowed ranges). Now, by monotonicity (Observation 10), we have $x_{k,j}(G, \mathbf{w}) \le x_{k,j}(G, \mathbf{w}')$, and therefore, $\ell_k(P, G, \mathbf{w}) \le \ell_k(P, G, \mathbf{w}')$. Note that for $\mathbf{w}'$, for any two agents i_1, i_2, $\frac{w_{i_1}}{w_{i_2}} \le 2 \cdot \frac{w^*_{i_1}}{w^*_{i_2}}$. Therefore, by Lemma 19, we have $\ell_k(P, G, \mathbf{w}') \le 2 \cdot \ell_k(P, G, \mathbf{w}^*)$. By combining the two inequalities, we have $\ell_k(P, G, \mathbf{w}) \le \ell_k(P, G, \mathbf{w}') \le 2 \cdot \ell_k(P, G, \mathbf{w}^*)$, as required. ◀

Algorithm 2 The online algorithm with predictions.

- Let $\hat{\mathbf{w}}$ a prediction vector and T is the offline optimal objective for the MinMax problem.
- Initialize: $\ell_i \leftarrow 0$ and $\tilde{w}_i \leftarrow \hat{w}_i$, for all $i \in [m]$

For each item j:

- Compute $x_{i,j} = \frac{f(p_{i,j}) \cdot \tilde{w}_i}{\sum_{i' \in [m]} f(p_{i',j}) \cdot \tilde{w}_{i'}}$
- $\ell_i \leftarrow \ell_i + p_{i.j} \cdot x_{i,j}$, for all $i \in [m]$
- If exists $i \in [m]$, s.t. $\ell_i > 2 \cdot T$
  - Set $\ell_i \leftarrow 0$
  - Update $\tilde{w}_i \leftarrow \tilde{w}_i/2$

Let us denote the predicted learned parameter vector that is given offline to the MinMax algorithm by $\hat{\mathbf{w}}$. We also assume that the algorithm knows the optimal objective value T. By scaling, we assume w.l.o.g that $\tilde{\mathbf{w}}$ is coordinate-wise larger than the optimal learned parameter vector $\mathbf{w}$. The algorithm uses a learned parameter vector $\hat{\mathbf{w}}$ that is iteratively refined, starting with $\hat{\mathbf{w}} = \tilde{\mathbf{w}}$ (see Algorithm 2). In each iteration, the current parameter vector $\hat{\mathbf{w}}$ is used to determine the assignment using proportional allocation until an agent's load in the current phase exceeds $2T$. If this happens for any agent i, then the algorithm halves the value of $\hat{w}_i$, starts a new phase for agent i, and continues doing proportional allocation with the updated learned parameter vector $\hat{\mathbf{w}}$.

▶ **Theorem 22.** *Fix any $P, G \in \mathbb{R}^{m \times n}_{>0}$. Let $\mathbf{w}$ be a learned parameter vector that gives a fractional solution with maximum load T using proportional allocation. Let $\tilde{\mathbf{w}}$ be an η-approximate prediction for $\mathbf{w}$. Then there exists an online algorithm that given $\tilde{\mathbf{w}}$ generates a fractional assignment of items to agents with maximum load at most $O(T \log \eta)$.*

Proof. By the algorithm's definition, an agent's total load is at most $2T$ times the number of phases for the agent. We show that for any agent i, the parameter $\tilde{w}_i$ is always at least $w_i/2$. This immediately implies that the number of phases for machine i is $O(\log \eta)$, which in turn establishes the theorem.

Suppose, for contradiction, in some phase for agent k, we have $\tilde{w}_k < w_k/2$. Moreover, assume w.l.o.g. that agent k is the first agent for which this happens. Clearly, by the algorithm definition, there is a preceding phase for agent k when $\tilde{w}_k < w_k$. Note that, in this entire preceding phase, we have $w_k > \tilde{w}_k \geq w_k/2$, and for all $i \neq k$, $\tilde{w}_i \geq w_i/2$ (by our assumption that k is the first agent to have a violation). However, by Lemma 21, the load of agent k in the preceding phase would be at most $2T$. This contradicts the fact that the algorithm started a new phase for agent k when its load exceeded $2T$ in the preceding phase. ◀

In the full version of the paper, we show that the bounds obtained above for the MaxMin and MinMax objectives are asymptotically tight.

