Online Primal-Dual Algorithms with Predictions for Non-Linear Packing/Covering Problems

Nguyễn Kim Thắng
IBISC, University Paris-Saclay, France

Abstract

The domain of online algorithms with predictions has been extensively studied with new algorithms designed for different problems of scheduling, caching (paging), clustering, ski rental, etc. Recently, Bamas et al., aiming for an unified method, has provided a primal-dual framework for linear covering problems. They extended the online primal-dual method by incorporating predictions in order to achieve a performance beyond the worst-case case analysis. In this paper, we consider this research line and present a framework to design algorithms with predictions for non-linear problems with packing/covering contraints. We illustrate the applicability of our framework through several important problems such as submodular maximization/minimization, energy minimization, load balancing.

1 Introduction

Online computation [9] is a well-established field in theoretical computer science. In online computation, inputs are released one-by-one in form of a request sequence. Requests arrive over time and after observing the requests, an algorithm needs to perform an irrevocable action without any information about the future ones. The goal is to design performant algorithms with respect to certain objectives without future knowledge. In online computation, the performance of an online algorithm is typically measured by the competitive ratio defined as the worst ratio between the objective of an algorithm and that of the optimal solution. Intuitively, the competitive ratio measures the price of not knowing the future requests.

The traditional worst-case analysis is a powerful framework in algorithm design and is central in the development of algorithms. However, there are several pitfalls when one uses that model to choose algorithms for practical use. Summarizing an algorithm performance by a pathological worst case can overestimate its performance on most inputs. Many practically well-performed algorithms admit mediocre theoretical guarantee whereas theoretically established ones behave poorly even on simple instances in practice. The need for theories that explain and advise algorithms for problems is well-known and is clearly identified as an important direction with high priority in algorithmic community [33, 34].

Much of the present and future work going beyond worst-case paradigm is motivated by spectacular advances in machine learning (ML). In particular, the capability of ML methods in predicting patterns of future requests would provide useful information to be exploited by online algorithms. A general framework for incorporating ML predictions into algorithm design in order to achieve a better performance than the worst-case one is formally introduced by [27]. This is rapidly followed by work studying online algorithms with predictions [30] in a large spectre of problems such as scheduling [26, 29], caching (paging) [27, 31, 2], ski rental [19, 25], etc. An issue in designing online algorithms with predictions is the lack of knowledge on the quality of predictive information

whereas still dealing with irrevocable decision making. On the one hand, if the predictions are accurate, one expects to achieve a good performance as in the offline setting where the input are completely given in advance. On the other hand, if the predictions are misleading, one still has to maintain a competitive solution as in the online setting where no predictive information is available.

This issue has been resolved in different work (for exemple the aforementioned work) by subtly incorporating predictions and exploiting specific problems' structures. Recently, toward an unified technique beyond different ad-hoc approaches, [7] presented a primal-dual method to design online algorithms with predictions. Primal-dual is an elegant and powerful technique in the design of algorithms [38], especially in online algorithms [10]. [7] considered the primal-dual approach formalized by [10] and extended into the design of online algorithms with predictions for problems with linear objective and covering constraints. In this paper, we follow the primal-dual perspective and show a further extension of the primal-dual method with predictions into problems with non-linear objective and covering/packing constraints.

1.1 Model and Preliminaries

We consider the setting presented in [7] (with several definitions rooted in [27, 25]). In the setting, we are given a prediction oracle $\mathcal{P}$ and a parameter $\eta \in (0, 1]$ which characterizes the confidence on the prediction oracle with smaller values of η representing higher confidence. At the arrival of a request, an algorithm needs to make an irrevocable decision in order to satisfy the request (under constraints of different natures) based on the prediction of oracle $\mathcal{P}$, the confidence parameter η and the current solution as well as the history of released requests. We say that an algorithm $\mathcal{A}$ is $C(\eta)$-*consistent* and $R(\eta)$-*robust* if for every instance I,

$$\begin{aligned} \mathcal{A}(I) &\geq \max\{C(\eta) \cdot \mathcal{P}(I), R(\eta) \cdot \mathcal{O}(I)\} \qquad \text{(maximization problem)}, \\ \mathcal{A}(I) &\leq \min\{C(\eta) \cdot \mathcal{P}(I), R(\eta) \cdot \mathcal{O}(I)\} \qquad \text{(minimization problem)}, \end{aligned}$$

where $\mathcal{A}(I), \mathcal{P}(I), \mathcal{O}(I)$ are the objective values of algorithm $\mathcal{A}$, prediction $\mathcal{P}$ and optimal solution $\mathcal{O}$ on instance I, respectively. When the prediction $\mathcal{P}$ provides an infeasible solution, conventionally $\mathcal{P}(I)$ gets value $-\infty$ and $+\infty$ for maximization and minimization problem, respectively. It is desirable that when η approaches 0 (high confidence to the prediction), $C(\eta)$ tends to 1; and when η approaches 0 (high confidence to the prediction), $R(\eta)$ tends to the same competitive ratio as in the classic online setting.

In high level, the primal-dual method consists in formulating a given problem by a mathematical (primal) program, usually a linear program, and considering also the corresponding dual. At the arrival of a request, an algorithm updates the fractional solutions to both the primal and dual in order to to maintain their feasibility. Then, the competitive ratio of the algorithm is established by showing that every time an update of primal and dual solutions is made, the increase of the primal can be bounded by that of the dual up to some desired factor.

In the paper, similar as [7], we focus on constructing fractional solution which is the main step in the primal-dual method. In order to derive algorithms for specific problems, online rounding schemes are needed and in many problems, such schemes already exist. We will provide references for the rounding schemes which are appropriate to specific problems studied in the applications.

1.2 Primal-Dual Framework for Maximization under Packing Constraints

Packing Problem. Let $\mathcal{E}$ be a set of n resources and let $f : \{0, 1\}^n \to \mathbb{R}^+$ be an *arbitrary* non-decreasing function. Let $x_e \in \{0, 1\}$ be a variable indicating whether resource e is selected. The set of packing constraints $\sum_e b_{i,e} x_e \leq 1 \ \forall i$ (including $x_e \leq 1 \ \forall e$) are given in advance and

resources e are revealed online one-by-one. At the arrival of resource e, one receives a prediction $x_e^{\texttt{pred}} \in \{0,1\}$ and needs to make a decision on x_e while maintaining $\mathbf{x} = (x_e)_{e\in\mathcal{E}}$ feasible to the set of constraints. The objective of the problem is to maximize $f(\mathbf{x})$. In this problem, we seek a fractional solution that is consistent to the prediction and robust to the optimal offline solution.

Approach and Contribution. We follow the primal-dual approach presented in [36] to design competitive algorithms for fractional non-linear packing problems. Given a function $f : \{0,1\}^n \to \mathbb{R}^+$, its *multilinear extension* $F : [0,1]^n \to \mathbb{R}^+$ is defined as $F(\mathbf{x}) := \sum_S \prod_{e\in S} x_e \prod_{e\notin S}(1-x_e)\cdot f(\mathbf{1}_S)$ where $\mathbf{1}_S$ is the characteristic vector of S (i.e., the e^{th}-component of $\mathbf{1}_S$ equals 1 if $e \in S$ and equals 0 otherwise). Alternatively, $F(\mathbf{x}) = \mathbb{E}\big[f(\mathbf{1}_T)\big]$ where T is a random set such that a resource e appears independently in T with probability x_e. Note that $F(\mathbf{1}_S) = f(\mathbf{1}_S)$. Consider the following crucial notion.

Definition 1 ([36]) *A differentiable function $F : [0,1]^n \to \mathbb{R}^+$ is* (λ,μ)-max-locally-smooth *if for any set $S \subset \mathcal{E}$, and for every vector $\mathbf{x} \in [0,1]^n$, the following inequality holds:*

$$\sum_{e\in S} \nabla_e F(\mathbf{x}) \geq \lambda F\big(\mathbf{1}_S\big) - \mu F\big(\mathbf{x}\big).$$

where $\nabla_e F(\mathbf{x})$ denotes $\partial F(\mathbf{x})/\partial x_e$.

Note that the (λ,μ)-smoothness notion here is not similar to the standard notion of smoothness of functions in convex optimization. The former, introduced in [36], is related to the definition of smooth games [32] in the context of algorithmic game theory. Intuitively, given a (λ,μ)-max-locally-smooth function, the quantity $\frac{\lambda}{1-\mu}$ measures how far the function from being linear. If a function is linear then it is $(1,0)$-max-locally-smooth.

In our approach, we incorporate the predictive information into the primal-dual approach by modifying the coefficients of the constraints. The algorithm, together with this modification, guarantees that: (i) if the confidence on the prediction is high (η is close to 0) then value of variables corresponding to the resources selected by the prediction would get a large value; (ii) and inversely, if the confidence on the prediction is low (η is close to 1) then the variables would be constructed similarly to the classic online primal-dual method. The construction follows the multiplicative weights update based on the gradient of the multilinear extension [36], which generalizes the multiplicative update in [11, 3]. Using the max-local-smoothness notion, we show the feasibility of our primal/dual solutions (even when the prediction provides an infeasible one) and also to establish the performance of the algorithm, which is determined in terms of the max-locally-smoothness and confidence parameters.

Theorem 1 *Let F be the multilinear extension of the objective function f. Denote the row sparsity $d := \max_i |\{b_{ie} : b_{ie} > 0\}|$ and $\rho := \max_i \max_{e,e':b_{ie'}>0} b_{ie}/b_{ie'}$. Assume that F is (λ,μ)-max-locally-smooth for some parameters $\lambda > 0$ and μ. Then, for every $0 < \eta \leq 1$, there exists a $r(\eta)$-consistent and $r(\eta)\cdot\big(\frac{\lambda}{2\ln(1+d\rho/\eta)+\mu}\big)$-robust algorithm for the fractional packing problem where $r(\eta) = \min_{\mathbf{0}\leq\mathbf{u}\leq\mathbf{1}} F(\frac{\mathbf{u}}{1+\eta})/F(\mathbf{u})$.*

Applications. In the following, we describe the applications of our framework to different classes of problems.

Linear objectives. Numerous combinatorial problems can be formalized by linear programs with packing constraints to which our framework applies. When function f can be expressed as a linear function, the smooth parameters of F are $\lambda = 1$ and $\mu = 0$. In this case, Algorithm 1 guarantees the consistency $C(\eta) = \frac{1}{1+\eta}$ and the robustness $R(\eta) = \frac{1}{2(1+\eta)\ln(1+d\rho/\eta)}$. Note that in this case, there is no need to consider the multilinear extension F of f but the function f directly in the algorithm.

Submodular objectives. Submodular maximization constitutes a major research agenda and has been widely studied in optimization, machine learning [5, 24, 8] and algorithm design [16, 12]. Using our framework, we derive an algorithm that yields a $(1-\eta)$-consistent and $\Omega\big(\frac{1}{2\ln(1+d\rho/\eta)}\big)$-robust fractional solution for online submodular maximization with packing constraints. This performance is similar to that of linear functions up to constant factors. Note that using the online contention resolution rounding schemes [17], that generalizes the offline counterpart [14], one can obtain randomized algorithms for several specific constraint polytopes such as knapsack polytopes, matching polytopes and matroid polytopes.

1.3 Primal-Dual Framework for Maximization under Covering Constraints

Covering Problem. Let $\mathcal{E}$ be a set of n resources and let $f : \{0,1\}^n \to \mathbb{R}^+$ be an *arbitrary* non-decreasing function. Let $x_e \in \{0,1\}$ be a variable indicating whether resource e is selected. In contrast to packing problems, the set of resources is known in advance and now the covering constraints $\sum_e a_{i,e} x_e \geq 1$ are revealed one-by-one. At the arrival of a constraint, an algorithm observes the predictions $x_e^{\texttt{pred}} \in \{0,1\}$ (provided by the oracle) on some variables e and needs to increasingly update the solution $\mathbf{x} = (x_e)_{e\in\mathcal{E}}$ in order to satisfy the new constraint as well as the previous ones in the sense of online algorithms (i.e., an irrevocable decision means that one cannot decrease the value of the decision variables). The goal is to design an algorithm that minimizes $f(\mathbf{x})$ subject to the online covering constraints. Again, we seek a fractional solution that is consistent to the prediction and robust to the optimal offline solution.

Approach and Contribution. Having inspired by the approach for the non-linear packing problems, we consider the notion introduced in [36].

Definition 2 *Let $\mathcal{E}$ be a set of n resources. A differentiable function $F : [0,1]^n \to \mathbb{R}^+$ is (λ,μ)-*min-locally-smooth *if for any set $S \subseteq \mathcal{E}$, and for all vectors $\mathbf{x}^e \in [0,1]^n$ where $e \in \mathcal{E}$, the following inequality holds:*

$$\sum_{e\in S} \nabla_e F(\mathbf{x}^e) \leq \lambda F\big(\mathbf{1}_S\big) + \mu F\big(\mathbf{x}\big)$$

where $\mathbf{x} := \bigvee_{e\in S} \mathbf{x}^e$, i.e., $x_{e'} = \max_e\{x^e_{e'}\}$ for every coordinate e' and $\nabla_e F(\mathbf{x})$ denotes $\partial F(\mathbf{x})/\partial x_e$.

This definition is slightly more complex than the version for maximization problem since we aim to study general functions F with non-monotone gradient. In particular, if $\nabla_e F(\mathbf{x})$ is non-decreasing on every coordinate e, we only need a simpler version. We say that a differentiable function $F : [0,1]^n \to \mathbb{R}^+$ with monotone gradient is (λ,μ)-*min-locally-smooth* if for any set $S \subseteq \mathcal{E}$, and for any vector $\mathbf{x} \in [0,1]^n$, the following inequality holds.

$$\sum_{e\in S} \nabla_e F(\mathbf{x}) \leq \lambda F\big(\mathbf{1}_S\big) + \mu F\big(\mathbf{x}\big)$$

Building upon the salient ideas of our approach for packing constraints and the recent method for online algorithms with predictions for linear covering problem [7], we show the following result.

Theorem 2 *Let F be the multilinear extension of the objective function f and d be the maximal row sparsity of the constraint matrix, i.e., $d = \max_i |\{a_{ie} : a_{ie} > 0\}|$. Assume that F is $\big(\lambda, \frac{\mu}{\ln(1+2d^2/\eta)}\big)$-min-locally-smooth for some parameters $\lambda > 0$, $\mu < 1$ and $0 < \eta \leq 1$. Then, for every $0 < \eta \leq 1$, there exists a $O\big(\frac{1}{1-\eta}\big)$-consistent and $O\big(\frac{\lambda}{1-\mu} \cdot \ln \frac{d}{\eta}\big)$-robust algorithm for the fractional covering problem.*

Applications. We show the applicability of our framework to the following problems.

Load Balancing. Load balancing is a classic problem in discrete optimization with wide applications (for example, resource management in data centers). In high level, one needs to assign online jobs to m machines in order to balance the machine loads, i.e., minimize the maximum load. Using our framework, we provide an $O(\frac{1}{1-\eta})$-consistent and and $O\big((\log m)\log^2 \frac{m}{\eta}\big)$-robust fractional solution for the problem.

Energy Minimization in Scheduling. Energy-efficient algorithms have been extensively studied in scheduling [1]. Informally, one needs to design an assignment policy of jobs to m machines and an execution policy of jobs on every machine in order to minimize the total energy consumption. The energy objective function is typically a polynomial of degree $k > 1$. As a consequence of our framework, we derive an $O(\frac{1}{1-\eta})$-consistent and $O\big(k^k \log^k \frac{m}{\eta}\big)$-robust fractional solution for the energy minimization problem.

Submodular Minimization. Submodular minimization has been widely considered in optimization, machine learning [21, 5, 4]. Using our framework, we present a $O(\frac{1}{1-\eta})$-consistent and $O\big(\frac{\log(d/\eta)}{1-\kappa_f}\big)$-robust algorithm for minimizing a submodular function under covering constraints where κ_f is the curvature (defined in Section 3.2.3) of the submodular function.

1.4 Related work

The primal-dual methods has been shown to be a powerful tool in online computation. Buchbinder and Naor [11] gave a primal-dual framework for linear programs with packing/covering constraints. Their method unifies several previous potential-function-based analyses and give a principled approach to design and analyze algorithms for problems with linear relaxations. Azar et al. [3] gave a framework for covering/packing problems with convex/concave objectives whose gradients are monotone. Subsequently, Thang [36] presented algorithms beyond the convexity of the functions and established the competitive ratio as a function of the smoothness parameters of the objective function. The smoothness notion introduced in [36] has rooted in smooth games defined by [32] in the context of algorithmic game theory.

The domain of algorithms with predictions [30], or learning augmented algorithms, has been recently emerged and rapidly grown at the intersection of (discrete) algorithm design and machine learning. The idea is to incorporate learning predictions, together with ML techniques, into algorithm design in order to achieve algorithms with performance guarantees beyond the worst-case analysis and provide specifically adapted solutions to different problems. Interesting results have been shown over a large spectre of problems such as scheduling [26, 29], caching (paging) [27, 31, 2], ski rental [19, 25], counting sketches [20], bloom filters [23, 28], etc. Recently, Bamas et al. [7] have proposed a primal-dual approach to design online algorithms with predictions for linear problems with covering constraints. In this paper, by combining their ideas and the ones in

[11, 3, 36], we present primal-dual frameworks for more general problems with non-linear objectives and packing/covering constraints (and so answer some questions raised in [7]).

2 Primal-Dual Framework for Packing Constraints

Packing Problem. Let $\mathcal{E}$ be a set of n resources and let $f : \{0,1\}^n \to \mathbb{R}^+$ be an *arbitrary* non-decreasing function. Let $x_e \in \{0,1\}$ be a variable indicating whether resource e is selected. The set of packing constraints $\sum_e b_{i,e} x_e \leq 1 \ \forall i$ (including $x_e \leq 1 \ \forall e$) are given in advance and resources e are revealed online one-by-one. At the arrival of resource e, one receives a prediction $x_e^{\texttt{pred}} \in \{0,1\}$ and needs to make a decision on x_e while maintaining $\mathbf{x} = (x_e)_{e \in \mathcal{E}}$ feasible to the set of constraints. The objective of the problem is to maximize $f(\mathbf{x})$. In this problem, we seek a fractional solution that is consistent to the prediction and robust to the optimal offline solution.

2.1 Algorithm for Fractional Packing

Recall that a differentiable function $F : [0,1]^n \to \mathbb{R}^+$ is (λ, μ)-max-locally-smooth if for any set $S \subset \mathcal{E}$, and for every vector $\mathbf{x} \in [0,1]^n$, the following inequality holds:

$$\sum_{e \in S} \nabla_e F(\mathbf{x}) \geq \lambda F(\mathbf{1}_S) - \mu F(\mathbf{x})$$

Formulation. We say that $S \subset \mathcal{E}$ is a *configuration* if $\mathbf{1}_S$ corresponds to a feasible solution. Let x_e be a variable indicating whether the resource e is used. For configuration S, let z_S be a variable such that $z_S = 1$ if and only if $x_e = 1$ for every resource $e \in S$, and $x_e = 0$ for $e \notin S$. In other words, $z_S = 1$ iff $\mathbf{1}_S$ is the output solution of the problem. We consider the following formulation and the dual of its relaxation.

$$\begin{aligned}
&\max \sum_S f(\mathbf{1}_S) z_S && && \min \sum_i \alpha_i + \gamma && \\
&\sum_e b_{i,e} \cdot x_e \leq 1 && \forall i && \sum_i b_{i,e} \cdot \alpha_i \geq \beta_e && \forall e \\
&\sum_{S : e \in S} z_S = x_e && \forall e && \gamma + \sum_{e \in S} \beta_e \geq f(\mathbf{1}_S) && \forall S \\
&\sum_S z_S = 1 && && \alpha_i \geq 0 && \forall i \\
&x_e, z_S \in \{0,1\} && \forall e, S && &&
\end{aligned}$$

In the primal, the first constraints represent the given polytope. Note that the box constraints $x_e \leq 1 \ \forall e$ is included among these constraints. The second constraint ensures that if a resource e is chosen then the selected solution must contain e. The third constraint guarantees that one solution (configuration) must be selected.