6 Learnability of the Parameters

We consider the learning model introduced by [18], and show that under this model, the parameter vector $\mathbf{w}$ can be learned efficiently from sampled instances. Specifically, we consider the following model: the jth item (i.e., the values of $\mathbf{p}_j = (p_{i,j} : i \in [m])$ is independently sampled from a (discrete) distribution $\mathcal{D}_j$. In other words, the matrix P of utilities is sampled from $\mathcal{D} = \times_j \mathcal{D}_j$.

We set up the model for the MaxMin objective; the setup for the MinMax objective is very similar and is omitted for brevity. Let $T = \mathbb{E}_{P\sim\mathcal{D}}[\ell^{\mathbf{SNT}}(P)]$ be the expected value of the MaxMin objective in the optimal solution for an instance $\ell^{\mathbf{SNT}}(P)$ drawn from $\mathcal{D}$. Morally, we would like to say that we can obtain a vector $\mathbf{w}$ that gives a nearly optimal solution (in expectation) using proportional allocation (i.e., a MaxMin objective of $(1-\epsilon)\cdot T$ in expectation for some error parameter ϵ) using a bounded (as a function of ϵ) number of samples. Similar to [18], we need the following assumption:

Small Items Assumption: Conceptually, this assumption states that each individual item has a small utility compared to the overall utility of any agent in an optimal solution. Precisely, we need $p_{i,j} \le \frac{T}{\zeta}$ for every $i \in [m], j \in [n]$ for some value $\zeta = \Theta\left(\frac{\log m}{\epsilon^2}\right)$.

Our main theorem in this section for the MaxMin and MinMax objectives are:

▶ **Theorem 23.** *Fix an $\epsilon > 0$ for which the small items assumption holds. Then, there is an (learning) algorithm that samples $O(\frac{m}{\log m}\cdot\log\frac{m}{\epsilon})$ independent instances from $\mathcal{D}$ and outputs (with high probability) a prediction vector $\mathbf{w}$ such that using $\mathbf{w}$ in the proportional allocation scheme gives a* MaxMin *objective of at least $(1-\Omega(\epsilon))\cdot T$ in expectation over instances $P \sim \mathcal{D}$.*

▶ **Theorem 24.** *Fix an $\epsilon > 0$ for which the small items assumption holds. Then, there is an (learning) algorithm that samples $O(\frac{m}{\log m}\cdot\log\frac{m}{\epsilon})$ independent instances from $\mathcal{D}$ and outputs (with high probability) a prediction vector $\mathbf{w}$ such that using $\mathbf{w}$ in the proportional allocation scheme gives a* MinMax *objective of at most $(1+O(\epsilon))T$ in expectation over instances $P \sim \mathcal{D}$.*

Importantly, the description of the entries of $\mathbf{w}$ in Theorem 23 and Theorem 24 are bounded. Specifically, let us define $\mathbf{NET}(m,\epsilon) \subseteq \mathbb{R}^m_{>0}$ as follows: (a) for the MaxMin objective, $\mathbf{w} \in \mathbf{NET}(m,\epsilon)$ if there exist vectors $\mathbf{u}, \delta \in \mathbb{R}^m_{>0}$ such that $w_i = \frac{\delta_i}{u_i^\alpha}$ and $u_i, \delta_i \in \left\{\left(\frac{1}{1-\epsilon}\right)^r : r \in [K]\right\}$ for some $K = O(\frac{m}{\epsilon}\log\frac{m}{\epsilon})$, and (b) for the MinMax objective, $\mathbf{w} \in \mathbf{NET}'(m,\epsilon)$ if there exist vectors $\mathbf{u}, \delta \in \mathbb{R}^m_{>0}$ such that $w_i = \frac{\delta_i}{u_i^\alpha}$ and $u_i, \delta_i \in \{(1+\epsilon)^r : r \in [K]\}$ for some $K = O(\frac{m}{\epsilon}\log\frac{m}{\epsilon})$. The vectors $\mathbf{w}$ produced by the learning algorithm in Theorem 23 and Theorem 24 will satisfy $\mathbf{w} \in \mathbf{NET}(m,\epsilon)$ and $\mathbf{w} \in \mathbf{NET}'(m,\epsilon)$ in the respective cases.