Algorithm. Assume that function $F(\cdot)$ is (λ, μ)-max-locally smooth. Let d be the maximal number of positive entries in a row, i.e., $d = \max_i |\{b_{ie} : b_{ie} > 0\}|$. Define $\rho = \max_i \max_{e,e' : b_{ie'} > 0} b_{ie}/b_{ie'}$. In the algorithm, we maintain a vector $\mathbf{y}$ which would be a solution returned by a primal-dual algorithm if there is no prediction (or no trust to the prediction) and an output solution $\mathbf{x}$ which is

defined as a function of $\mathbf{y}$ and the prediction $\mathbf{x}^{\mathtt{pred}}$. At the arrival of a new resource e, we consider packing constraints with new coefficients $\overline{b}_{i,e}$ instead of $b_{i,e}$ to build the solution $\mathbf{y}$. The value of $\overline{b}_{i,e}$ depends on coefficient $b_{i,e}$ and the prediction. Specifically, $\overline{b}_{i,e} = (1+\eta)b_{i,e}$ if $x_e^{\mathtt{pred}} = 1$ and the predictive solution is still feasible; $\overline{b}_{i,e} = \frac{1+\eta}{\eta} b_{i,e}$ otherwise. Intuitively, if we do not trust the prediction (i.e., η is close to 1) then $\overline{b}_{i,e} = 2b_{i,e}$ and so x_e, y_e would get a value proportional to the one returned by a primal-dual algorithm in the classic setting. Inversely, if we trust the prediction (i.e., η is close to 0) then $\overline{b}_{i,e} \approx b_{i,e}$ for $x_e^{\mathtt{pred}} = 1$ and $\overline{b}_{i,e} \approx b_{i,e}/\eta$ for $x_e^{\mathtt{pred}} = 0$. Therefore, the new constraint $\sum_{e'} \overline{b}_{i,e'} y_{e'} \leq B_i$ will likely prevent y_e for e such that $x_e^{\mathtt{pred}} = 0$ from getting a large value. Hence, y_e for e such that $x_e^{\mathtt{pred}} = 1$ would get a large value. In the end of the algorithm, we set the output solution x_e roughly by rescaling a factor $\frac{1}{1+\eta}$ to y_e or $x_e^{\mathtt{pred}}$ (depending on cases) in order to maintain the feasibility and the consistency to the prediction.

The construction of $\mathbf{y}$ follows the scheme in [36]. Define $\overline{\rho} = \max_i \max_{e,e':\overline{b}_{ie'}>0} \overline{b}_{i,e}/\overline{b}_{ie'}$. So in particular, $\overline{\rho} \leq \rho/\eta$. Recall that $\nabla_e F(\mathbf{y}) = \partial F(\mathbf{y})/\partial y_e$. Conventionally, when a resource e is not released, $\nabla_e F(\mathbf{y}) = 0$. While $\nabla_e F(\mathbf{y}) > 0$ (i.e., one can still improve the cost by increasing y_e) and $\sum_i \overline{b}_{i,e}\alpha_e \leq \frac{1}{\lambda}\nabla_e F(\mathbf{y})$, the primal variable y_e and dual variables α_i's are increased by appropriate rates. We will argue in the analysis that the primal and dual solutions returned by the algorithm are feasible. Recall that by definition of the multilinear extension, $\nabla_e F(\mathbf{y}) = \mathbb{E}\big[f\big(\mathbf{1}_{R\cup\{e\}}\big) - f\big(\mathbf{1}_R\big)\big]$ where R is a random subset of resources $N \setminus \{e\}$ such that e' is included with probability $y_{e'}$. Therefore, during the iteration of the while loop with respect to resource e, only y_e is modified and $y_{e'}$ remain fixed for $e' \neq e$, so $\nabla_e F(\mathbf{y})$ is constant during the iteration.

Algorithm 1 Algorithm for Packing Constraints.

All primal and dual variables are initially set to 0.
At every step, always maintain $z_S = \prod_{e\in S} x_e \prod_{e\notin S}(1-x_e)$.
for each arrival of a new resource e **do**
  if $x_e^{\mathtt{pred}} = 1$ **and** the predictive solution is still feasible **then** set $\overline{b}_{i,e} = b_{i,e}$
  else set $\overline{b}_{i,e} = b_{i,e}/\eta$.
  while $\sum_i \overline{b}_{i,e}\alpha_i \leq \frac{1}{\lambda}\nabla_e F(\mathbf{y})$ **and** $\nabla_e F(\mathbf{y}) > 0$ **do**
    Increase τ at rate 1 and increase y_e at rate $\frac{1}{\nabla_e F(\mathbf{y})\cdot\ln(1+d\overline{\rho})}$.
    for i such that $\overline{b}_{i,e} > 0$ **do**
      Increase α_i according to the following function $\frac{\partial\alpha_i}{\partial\tau} \leftarrow \frac{\overline{b}_{i,e}\cdot\alpha_i}{\nabla_e F(\mathbf{y})} + \frac{1}{d\lambda}$
    end for
  end while
  if $x_e^{\mathtt{pred}} = 1$ and the predictive solution is still feasible **then** set $x_e \leftarrow \frac{1}{1+\eta}$
  else set $x_e \leftarrow \frac{1}{1+\eta} y_e$.
end for

Dual variables. Variables α_i's are constructed in the algorithm. Define $\gamma = \frac{\mu}{\lambda}F(\mathbf{y})$ and $\beta_e = \frac{1}{\lambda}\cdot\nabla_e F(\mathbf{y})$.

The following lemma provides a lower bound on α-variables.

Lemma 1 *At any moment during the iteration related to resource e, for every constraint i it always holds that*

$$\alpha_i \geq \frac{\nabla_e F(\mathbf{y})}{\max_{e'} \overline{b}_{i,e'} \cdot d\lambda} \left[\exp\left(\ln\big(1+d\overline{\rho}\big)\cdot\sum_{e'} \overline{b}_{i,e'}\cdot y_{e'}\right) - 1\right].$$

Proof Consider any moment τ during the loop corresponding to resource e. Recall that F is a linear extension, so by its definition, F is linear w.r.t y_e; hence $\nabla_e F(\mathbf{y})$ is independent of y_e. Moreover, during the loop corresponding to resource e, only y_e is modified whereas other $y_{e'}$'s for $e' \neq e$ remain unchanged. As $\nabla_e F(\mathbf{y})$ is independent of y_e and other $y_{e'}$'s remain unchanged, $\partial \nabla_e F(\mathbf{y})/\partial\tau = 0$. Therefore, the rate at time τ of the the right-hand-side of the lemma inequality is:

$$\begin{aligned}
&\frac{\nabla_e F(\mathbf{y})}{\max_{e'} \overline{b}_{i,e'} \cdot d\lambda} \cdot \ln\big(1 + d\overline{\rho}\big) \cdot \overline{b}_{i,e} \cdot \frac{\partial y_e}{\partial \tau} \cdot \exp\left(\ln\big(1 + d\overline{\rho}\big) \cdot \sum_{e'} \overline{b}_{i,e'} \cdot y_{e'}\right) \\
&\leq \frac{\nabla_e F(\mathbf{y})}{\max_{e'} \overline{b}_{i,e'} \cdot d\lambda} \cdot \ln\big(1 + d\overline{\rho}\big) \cdot \overline{b}_{i,e} \cdot \frac{1}{\nabla_e F(\mathbf{y}) \cdot \ln(1 + d\overline{\rho})} \cdot \left(\frac{\max_{e'} \overline{b}_{i,e'} \cdot d\lambda \cdot \alpha_i}{\nabla_e F(\mathbf{y})} + 1\right) \\
&\leq \frac{\overline{b}_{i,e} \cdot \alpha_i}{\nabla_e F(\mathbf{y})} + \frac{1}{d\lambda} = \frac{\partial \alpha_i}{\partial \tau},
\end{aligned}$$

where in the first inequality we use the induction hypothesis and the increasing rate of y_e. So at any time during the iteration related to e, the increasing rate of the left-hand side is always larger than that of the right-hand side. Hence, the lemma follows. □

Lemma 2 *The primal variables constructed by the algorithm are feasible.*

Proof First observe that if during the execution of the algorithm in the iteration related to some resource e, $\sum_{e'} \overline{b}_{i,e'} y_{e'} > 1$ for some constraint i then by Lemma 1,

$$\alpha_i > \frac{\nabla_e F(\mathbf{y})}{\max_{e'} \overline{b}_{i,e'} \cdot d\lambda} \left[\exp\left(\ln\big(1 + d\overline{\rho}\big)\right) - 1\right] = \frac{\overline{\rho} \cdot \nabla_e F(\mathbf{y})}{\lambda \max_{e'} \overline{b}_{i,e'}} \geq \frac{\nabla_e F(\mathbf{y})}{\lambda \overline{b}_{i,e}}$$

Therefore, $\sum_i \overline{b}_{i,e} \alpha_i > \frac{1}{\lambda} \nabla_e F(\mathbf{y})$ and hence the algorithm would have stopped increasing y_e at some earlier point. Therefore, every constraint $\sum_{e'} \overline{b}_{i,e'} y_{e'} \leq 1$ is alway maintained during the execution of the algorithm.

We are now proving the primal feasibility (even when the prediction oracle provides an infeasible solution). Let e^* be the the first resource where one realizes that the prediction oracle provides an infeasible solution (i.e., $x^{\texttt{pred}}_{e^*} = 1$ and $\sum_e b_{i,e} x^{\texttt{pred}}_e > 1$ for some constraint i). (In case the prediction always provides a feasible solution then e^* does not exists.) Let S_1 be the set of all resources e such that $x^{\texttt{pred}}_e = 1$ and e is released before e^* (so the prediction restricted on S_1 constitutes a feasible solution) and $S_2 = \{e : x^{\texttt{pred}}_e = 1\} \setminus S_1$. Again, if the prediction always provides a feasible solution then $S_1 = \{e : x^{\texttt{pred}}_e = 1\}$ and $S_2 = \emptyset$.

For every constraint i,

$$\begin{aligned}
\sum_e b_{i,e}x_e &= \sum_{e:x_e^{\texttt{pred}}=1} b_{i,e}x_e + \sum_{e:x_e^{\texttt{pred}}=0} b_{i,e}x_e \\
&= \sum_{e\in S_1} b_{i,e}x_e + \sum_{e\in S_2} b_{i,e}x_e + \sum_{e:x_e^{\texttt{pred}}=0} b_{i,e}x_e \\
&\le \frac{1}{1+\eta}\cdot 1 + \eta\cdot\sum_{e\in S_2}\bar{b}_{i,e}x_e + \eta\cdot\sum_{e:x_e^{\texttt{pred}}=0}\bar{b}_{i,e}x_e \\
&= \frac{1}{1+\eta} + \frac{\eta}{1+\eta}\sum_{e\in S_2}\bar{b}_{i,e}y_e + \frac{\eta}{1+\eta}\sum_{e:x_e^{\texttt{pred}}=0}\bar{b}_{i,e}y_e \\
&\le \frac{1}{1+\eta} + \frac{\eta}{1+\eta}\sum_{e}\bar{b}_{i,e}y_e \\
&\le \frac{1}{1+\eta} + \frac{\eta}{1+\eta}\cdot 1 = 1.
\end{aligned}$$

The first inequality is due to: (1) the fact that the prediction restricted on S_1 is a feasible solution, i.e., $\sum_{e\in S_1} b_{i,e}x_e^{\texttt{pred}} \le 1$; and (2) $x_e = \frac{1}{1+\eta} = \frac{1}{1+\eta}x_e^{\texttt{pred}}$ for $e \in S_1$; and (3) the definitions of $\bar{b}_{i,e}$ in Algorithm 1. The third equality follows the algorithm: $x_e = \frac{1}{1+\eta}\cdot y_e$ for $e \notin S_1$. The last inequality holds since $\sum_e \bar{b}_{i,e}y_e \le 1$ by the observation in the beginning of the lemma. Hence, $\sum_e b_{i,e}x_e \le 1$ for every constraint i.

Besides, by definition of $z_S = \prod_{e\in S} x_e \prod_{e\notin S}(1-x_e)$ where $0 \le x_e \le 1$ for all e, the identity $\sum_S z_S = 1$ always holds. In fact, if one chooses an element e with probability x_e then z_S is the probability that the set of selected elements is S. So the total probability $\sum_S z_S$ must be 1. Similarly, $\sum_{S:e\in S} z_S = x_e \sum_{S'\subset E\setminus\{e\}} \prod_{e'\in S'} x_{e'} \prod_{e'\notin S'}(1-x_{e'}) = x_e$ since $\sum_{S'\subset E\setminus\{e\}} \prod_{e'\in S'} x_{e'} \prod_{e'\notin S'}(1-x_{e'}) = 1$ (by the same argument).

Therefore, the solution $(\mathbf{x}, \mathbf{z})$ is primal feasible. □

Lemma 3 *The dual variables defined as above are feasible.*

Proof The first dual constraint $\sum_i b_{i,e}\alpha_i \ge \beta_e$ is satisfied by the while loop condition of the algorithm and the definition of β_e. The second dual constraint $\gamma + \sum_{e\in S}\beta_e \ge f(\mathbf{1}_S)$ reads

$$\frac{1}{\lambda}\sum_{e\in S}\nabla_e F(\mathbf{y}) + \frac{\mu}{\lambda}F(\mathbf{y}) \ge F(\mathbf{1}_S),$$

which is, by arranging terms, exactly the (λ,μ)-max-local smoothness of F. (Recall that $F(\mathbf{1}_S) = f(\mathbf{1}_S)$.) Hence, the lemma follows. □

Theorem 1 *Let F be the multilinear extension of the objective function f. Denote the row sparsity $d := \max_i |\{b_{ie} : b_{ie} > 0\}|$ and $\rho := \max_i \max_{e,e':b_{ie'}>0} b_{ie}/b_{ie'}$. Assume that F is (λ,μ)-max-locally-smooth for some parameters $\lambda > 0$ and μ. Then, for every $0 < \eta \le 1$, there exists a $r(\eta)$-consistent and $r(\eta)\cdot\left(\frac{\lambda}{2\ln(1+d\rho/\eta)+\mu}\right)$-robust algorithm for the fractional packing problem where $r(\eta) = \min_{\mathbf{0}\le\mathbf{u}\le\mathbf{1}} F(\frac{\mathbf{u}}{1+\eta})/F(\mathbf{u})$.*

Proof

Robustness. First, we bound the increases of $F(\mathbf{y})$ and the dual objective at any time τ in the execution of Algorithm 1. The derivative of $F(\mathbf{y})$ with respect to τ is:

$$\nabla_e F(\mathbf{y}) \cdot \frac{\partial y_e}{\partial \tau} = \nabla_e F(\mathbf{y}) \cdot \frac{1}{\nabla_e F(\mathbf{y}) \cdot \ln(1+d\overline{\rho})} = \frac{1}{\ln(1+d\overline{\rho})}$$

Besides, the rate of the dual at time τ is:

$$\begin{aligned}\frac{\partial D}{\partial \tau} &= \sum_i \frac{\partial \alpha_i}{\partial \tau} + \frac{\partial \gamma}{\partial \tau} = \sum_{i:\overline{b}_{i,e}>0} \left(\frac{\overline{b}_{i,e} \cdot \alpha_i}{\nabla_e F(\mathbf{y})} + \frac{1}{d\lambda} \right) + \frac{\mu}{\lambda} \frac{\partial F(\mathbf{y})}{\partial \tau} \\ &= \sum_{i:\overline{b}_{i,e}>0} \frac{\overline{b}_{i,e} \cdot \alpha_i}{\nabla_e F(\mathbf{y})} + \sum_{i:\overline{b}_{i,e}>0} \frac{1}{d\lambda} + \frac{\mu}{\lambda} \cdot \frac{1}{\ln(1+d\overline{\rho})} \\ &\leq \frac{2}{\lambda} + \frac{\mu}{\lambda \cdot \ln(1+d\overline{\rho})} = \frac{2\ln(1+d\overline{\rho})+\mu}{\lambda \cdot \ln(1+d\overline{\rho})},\end{aligned}$$

where the inequality holds since during the algorithm $\sum_i \overline{b}_{i,e} \cdot \alpha_i \leq \frac{1}{\lambda} \nabla_e F(\mathbf{y})$. Hence, the ratio between $F(\mathbf{y})$ and the dual D is at least $\frac{\lambda}{2\ln(1+d\overline{\rho})+\mu}$.

Besides, $x_e = \frac{1}{1+\eta} \geq \frac{1}{1+\eta} y_e$ if $e \in S_1$ and $x_e = \frac{1}{1+\eta} y_e$ if $e \notin S_1$. Therefore, $\mathbf{x} \geq \frac{\mathbf{y}}{1+\eta}$ and so $F(\mathbf{x}) \geq F\left(\frac{\mathbf{y}}{1+\eta}\right)$ by monotonicity of F. Hence the robustness is at least

$$\frac{F(\mathbf{x})}{F(\mathbf{y})} \cdot \frac{\lambda}{2\ln(1+d\overline{\rho})+\mu} \geq \min_{\mathbf{0} \leq \mathbf{u} \leq \mathbf{1}} \frac{F(\frac{1}{1+\eta}\mathbf{u})}{F(\mathbf{u})} \cdot \frac{\lambda}{2\ln(1+d\rho/\eta)+\mu}$$

where the latter is due to $\overline{\rho} \leq \rho/\eta$.

Consistency. By our algorithm, for every resource e if $x_e^{\mathtt{pred}} = 1$ (and the prediction oracle provides a feasible solution) then $x_e = \frac{1}{1+\eta}$. Hence, the consistency of the algorithm $F(\mathbf{x})/F(\mathbf{x}^{\mathtt{pred}}) \geq F(\frac{\mathbf{x}^{\mathtt{pred}}}{1+\eta})/F(\mathbf{x}^{\mathtt{pred}}) \geq r(\eta)$. □

2.2 Applications

2.2.1 Applications to linear functions

When the objective can be expressed as a linear functions f, its multilinear extension F is $(1,1)$-max-locally-smooth. Moreover, $r(\eta) = 1/(1+\eta)$. Consequently, Algorithm 1 gives $1/(1+\eta)$-consistent and $O\big(1/\ln(1+d\rho/\eta)\big)$-robust fractional solution for online linear maximization under packing constraints.

2.2.2 Applications to online submodular maximization

Consider the online problem of maximizing a monotone submodular function subject to packing constraints. A set-function $f : 2^{\mathcal{E}} \to \mathbb{R}+$ is *submodular* if $f(S \cup e) - f(S) \geq f(T \cup e) - f(T)$ for all $S \subset T \subseteq \mathcal{E}$. Let F be the multilinear extension of a monotone submodular function f. Function F admits several useful properties: (i) if f is monotone then so is F; (ii) F is concave in positive direction, i.e., $\nabla F(\mathbf{x}) \geq \nabla F(\mathbf{y})$ for all $\mathbf{x} \leq \mathbf{y}$ ($\mathbf{x} \leq \mathbf{y}$ means $x_e \leq y_e \ \forall e$).

Lemma 4 *Let f be an arbitrary monotone submodular function. Then, the multilinear extension F of f is (1,1)-max-locally-smooth.*

Proof As F is the linear extension of a submodular function, $\nabla_e F(\mathbf{x}) = \mathbb{E}_R\big[f\big(\mathbf{1}_{R\cup\{e\}}\big) - f\big(\mathbf{1}_R\big)\big]$ where R is a random subset of resources $\mathcal{E}\setminus\{e\}$ such that e' is included with probability $x_{e'}$. For any subset $S = \{e_1, \ldots, e_\ell\}$, we have

$$\begin{aligned}
F(\mathbf{x}) + \sum_{e\in S} \nabla_e F(\mathbf{x}) &= F(\mathbf{x}) + \sum_{e\in S} \mathbb{E}_R\big[f\big(\mathbf{1}_{R\cup\{e\}}\big) - f\big(\mathbf{1}_R\big)\big] \\
&= \mathbb{E}_R\left[f(\mathbf{1}_R) + \sum_{e\in S}\big[f\big(\mathbf{1}_{R\cup\{e\}}\big) - f\big(\mathbf{1}_R\big)\big]\right] \\
&\geq \mathbb{E}_R\left[f(\mathbf{1}_R) + \sum_{i=1}^{\ell}\big[f(\mathbf{1}_{R\cup\{e_1,\ldots,e_i\}}) - f(\mathbf{1}_{R\cup\{e_1,\ldots,e_{i-1}\}})\big]\right] \\
&= \mathbb{E}_R\big[f(\mathbf{1}_{R\cup S})\big] \geq \mathbb{E}_R\big[f(\mathbf{1}_S)\big] \\
&= F\big(\mathbf{1}_S\big)
\end{aligned}$$

the first and second inequalities are due to the submodularity and monotonicity of f, respectively. The lemma follows. □