Proof Idea for Theorem 23 and Theorem 24. Recall that in PAC theory, the number of samples needed to learn a function from a family of N functions is about $O(\log N)$. Indeed, restricting $\mathbf{w}$ to be in the class $\mathbf{NET}(m,\epsilon)$ or $\mathbf{NET}'(m,\epsilon)$ serves this role of limiting the hypothesis class to a finite, bounded set since $|\mathbf{NET}(m,\epsilon)| = |\mathbf{NET}'(m,\epsilon)| = K^{2m}$ where $K = O(\frac{m}{\epsilon}\log\frac{m}{\epsilon})$. Using standard PAC theory, this implies that using about $O(m\log K) = O(m\cdot\log\frac{m}{\epsilon})$ samples, we can learn the "best" vector in $\mathbf{NET}(m,\epsilon)$ or $\mathbf{NET}'(m,\epsilon)$ depending on whether we have the MaxMin or MinMax objective. Our main technical work is to show that this "best" vector produces an approximately optimal solution when used in proportional allocation. We state this lemma next:

▶ **Lemma 25.** *Fix any P. For the* MaxMin *objective, there exists a learned parameter vector $\mathbf{w} \in \boldsymbol{NET}(m,\epsilon)$ which when used in* EP*-allocation gives a $1-\Omega(\epsilon)$ approximation. For the* MaxMin *objective, there exists a learned parameter vector $\mathbf{w}' \in \boldsymbol{NET}'(m,\epsilon)$ which when used in* EP*-allocation gives a $1+O(\epsilon)$ approximation.*

7 Generalization to Well-Behaved Objectives

We first generalize Theorem 6 to all well-behaved functions.

▶ **Theorem 26.** *Fix any instance of an online allocation problem with divisible items where the goal is to maximize or minimize a monotone homogeneous objective function. Then, there exists an online algorithm and a learned parameter vector in $\mathbb{R}^m_{>0}$ that achieves a competitive ratio of $1-\epsilon$ (for maximization) or $1+\epsilon$ (for minimization).*

Proof. Fix an objection function f and a matrix $P \in \mathbb{R}^{m\times n}_{>0}$. Let ℓ_i^f denote the load of agent i in an optimal solution for objective function f. Also, let $x_{i,j}$ denote the fraction of item j assigned to agent i in this optimal solution. Now, consider the matrix $\tilde{P}$, where $\tilde{p}_{i,j} = \frac{p_{i,j}}{\ell_i^f}$. By the monotonicity property of f, the optimal objective value for $\tilde{P}$ is 1. Therefore, by Theorem 8, there exist α and $\tilde{\mathbf{w}}$, such that using an EP-allocation, we get $\ell^*(\tilde{P}, \alpha, \tilde{\mathbf{w}}) \geq 1-\epsilon$ for maximization and $\ell^*(\tilde{P}, \alpha, \tilde{\mathbf{w}}) \leq 1+\epsilon$ for minimization. Let $x^*_{i,j}$ be the fraction of item j assigned to agent i in this approximate solution. By the definition of EP-allocation, $x^*_{i,j}$ is proportional to $\tilde{p}^\alpha_{i,j} \cdot \tilde{w}_i = \left(\frac{p_{i,j}}{\ell_i^f}\right)^\alpha \cdot \tilde{w}_i = p^\alpha_{i,j} \cdot \frac{\tilde{w}_i}{(\ell_i^f)^\alpha}$. Thus, if we define $\mathbf{w}$ such that $w_i = \frac{\tilde{w}_i}{(\ell_i^f)^\alpha}$, then the corresponding EP-allocation gives a $(1-\epsilon)$-approximate solution for maximization and $(1+\epsilon)$-approximate solution for minimization. ◀

7.1 Noise Resilience

Next, we consider noise resilience for well-behaved functions, i.e., we generalize Theorem 20 to all well-behaved objective functions. This follows immediately from Lemma 19 and the observation that if all loads are scaled by η, then the objective value for a well-behaved objective is also scaled by η. We state this generalized theorem below:

▶ **Theorem 27.** *Fix any $P, G \in \mathbb{R}^{m\times n}_{>0}$ and any monotone, homogeneous function f. Let $\mathbf{w}$ be a learned parameter vector that gives a solution of objective value γ using* EP*-allocation. Let $\tilde{\mathbf{w}}$ be η-approximate to $\mathbf{w}$ for some $\eta > 1$. Then, the* EP*-allocation for $\tilde{\mathbf{w}}$ gives a solution with value at least γ/η for maximization and at most $\eta\gamma$ for minimization.*

7.2 Learnability

Finally, we consider learnability of parameters for well-behaved functions, i.e., we generalize Theorem 23 and by assuming additional property of the objective function:

- For a maximization objective f, we need *superadditivity*: $f(\sum_r \ell_r) \geq \sum_r f(\ell_r)$.
- For a minimization objective f, we need *subadditivity*: $f(\sum_r \ell_r) \leq \sum_r f(\ell_r)$.

▶ **Theorem 28.** *Let f be a well-behaved function. If f is superadditive, the following theorem holds for maximization of f, while if f is subadditive, the following theorem holds for minimization of f. Let T be the expectation of the maximum value of f over instances sampled from $\mathcal{D}$. Fix an $\epsilon > 0$ for which the small items assumption holds. Then, there is an (learning) algorithm that samples $O(\frac{m}{\log m} \cdot \log \frac{m}{\epsilon})$ independent instances from $\mathcal{D}$ and outputs (with high probability) a prediction vector $\mathbf{w}$ such that using $\mathbf{w}$ in the* EP*-allocation gives a value of f that is at least $(1-\Omega(\epsilon)) \cdot T$ for maximization and at most $(1+O(\epsilon)) \cdot T$ for minimization, in expectation over instances $P \sim \mathcal{D}$.*

8 Conclusion and Future Directions

In this paper, we gave a unifying framework for designing near-optimal algorithm for fractional allocation problems for essentially all well-studied minimization and maximization objectives in the literature. The existence of this overarching framework is rather surprising because the corresponding worst-case problems exhibit a wide range of behavior in terms of the best competitive ratio achievable, as well as the techniques required to achieve those bounds. It would be interesting to gain further understanding of the optimal learned parameters introduced in this paper. One natural conjecture is that these are optimal dual variables for a suitably defined convex program (for instance, such convex programs are known for restricted assignment and b-matching [1]). Another interesting direction of future work would be to explore other polytopes beyond the simple assignment polytope considered in this paper, such as that corresponding to congestion minimization problems.

References

1. Shipra Agrawal, Morteza Zadimoghaddam, and Vahab Mirrokni. Proportional allocation: Simple, distributed, and diverse matching with high entropy. In *International Conference on Machine Learning*, pages 99–108. PMLR, 2018.
2. Antonios Antoniadis, Themis Gouleakis, Pieter Kleer, and Pavel Kolev. Secretary and online matching problems with machine learned advice. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020.
3. James Aspnes, Yossi Azar, Amos Fiat, Serge A. Plotkin, and Orli Waarts. On-line routing of virtual circuits with applications to load balancing and machine scheduling. *J. ACM*, 44(3):486–504, 1997.
4. Baruch Awerbuch, Yossi Azar, Edward F. Grove, Ming-Yang Kao, P. Krishnan, and Jeffrey Scott Vitter. Load balancing in the l_p norm. In *36th Annual Symposium on Foundations of Computer Science*, pages 383–391. IEEE Computer Society, 1995.
5. Yossi Azar, Stefano Leonardi, and Noam Touitou. Flow time scheduling with uncertain processing time. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1070–1080. ACM, 2021.
6. Yossi Azar, Stefano Leonardi, and Noam Touitou. Distortion-oblivious algorithms for minimizing flow time. In *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022*, pages 252–274. SIAM, 2022.
7. Yossi Azar, Joseph Naor, and Raphael Rom. The competitiveness of on-line assignments. *J. Algorithms*, 18(2):221–237, 1995.
8. Étienne Bamas, Andreas Maggiori, Lars Rohwedder, and Ola Svensson. Learning augmented energy minimization via speed scaling. In *Advances in Neural Information Processing Systems 33, NeurIPS 2020*, 2020.
9. Siddhartha Banerjee, Vasilis Gkatzelis, Artur Gorokh, and Billy Jin. Online nash social welfare maximization with predictions. In *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022*, pages 1–19. SIAM, 2022.
10. Siddharth Barman, Arindam Khan, and Arnab Maiti. Universal and tight online algorithms for generalized-mean welfare. In *Thirty-Sixth AAAI Conference on Artificial Intelligence*, pages 4793–4800. AAAI Press, 2022.
11. DK Callebaut. Generalization of the cauchy-schwarz inequality. *Journal of mathematical analysis and applications*, 12(3):491–494, 1965.
12. Ioannis Caragiannis. Better bounds for online load balancing on unrelated machines. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008*, pages 972–981. SIAM, 2008.