Proposition 1 *For any $0 < \eta \leq 1$, Algorithm 1 gives a $(1-\eta)$-consistent and $O\big(1/\ln(1+d\rho/\eta)\big)$-robust fractional solution to the problem of online submodular maximization under packing constraints.*

Proof We first bound $r(\eta) = \min_{\mathbf{0}\leq\mathbf{u}\leq\mathbf{1}} F(\frac{1}{1+\eta}\mathbf{u})/F(\mathbf{u})$. By the non-negativity and the concavity in positive direction of F, for any set $S \subset \mathcal{E}$,

$$F\left(\frac{\mathbf{u}}{1+\eta}\right) \geq F(\mathbf{u}) - \left\langle \nabla F\left(\frac{\mathbf{u}}{1+\eta}\right), \mathbf{u} - \frac{\mathbf{u}}{1+\eta}\right\rangle \tag{1}$$

$$F\left(\frac{\mathbf{u}}{1+\eta}\right) \geq F(\mathbf{0}) + \left\langle \nabla F\left(\frac{\mathbf{u}}{1+\eta}\right), \frac{\mathbf{u}}{1+\eta}\right\rangle \geq \left\langle \nabla F\left(\frac{\mathbf{u}}{1+\eta}\right), \frac{\mathbf{u}}{1+\eta}\right\rangle. \tag{2}$$

Therefore,

$$\begin{aligned}
\frac{F\left(\frac{\mathbf{u}}{1+\eta}\right)}{F(\mathbf{u})} &\geq 1 - \frac{\eta}{1+\eta}\cdot\frac{\left\langle \nabla F\left(\frac{\mathbf{u}}{1+\eta}\right), \mathbf{u}\right\rangle}{F(\mathbf{u})} && \text{(by (1))} \\
&\geq 1 - \frac{\eta}{1+\eta}\cdot\frac{\left\langle \nabla F\left(\frac{\mathbf{u}}{1+\eta}\right), \mathbf{u}\right\rangle}{F\left(\frac{\mathbf{u}}{1+\eta}\right)} && \text{(by monotonicity of } F) \\
&\geq 1 - \frac{\eta}{1+\eta}\cdot\frac{\left\langle \nabla F\left(\frac{\mathbf{u}}{1+\eta}\right), \mathbf{u}\right\rangle}{\left\langle \nabla F\left(\frac{\mathbf{u}}{1+\eta}\right), \frac{\mathbf{u}}{1+\eta}\right\rangle} && \text{(by (2))} \\
&= 1 - \eta
\end{aligned}$$

So, $r(\eta) \geq 1-\eta$. Therefore, by Theorem 1, the proposition follows Theorem 1 and the (1,1)-max-locally-smoothness of F (Lemma 4). □

One can derive online randomized algorithms for specific problems by rounding the fractional solutions. For example, using the online contention resolution rounding schemes [17], one can obtain randomized algorithms for several specific constraint polytopes, for example, knapsack polytopes, matching polytopes and matroid polytopes.

3 Primal-Dual Framework for Covering Constraints

Covering Problem. Let $\mathcal{E}$ be a set of n resources and let $f : \{0,1\}^n \to \mathbb{R}^+$ be an *arbitrary* non-decreasing function. Let $x_e \in \{0,1\}$ be a variable indicating whether resource e is selected. In contrast to packing problems, the set of resources is known in advance and now the covering constraints $\sum_e a_{i,e}x_e \geq 1$ are revealed one-by-one. At the arrival of a constraint, an algorithm observes the predictions $x_e^{\texttt{pred}} \in \{0,1\}$ (provided by the oracle) on some variables e and needs to increasingly update the solution $\mathbf{x} = (x_e)_{e\in\mathcal{E}}$ in order to satisfy the new constraint as well as the previous ones in the sense of online algorithms (i.e., an irrevocable decision means that one cannot decrease the value of the decision variables). The goal is to design an algorithm that minimizes $f(\mathbf{x})$ subject to the online covering constraints. Again, we seek a fractional solution that is consistent to the prediction and robust to the optimal offline solution.

3.1 Algorithm for Fractional Covering

Recall that a differentiable function $F : [0,1]^n \to \mathbb{R}^+$ is (λ,μ)-min-locally-smooth if for any set $S \subset \mathcal{E}$, and for any vectors $\mathbf{x}^e \in [0,1]^n$ where $e \in \mathcal{E}$, the following inequality holds:

$$\sum_{e\in S} \nabla_e F(\mathbf{x}^e) \leq \lambda F\big(\mathbf{1}_S\big) + \mu F\big(\mathbf{x}\big)$$

where $\mathbf{x} := \bigvee_{e\in S} \mathbf{x}^e$, meaning that $x_{e'} = \max_e\{x^e_{e'}\}$ for any coordinate e' and $\nabla_e F(\mathbf{x})$ denotes $\partial F(\mathbf{x})/\partial x_e$.

Formulation. We say that $S \subset \mathcal{E}$ is a *configuration* if $\mathbf{1}_S$ corresponds to a feasible solution. Let x_e be a variable indicating whether the resource e is used. For configuration S, let z_S be a variable such that $z_S = 1$ if and only if $x_e = 1$ for every resource $e \in S$, and $x_e = 0$ for $e \notin S$. In other words, $z_S = 1$ iff $\mathbf{1}_S$ is the selected solution to the problem. For any subset $A \subset \mathcal{E}$, define $c_{i,A} = \max\{1 - \sum_{e'\in A} a_{i,e'}; 0\}$ and $a_{i,e,A} := \min\{a_{i,e}; c_{i,A}\}$. Denote $b_{i,e,A} = \frac{a_{i,e,A}}{c_{i,A}}$ where $c_{i,A} > 0$. We consider the following formulation and the dual of its relaxation.

$$\begin{aligned}
&\min \sum_S f(\mathbf{1}_S) z_S && \\
&\sum_{e\notin A} b_{i,e,A}\cdot x_e \geq 1 && \forall i, A \subset \mathcal{E} \\
&\sum_{S: e\in S} z_S = x_e && \forall e \\
&\sum_S z_S = 1 && \\
&x_e, z_S \in \{0,1\} && \forall e, S
\end{aligned}
\qquad\qquad
\begin{aligned}
&\max \sum_{i,A} \alpha_{i,A} + \gamma && \\
&\sum_i \sum_{A: e\notin A} b_{i,e,A}\cdot \alpha_{i,A} \leq \beta_e && \forall e \\
&\gamma + \sum_{e\in S} \beta_e \leq f(\mathbf{1}_S) && \forall S \\
&\alpha_i \geq 0 && \forall i
\end{aligned}$$

In the primal, the first constraints are knapsack-constraints of the form $\sum_{e\notin A} a_{i,e,A}\cdot x_e \geq c_{i,A}$ corresponding to the given polytope. Note that it is sufficient to consider only constraints with $c_{i,A} > 0$. The second constraint ensures that if a resource e is chosen then the selected solution must contain e. The third constraint says that one solution (configuration) must be selected.

Algorithm. Assume that function $F(\cdot)$ is $\left(\lambda, \frac{\mu}{4\ln(1+2d^2/\eta)}\right)$–min-locally smooth. Let d be the maximal number of positive entries in a row, i.e., $d = \max_i |\{a_{ie} : a_{ie} > 0\}|$. Consider Algorithm 2 which follows the scheme in [36]. In the algorithm, the current dual variable α increases at a constant rate (Step 7) and the update of dual variables β's is shown in Step 9. We note an subtle point here: if $\beta_e < \frac{1}{\lambda}\nabla_e F(\mathbf{x})$ then set $\beta_e = \frac{1}{\lambda}\nabla_e F(\mathbf{x})$; otherwise if $\beta_e > \frac{1}{\lambda}\nabla_e F(\mathbf{x})$ then we do not set β_e as $\frac{1}{\lambda}\nabla_e F(\mathbf{x})$. By that, we preserve the following invariants during the execution of the algorithm: $\beta_e \geq \frac{1}{\lambda}\nabla_e F(\mathbf{x})$ and β_e is non-decreasing. (Note that if one assumes the monotonicity of $\nabla_e F(\mathbf{x})$ on every coordinate e then it is sufficient to always kept $\beta_e \leftarrow \frac{1}{\lambda}\nabla_e F(\mathbf{x})$.) The update rule (Step 10) is inspired by the multiplicative update (where the increasing rate of x_e is inversely proportional to β_e) and the update in the current work of [7]. Finally, using the same idea as in [3], we decrease some dual variables α if necessary in order to maintain the feasibility of our dual solution.

Algorithm 2 Algorithm for Covering Constraints.

1: Initially, set $A^* \leftarrow \emptyset$. Intuitively, A^* consists of all resources e such that $x_e = 1$.
2: All primal and dual variables are initially set to 0.
3: At every step, always maintain $z_S = \prod_{e\in S} x_e \prod_{e\notin S}(1 - x_e)$.
4: Upon the arrival of primal constraint $\sum_e a_{k,e}x_e \geq 1$ and the new corresponding dual variable α_k.
5: **while** $\sum_{e\notin A^*} b_{k,e,A^*}x_e < 1$ **do** `# Increase primal, dual variables`
6: Let τ be the current time in the execution of the algorithm.
7: Increase τ at rate 1 and increase α_{k,A^*} at rate $\frac{1}{\lambda\cdot\ln(1+2d^2/\eta)}$.
8: **for** $e \notin A^*$ such that $b_{k,e,A^*} > 0$ **do**
9: **if** $\beta_e < \frac{1}{\lambda}\nabla_e F(\mathbf{x})$ **then** $\beta_e \leftarrow \frac{1}{\lambda}\nabla_e F(\mathbf{x})$
10: Increase x_e at rate according to the following function

$$\frac{\partial x_e}{\partial \tau} \leftarrow \frac{b_{k,e,A^*}\cdot x_e}{\lambda\beta_e} + \frac{\eta}{\lambda\beta_e d} + \frac{(1-\eta)\cdot \mathbb{1}_{\{x_e^{\texttt{pred}}=1\}}}{\nabla_e F(\mathbf{x})\cdot|\{e' : x_{e'}^{\texttt{pred}} = 1\}|}$$

11: **end for**
12: **if** $x_e = 1$ **then** update $A^* \leftarrow A^* \cup \{e\}$.
13: **while** $\sum_{i=1}^{k}\sum_{A:e\notin A} b_{i,e,A}\cdot\alpha_{i,A} > \beta_e$ for some $e \notin A^*$ **do** `# Decrease dual variables`
14: **for** (m_e^*, A) such that $b_{m_e^*,e,A} = \max_i\{b_{i,e,A}|\forall A : e \notin A, \forall 1 \leq i \leq k : \alpha_{i,A} > 0\}$ **do**
15: Increase $\alpha_{m_e^*,A}$ continuously at rate $-\frac{b_{k,e,A^*}}{b_{m_e^*,e,A}}\cdot\frac{1}{\lambda\cdot\ln(1+2d^2/\eta)}$.
16: **end for**
17: **end while**
18: **end while**

Dual variables. Variables $\alpha_{i,A}$ and β_e have been constructed in the algorithm. Let $\mathbf{x}$ be the current solution of the algorithm. Define $\gamma = -\frac{\mu}{4\lambda\cdot\ln(1+2d^2/\eta)}F(\mathbf{x})$. Note that due to the algorithm, for each e, $\beta_e = \frac{1}{\lambda}\cdot\nabla_e F(\mathbf{x}^e)$ for some vector $\mathbf{x}^e$ which is the solution of the algorithm at some point during its execution. Moreover, $\mathbf{x} \geq \bigvee_{e\in S}\mathbf{x}^e$ (since x-variables are maintained non-decreasing) and $\beta_e \geq \frac{1}{\lambda}\cdot\nabla_e F(\mathbf{x})$.

The following lemma gives a lower bound on x-variables. Remark that in the proof, we do not assume the monotonicity of the gradient. (If the latter is true than again one can proceed in a simpler manner by defining $\beta_e = \frac{1}{\lambda}\nabla_e F(\mathbf{x})$ at every time in the algorithm.)

Lemma 5 *Let e be an arbitrary resource. At any moment during the execution of the algorithm where the k^{th} constraint has been released , it always holds that*

$$x_e \geq \frac{\eta}{\max b_{i,e,A} \cdot d} \left[\exp\left(\frac{\ln(1+2d^2/\eta)}{\beta_e} \cdot \sum_{A: e \notin A} \sum_i b_{i,e,A} \cdot \alpha_{i,A} \right) - 1 \right]$$

where $\max b_{i,e,A} := \max\{b_{i,e,A} > 0 | \forall A : e \notin A, \forall 1 \leq i \leq k : \alpha_{i,A} > 0\}$.

Proof Fix a resource e. We prove the lemma by induction. At the beginning of the instance, while no constraint has been released yet, both sides of the lemma are 0. Assume that the lemma holds until the arrival of constraint $\sum_e a_{k,e} x_e \geq 1$. Consider a moment τ during the execution of the algorithm and let A^* be the current set of resources e' such that $x_{e'} = 1$. If at time τ, $x_e = 1$ then by the algorithm, the set A^* has been updated so that $e \in A^*$. So the increasing rates of both sides in the lemma inequality are 0. In the remaining, assume that $x_e < 1$. Recall that by the algorithm, $\beta_e \geq \frac{1}{\lambda} \nabla_e F(\mathbf{x})$. We consider two cases $\beta_e > \frac{1}{\lambda} \nabla_e F(\mathbf{x})$ and $\beta_e = \frac{1}{\lambda} \nabla_e F(\mathbf{x})$.

Case 1: $\beta_e > \frac{1}{\lambda} \nabla_e F(\mathbf{x})$**.** In this case, by the algorithm, the value of β_e remains unchanged at time τ (Step 9), i.e., $\frac{\partial \beta_e}{\partial \tau} = 0$. Hence, the derivative of the right hand side of the lemma inequality according to τ is

$$\begin{aligned}
&\sum_i \frac{\partial \alpha_{i,A^*}}{\partial \tau} \cdot \frac{b_{i,e,A^*} \cdot \eta}{\max b_{i,e,A} \cdot d} \cdot \frac{\ln(1+2d^2/\eta)}{\beta_e} \cdot \exp\left(\frac{\ln(1+2d^2/\eta)}{\beta_e} \cdot \sum_{A: e \notin A} \sum_i b_{i,e,A} \alpha_{i,A} \right) \\
&\leq \frac{\partial \alpha_{k,A^*}}{\partial \tau} \cdot \frac{b_{k,e,A^*} \cdot \eta}{\max b_{i,e,A} \cdot d} \cdot \frac{\ln(1+2d^2/\eta)}{\beta_e} \left(\frac{\max b_{i,e,A} \cdot d}{\eta} x_e + 1 \right) \\
&= \frac{1}{\lambda \ln(1+2d^2/\eta)} \cdot \frac{b_{k,e,A^*} \cdot \eta}{\max b_{i,e,A} \cdot d} \cdot \frac{\ln(1+2d^2/\eta)}{\beta_e} \left(\frac{\max b_{i,e,A} \cdot d}{\eta} x_e + 1 \right) \\
&\leq \frac{b_{k,e,A^*} \cdot x_e}{\lambda \beta_e} + \frac{\eta}{\lambda \beta_e d} \leq \frac{\partial x_e}{\partial \tau}
\end{aligned}$$

In the first inequality, we use the induction hypothesis and $\frac{\partial \alpha_{k,A^*}}{\partial \tau} > 0$ and $\frac{\partial \alpha_{i,A^*}}{\partial \tau} \leq 0$ for $i \neq k$ and $\frac{\partial \beta_e}{\partial \tau} = 0$. The equality follows the increasing rate of α_{k,A^*} according to the algorithm. The last inequality is due to the increasing rate of x_e. So the rate in the left-hand side is always larger than that in the right-hand side. Hence, the lemma inequality holds.

Case 2: $\beta_e = \frac{1}{\lambda} \nabla_e F(\mathbf{x})$**.** In this case, by the algorithm, $\frac{1}{\lambda} \nabla_e F(\mathbf{x})$ is locally non-decreasing at τ (since otherwise, by Step 9, β_e is not maintained to be equal to $\frac{1}{\lambda} \nabla_e F(\mathbf{x})$). Therefore, $\frac{\partial \beta_e}{\partial \tau} \geq 0$ and so $\partial\left(\frac{1}{\beta_e}\right)/\partial \tau \leq 0$. Hence, the derivative of the right hand side of the lemma inequality according to τ is upper bounded by

$$\sum_i \frac{\partial \alpha_{i,A^*}}{\partial \tau} \cdot \frac{b_{i,e,A^*} \cdot \eta}{\max b_{i,e,A} \cdot d} \cdot \frac{\ln(1+2d^2/\eta)}{\beta_e} \cdot \exp\left(\frac{\ln(1+2d^2/\eta)}{\beta_e} \cdot \sum_{A: e \notin A} \sum_i b_{i,e,A} \alpha_{i,A} \right)$$

which is bounded by $\frac{\partial x_e}{\partial \tau}$ by the same argument as the previous case. The lemma follows. □

Lemma 6 *The primal and dual variables are feasible.*

Proof As long as a primal covering constraint is unsatisfied, the x-variables are always increased. Therefore, at the end of an iteration, the first primal constraints (the given covering constraints) are satisfied. The other primal constraints are also satisfied by the same argument as in Lemma 2.

Consider the first dual constraint. The algorithm always maintains that $\sum_i \sum_{A:e\notin A} b_{i,e,A}\alpha_{i,A} \leq \beta_e$.Whenever this inequality is violated, by the algorithm, some α-variables are decreased (Step 15) in such a way that the increasing rate of $\sum_i \sum_{A:e\notin A} b_{i,e,A}\alpha_{i,A}$ is at most 0. Hence, by the definition of β-variables, the first dual constraint holds.

Consider the second dual constraint. Let $\mathbf{x}$ be the final solution of the algorithm. By the algorithm, for each fixed resource e, $\beta_e = \frac{1}{\lambda}\nabla_e F(\mathbf{x}^e)$ for some $\mathbf{x}^e$ where $x^e_{e'} \leq x_{e'}$ for every e'. Moreover, $\mathbf{y} := \bigvee_e \mathbf{x}^e \leq \mathbf{x}$. By definitions of dual variables, the second dual constraint (after rearranging terms) reads

$$\frac{1}{\lambda}\sum_{e\in S} \nabla_e F(\mathbf{x}^e) \leq F(\mathbf{1}_S) + \frac{\mu}{4\lambda\cdot\ln(1+2d^2/\eta)}F(\mathbf{x}).$$

Besides, as F is monotone, $F(\mathbf{x}) \geq F(\mathbf{y})$. To prove the above inequality, it is sufficient to prove that

$$\frac{1}{\lambda}\sum_{e\in S} \nabla_e F(\mathbf{x}^e) \leq F(\mathbf{1}_S) + \frac{\mu}{4\lambda\cdot\ln(1+2d^2/\eta)}F(\mathbf{y}).$$

This inequality is exactly the $\big(\lambda, \frac{\mu}{4\ln(1+2d^2/\eta)}\big)$-min-local smoothness of F. Hence, the lemma follows. □

Theorem 2 *Let F be the multilinear extension of the objective function f and d be the maximal row sparsity of the constraint matrix, i.e., $d = \max_i |\{a_{ie} : a_{ie} > 0\}|$. Assume that F is $\big(\lambda, \frac{\mu}{\ln(1+2d^2/\eta)}\big)$-min-locally-smooth for some parameters $\lambda > 0$, $\mu < 1$ and $0 < \eta \leq 1$. Then, for every $0 < \eta \leq 1$, there exists a $O\big(\frac{1}{1-\eta}\big)$-consistent and $O\big(\frac{\lambda}{1-\mu}\cdot\ln\frac{d}{\eta}\big)$-robust algorithm for the fractional covering problem.*

Proof

Robustness. We prove the robustness by bounding the increases of the cost and the dual objective at any time τ in the execution of Algorithm 2. Let A^* be the current set of resources e such that $x_e = 1$. The derivative of the objective with respect to τ is:

$$\begin{aligned}\sum_e \nabla_e F(\mathbf{x})\cdot\frac{\partial x_e}{\partial\tau} &= \sum_e \nabla_e F(\mathbf{x})\cdot\left(\frac{b_{k,e,A^*}\cdot x_e}{\lambda\beta_e} + \frac{\eta}{\lambda\beta_e d} + \frac{(1-\eta)\cdot\mathbb{1}_{\{x^{\texttt{pred}}_e=1\}}}{\nabla_e F(\mathbf{x})\cdot|\{e' : x^{\texttt{pred}}_{e'}=1\}|}\right)\\ &\leq \sum_e\left(b_{k,e,A^*}\cdot x_e + \frac{\eta}{d}\right) + \sum_{e:x^{\texttt{pred}}=1}\frac{(1-\eta)}{|\{e' : x^{\texttt{pred}}_{e'}=1\}|} \leq 2. \end{aligned} \tag{3}$$

The first inequality follows $\nabla_e F(\mathbf{x}) \leq \lambda\cdot\beta_e$. The second inequality is due to the definition of d and the fact that $\sum_{e\notin A^*} b_{k,e,A^*}\cdot x_e \leq 1$ always holds during the algorithm.