13 Justin Y. Chen and Piotr Indyk. Online bipartite matching with predicted degrees. *CoRR*, 2021. arXiv:2110.11439.

14 MohammadTaghi Hajiaghayi, MohammadReza Khani, Debmalya Panigrahi, and Max Springer. Online algorithms for the santa claus problem. In *Advances in Neural Information Processing Systems 35, NeurIPS 2022*, 2022.

15 Sungjin Im, Ravi Kumar, Mahshid Montazer Qaem, and Manish Purohit. Non-clairvoyant scheduling with predictions. In *SPAA '21: 33rd ACM Symposium on Parallelism in Algorithms and Architectures, Virtual Event, USA, 6-8 July, 2021*, pages 285–294. ACM, 2021.

16 Ravi Kumar, Manish Purohit, Aaron Schild, Zoya Svitkina, and Erik Vee. Semi-online bipartite matching. In *10th Innovations in Theoretical Computer Science Conference, ITCS 2019*, volume 124 of *LIPIcs*, pages 50:1–50:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

17 Silvio Lattanzi, Thomas Lavastida, Benjamin Moseley, and Sergei Vassilvitskii. Online scheduling via learned weights. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020*, pages 1859–1877. SIAM, 2020.

18 Thomas Lavastida, Benjamin Moseley, R. Ravi, and Chenyang Xu. Learnable and instance-robust predictions for online matching, flows and load balancing. In *29th Annual European Symposium on Algorithms, ESA 2021*, volume 204 of *LIPIcs*, pages 59:1–59:17, 2021.

19 Thomas Lavastida, Benjamin Moseley, R. Ravi, and Chenyang Xu. Using predicted weights for ad delivery. In *Applied and Computational Discrete Algorithms, ACDA 2021*, 2021.

20 Shi Li and Jiayi Xian. Online unrelated machine load balancing with predictions revisited. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021*, 2021.

21 Thodoris Lykouris and Sergei Vassilvitskii. Competitive caching with machine learned advice. *J. ACM*, 68(4):24:1–24:25, 2021.

22 Mohammad Mahdian, Hamid Nazerzadeh, and Amin Saberi. Online optimization with uncertain information. *ACM Trans. Algorithms*, 8(1):2:1–2:29, 2012.

23 EA Milne. Note on rosseland's integral for the stellar absorption coefficient. *Monthly Notices of the Royal Astronomical Society*, 85:979–984, 1925.

24 Michael Mitzenmacher. Scheduling with predictions and the price of misprediction. In *11th Innovations in Theoretical Computer Science Conference, ITCS 2020*, volume 151 of *LIPIcs*, pages 14:1–14:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

25 Michael Mitzenmacher and Sergei Vassilvitskii. Algorithms with predictions. In *Beyond the Worst-Case Analysis of Algorithms*, pages 646–662. Cambridge University Press, 2020.

26 Michael Mitzenmacher and Sergei Vassilvitskii. Algorithms with predictions. *Commun. ACM*, 65(7):33–35, 2022.

27 Manish Purohit, Zoya Svitkina, and Ravi Kumar. Improving online algorithms via ML predictions. In *Advances in Neural Information Processing Systems 31, NeurIPS 2018*, 2018.