For a time τ, let $U(\tau)$ be the set of resources e such that $\sum_i\sum_{A:e\notin A} b_{i,e,A}\alpha_{i,A} = \beta_e$ and $b_{k,e,A^*} > 0$. Note that $|U(\tau)| \leq d$ by definition of d. As long as $\sum_{e\notin A^*} b_{k,e,A^*}x_e < 1$, by Lemma 5, we have for every $e \in U(\tau)$,

$$\frac{1}{b_{k,e,A^*}} > x_e \geq \frac{\eta}{\max b_{i,e,A}\cdot d}\left[\exp\Big(\ln(1+2d^2/\eta)\Big) - 1\right] = \frac{2d}{\max b_{i,e,A}}.$$

Therefore, $\frac{b_{k,e,A^*}}{\max_i b_{i,e,A}} \leq \frac{1}{2d}$.

We are now bounding the increase of the dual at time τ. The derivative of the dual with respect to τ is:

$$\begin{aligned}
\frac{\partial D}{\partial \tau} &= \sum_i \sum_A \frac{\partial \alpha_{i,A}}{\partial \tau} + \frac{\partial \gamma}{\partial \tau} = \sum_i c_{i,A^*} \cdot \frac{\partial \alpha_{i,A^*}}{\partial \tau} + \frac{\partial \gamma}{\partial \tau} \\
&= \frac{1}{\lambda \cdot \ln(1+2d^2/\eta)} \left(1 - \sum_{e \in U(\tau)} \frac{b_{k,e,A^*}}{b_{m^*_e,e,A}}\right) - \frac{\mu}{4\lambda \cdot \ln(1+2d^2/\eta)} \sum_e \nabla_e F(\mathbf{x}) \cdot \frac{\partial x_e}{\partial \tau} \\
&\geq \frac{1}{\lambda \cdot \ln(1+2d^2/\eta)} \left(1 - \sum_{e \in U(\tau)} \frac{1}{2d}\right) - \frac{\mu}{2\lambda \cdot \ln(1+2d^2/\eta)} \\
&\geq \frac{1-\mu}{2\lambda \cdot \ln(1+2d^2/\eta)}.
\end{aligned}$$

The third equality holds since α_{k,A^*} is increased and other α-variables in $U(\tau)$ are decreased. The first inequality uses the fact that $\frac{b_{k,e,A^*}}{\max_i b_{i,e,A^*}} \leq \frac{1}{2d}$ and Inequality (3). The last inequality holds since $|U(\tau)| \leq d$. Hence, the robustness is at least $\frac{4\lambda}{1-\mu} \cdot \ln(1+2d^2/\eta)$.

Consistency. The consistency holds by the same argument as in [7]. Consider an arbitrary moment τ in the execution of the algorithm. Let $S_1 = S_1(\tau)$ be the set of resources e selected by the prediction, i.e., $x_e^{\texttt{pred}} = 1$ up to time τ. Let $S_2 = S_2(\tau)$ be the set of remaining resources. We compare the contributions of S_1 and S_2 to the increase of the primal objective.

$$\begin{aligned}
\sum_{e \in S_1} \nabla_e F(\mathbf{x}) \cdot \frac{\partial x_e}{\partial \tau} &= \sum_{e \in S_1} \nabla_e F(\mathbf{x}) \cdot \left(\frac{b_{k,e,A^*} \cdot x_e}{\lambda \beta_e} + \frac{\eta}{\lambda \beta_e d} + \frac{(1-\eta)}{\nabla_e F(\mathbf{x}) \cdot |\{e' : x_{e'}^{\texttt{pred}} = 1\}|}\right) \geq 1 - \eta, \\
\sum_{e \in S_1} \nabla_e F(\mathbf{x}) \cdot \frac{\partial x_e}{\partial \tau} &= \sum_{e \in S_1} \nabla_e F(\mathbf{x}) \cdot \left(\frac{b_{k,e,A^*} \cdot x_e}{\lambda \beta_e} + \frac{\eta}{\lambda \beta_e d}\right) \leq 1 + \eta.
\end{aligned}$$

Therefore, the increase of the primal objective is at most $\left(1 + \frac{1+\eta}{1-\eta}\right)$ the increase restricted to the set S_1 — the predictive solution. So we deduce that the algorithm is $O\left(\frac{1}{1-\eta}\right)$-consistent. □

3.2 Applications

In order to apply Theorem 2, one needs to determine the min-local-smoothness parameters. [36] provided these parameters for some wide classes of functions, in particular polynomials with non-negative coefficients. Specifically, let $g_\ell : \mathbb{R} \to \mathbb{R}$ for $1 \leq \ell \leq L$ be degree-k polynomials with non-negative coefficients and the cost function $f : \{0,1\}^n \to \mathbb{R}^+$ defined as $f(\mathbf{1}_S) = \sum_\ell b_\ell g_\ell\left(\sum_{e \in S} a_e\right)$ where $a_e \geq 0$ for every e and $b_\ell \geq 0$ for every $1 \leq \ell \leq L$. Then the multilinear extension F of f is $\left(O(k \ln(d/\eta))^{k-1}, \frac{k-1}{k \ln(1+2d^2/\eta)}\right)$-min-locally smooth. We will use these parameters to derive the guarantees for the following problems.

3.2.1 Load Balancing

Problem. Load balancing is a classic problem in discrete optimization with wide applications (for example, in machine load management in data centers). In the problem, we are given m unrelated machines and jobs arrive online. A job j, specifying by its processing times p_{ij} if it is assigned to

machine i, needs to be assigned to a machine. The load ℓ_i of a machine i is the total processing of jobs assigned to i. The objective is to minimize the maximum load. This problem in the classic online setting is well understood with tight competitive ratio of $\Theta(\log m)$ [9, 13].

In the online setting with predictions, each job j arrives over time and is represented by the constraint $\sum_{i=1}^{m} x_{ij} = 1$ where $x_{ij} \in \{0,1\}$ indicates whether job j is assigned to machine i. The constraint guarantees that job j is assigned to some machine. At the arrival of job j, the prediction oracle provides $x_{ij}^{\texttt{pred}}$. An online algorithm, based on the prediction, decides the assignment of the job. The objective can be written as $\min \max_{i=1}^{m} \ell_i = \min \max_{i=1}^{m} \big(\sum_j p_{ij} x_{ij}\big)$.

Applying our framework for covering constraints, we deduce the following result.

Proposition 2 *Algorithm 2 gives a $O(\frac{1}{1-\eta})$-consistent and $O\big((\log m)\log^2 \frac{m}{\eta}\big)$-robust fractional solution for the load balancing problem.*

Proof It is known that ∞-norm of a m-dim vector can be approximated by the $(\log m)$-norm, in particular for $m \geq 2$,

$$\|(\ell_1, \ell_2, \ldots, \ell_m)\|_\infty \leq \|(\ell_1, \ell_2, \ldots, \ell_m)\|_{\log m} \leq m^{1/m} \|(\ell_1, \ell_2, \ldots, \ell_m)\|_\infty \leq 2\|(\ell_1, \ell_2, \ldots, \ell_m)\|_\infty.$$

Hence, one can instead consider the objective of minimizing the $(\log m)$-norm of the load vectors while loosing a constant factor of 2. More precisely, we consider the $(\log m)$-th power of the $(\log m)$-norm as the objective.

$$\min \sum_{i=1}^{m} \Big(\sum_j p_{ij} x_{ij} \Big)^{\log m} \qquad \text{s.t.} \qquad \sum_{i=1}^{m} x_{ij} = 1 \ \forall j$$

The objective function is a polynomial of degree $\log m$. So its multilinear extension is $(O(k \ln(d/\eta))^{k-1}, \frac{k-1}{k \ln(1+2d^2/\eta)})$-min-locally smooth with $k = \log m$ and $d = m$ (the maximal number of positive coefficients in a constraint). Therefore, applying Theorem 2, the robustness (w.r.t the objective as the $(\log m)$-th power of the $(\log m)$-norm) is $O\big((\log m \log \frac{m}{\eta})^{\log m}\big)$. Getting back to the $(\log m)$-norm objective by taking the $(\log m)$-root, the robustness is $O\big((\log m)\log^2 \frac{m}{\eta}\big)$. Hence, Algorithm 2 is $O(\frac{1}{1-\eta})$-consistent and $O\big((\log m)\log^2 \frac{m}{\eta}\big)$-robust. □

3.2.2 Energy Minimization in Scheduling

Problem. Energy-efficient algorithms have received considerable attention and have been widely studied in scheduling [1]. In the problem, we are given m unrelated machines and jobs arrive online. Each job j is specified by its released date r_j, deadline d_j and processing volumes p_{ij} if job j is processed on machine i. For every job j, an algorithm needs to assign j to some machine i and decide the speed $s_{ij}(t)$ during time interval $[t, t+1)$ in order to execute completely the volume of the job. Every machine i has a non-decreasing energy power function $P_i(\cdot)$ and typically, $P_i(z) = z^{k_i}$ for some constant $k_i \geq 1$. The total energy is $\sum_i \sum_t P(\sum_j s_{ij}(t))$. The objective is to minimize the total energy consumption while completing all jobs. In the classic online setting, the problem is well understood: there exists an $O(k^k)$-competitive algorithm [36] where $k = \max_i k_i$ and this bound is tight up to a constant factor [13].

In the online setting with predictions, job j arrives over time and is represented by constraints:

$$\sum_{i=1}^{m} x_{ij} = 1, \qquad \sum_{t=r_j}^{d_j - 1} s_{ij}(t) \geq p_{ij} x_{ij}, \qquad s_{ij}(t) \geq 0 \ \forall i, t.$$

where $x_{ij} \in \{0,1\}$ indicates whether job j is assigned to machine i and $s_{ij}(t) \geq 0$ denotes the speed of machine i executing job j during time interval $[t, t+1)$. The first constraint guarantees that job j is assigned to some machine and the second one ensures that the total volume of job j will be completed (on the machine where it is assigned). At the arrival of job j, the prediction provides a solution $x_{ij}^{\texttt{pred}}$ and $s_{ij}^{\texttt{pred}}(t)$ for $r_j \leq t \leq d_j - 1$. An online algorithm, based on the prediction, decides the assignment and the execution of the job.

By our framework, we deduce the following result.

Proposition 3 *Algorithm 2 gives a $O(\frac{1}{1-\eta})$-consistent and $O\big(k^k \log^k \frac{m}{\eta}\big)$-robust fractional solution for the energy minimization problem.*

Proof The objective function $\sum_i \sum_t P(\sum_j s_{ij}(t))$ is a polynomial of degree $k = \max_i k_i$; so its multilinear extension is $(O(k \ln(m/\eta))^{k-1}, \frac{k-1}{k \ln(1+2m^2/\eta)})$-min-locally smooth (the maximal number of positive coefficients in a constraint $d = m$). Therefore, applying Theorem 2, Algorithm 2 provides a $O(\frac{1}{1-\eta})$-consistent and $O\big(k^k \ln^k \frac{m}{\eta}\big)$-robust fractional solution. □

3.2.3 Online Submodular Mimimization

Consider the problem of minimizing an online monotone submodular function subject to covering constraints. (The definition of submodular functions was given in Section 2.2.2.) In order to apply Algorithm 2, we need to determine the min-local-smoothness parameters.

An important concept in studying submodular functions is the *curvature*. Given a submodular function f, the *total curvature* κ_f [15] of f is defined as

$$\kappa_f = 1 - \min_e \frac{f(\mathbf{1}_{\mathcal{E}}) - f(\mathbf{1}_{\mathcal{E}\setminus\{e\}})}{f(\mathbf{1}_{\{e\}})}.$$

Intuitively, the total curvature mesures how far away f is from being *modular*. The concept of curvature has been used to determines both upper and lower bounds on the approximation ratios for many submodular and learning problems [15, 18, 6, 37, 22, 35]. The following lemma shows an useful property of the total curvature.

Lemma 7 *For any set S, it always holds that*

$$f(\mathbf{1}_S) \geq (1 - \kappa_f) \sum_{e \in S} f(\mathbf{1}_{\{e\}}).$$

Proof Let $S = \{e_1, \ldots, e_m\}$ be an arbitrary subset of $\mathcal{E}$. Let $S_i = \{e_1, \ldots, e_i\}$ for $1 \leq i \leq m$ and $S_0 = \emptyset$. We have

$$\begin{aligned} f(\mathbf{1}_S) &\geq f(\mathbf{1}_{\mathcal{E}}) - f(\mathbf{1}_{\mathcal{E}\setminus S}) = \sum_{i=0}^{m-1} f(\mathbf{1}_{\mathcal{E}\setminus S_i}) - f(\mathbf{1}_{\mathcal{E}\setminus S_{i+1}}) \geq \sum_{i=1}^{m} f(\mathbf{1}_{\mathcal{E}}) - f(\mathbf{1}_{\mathcal{E}\setminus\{e_i\}}) \\ &\geq (1 - \kappa_f) \sum_{i=1}^{m} f(\mathbf{1}_{e_i}) \end{aligned}$$

where the first two inequalities are due to submodularity of f and the last inequality follows by the definition of the curvature. □

Proposition 4 *Algorithm 2 gives a $O(\frac{1}{1-\eta})$-consistent and $O\big(\frac{\log(d/\eta)}{1-\kappa_f}\big)$-robust fractional solution for the submodular minimization under covering constraints.*

Proof Let F be the multilinear extension of f. It is sufficient to verify that F is $\big(\frac{1}{1-\kappa_f}, 0\big)$-min-locally smooth. Recall that, by definition of the multilinear extension, $F(\mathbf{x}) = \mathbb{E}\big[f(\mathbf{1}_T)\big]$ where T is a random set such that a resource e appears in T with probability x_e. Moreover, as F is linear in x_e, we have

$$\begin{aligned}\nabla_e F(\mathbf{x}) &= F(x_1, \ldots, x_{e-1}, 1, x_{e+1}, \ldots, x_n) - F(x_1, \ldots, x_{e-1}, 0, x_{e+1}, \ldots, x_n)\\ &= \mathbb{E}\Big[f\big(\mathbf{1}_{R\cup\{e\}}\big) - f\big(\mathbf{1}_R\big)\Big]\end{aligned}$$

where R is a random subset of resources $N \setminus \{e\}$ such that e' is included with probability $x_{e'}$. Therefore, in order to prove that F is (λ, μ)-min-locally-smooth, it is equivalent to show that, for any set $S \subset \mathcal{E}$ and for any vectors $\mathbf{x}^e \in [0,1]^n$ for $e \in \mathcal{E}$,

$$\sum_{e\in S} \mathbb{E}\Big[f\big(\mathbf{1}_{R^e\cup\{e\}}\big) - f\big(\mathbf{1}_{R^e}\big)\Big] \le \lambda f\big(\mathbf{1}_S\big) + \mu\mathbb{E}\Big[f\big(\mathbf{1}_R\big)\Big]$$

where R^e is a random subset of resources $N \setminus \{e\}$ such that e' is included with probability $x^e_{e'}$ and R is a random subset of resources $N \setminus \{e\}$ such that e' is included with probability $\max_{e\in S} x^e_{e'}$.

Indeed, the $\big(\frac{1}{1-\kappa_f}, 0\big)$-min-local smoothness of F holds due to the submodularity and Lemma 7: for any subsets R^e, we have

$$\sum_{e\in S}\big[f\big(\mathbf{1}_{R^e\cup\{e\}}\big) - f\big(\mathbf{1}_{R^e}\big)\big] \le \sum_{e\in S}\big[f\big(\mathbf{1}_{\{e\}}\big)\big] \le \frac{1}{1-\kappa_f}\cdot f(\mathbf{1}_S)$$

Therefore, applying Theorem 2, the proposition follows. □

4 Conclusion

In the paper, we have presented primal-dual frameworks to design algorithms with predictions for non-linear problems with packing/covering constraints. Through applications, we show the potential of our approach and provide useful ideas/guarantees in incorporating predictions into solutions to problems with high impact such as load balancing, energy minimization, submodular optmization. An interesting direction is to prove lower bounds for non-linear packing/covering problems in terms of smoothness and confidence parameters in general and for specific problems (load balancing, energy minimization, etc) in particular.

References

[1] Susanne Albers. Energy-efficient algorithms. *Commun. ACM*, 53(5):86–96, 2010.

[2] Antonios Antoniadis, Christian Coester, Marek Elias, Adam Polak, and Bertrand Simon. Online metric algorithms with untrusted predictions. In *International Conference on Machine Learning*, pages 345–355, 2020.

[3] Yossi Azar, Niv Buchbinder, TH Hubert Chan, Shahar Chen, Ilan Reuven Cohen, Anupam Gupta, Zhiyi Huang, Ning Kang, Viswanath Nagarajan, Joseph Seffi Naor, and Debmalya Panigrahi. Online algorithms for covering and packing problems with convex objectives. In *Proc. Symposium on Foundations of Computer Science (FOCS)*, 2016.

[4] Francis Bach. Submodular functions: from discrete to continuous domains. *Mathematical Programming*, pages 1–41, 2016.

[5] Francis Bach et al. Learning with submodular functions: A convex optimization perspective. *Foundations and Trends® in Machine Learning*, 6(2-3):145–373, 2013.

[6] Maria-Florina Balcan and Nicholas J. A. Harvey. Learning submodular functions. In *Proc. Machine Learning and Knowledge Discovery in Databases*, pages 846–849, 2012.

[7] Etienne Bamas, Andreas Maggiori, and Ola Svensson. The primal-dual method for learning augmented algorithms. In *Proc. 34th Conference on Neural Information Processing Systems*, 2020.

[8] Andrew An Bian, Kfir Levy, Andreas Krause, and Joachim M. Buhmann. Continuous DR-submodular maximization: Structure and algorithms. In *Neural Information Processing Systems (NIPS)*, 2017.

[9] Allan Borodin and Ran El-Yaniv. *Online computation and competitive analysis*. Cambridge University Press, 2005.

[10] Niv Buchbinder and Joseph Naor. Online primal-dual algorithms for covering and packing. *Mathematics of Operations Research*, 34(2):270–286, 2009.

[11] Niv Buchbinder and Joseph Naor. The design of competitive online algorithms via a primal-dual approach. *Foundations and Trends in Theoretical Computer Science*, 3(2-3):93–263, 2009.

[12] Niv Buchbinder, Moran Feldman, Joseph Seffi, and Roy Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. *SIAM Computing*, 44(5):1384–1402, 2015.

[13] Ioannis Caragiannis. Better bounds for online load balancing on unrelated machines. In *Proc. 19th ACM-SIAM Symposium on Discrete Algorithms*, pages 972–981, 2008.

[14] Chandra Chekuri, Jan Vondr'ak, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. *SIAM Journal on Computing*, 43(6):1831–1879, 2014.

[15] Michele Conforti and Gérard Cornuéjols. Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the rado-edmonds theorem. *Discrete applied mathematics*, 7(3):251–274, 1984.

[16] Moran Feldman, Joseph Naor, and Roy Schwartz. A unified continuous greedy algorithm for submodular maximization. In *Symposium on Foundations of Computer Science (FOCS)*, pages 570–579, 2011.

[17] Moran Feldman, Ola Svensson, and Rico Zenklusen. Online contention resolution schemes. In *Proc. 27th Symposium on Discrete Algorithms*, pages 1014–1033, 2016.

[18] Michel X Goemans, Nicholas JA Harvey, Satoru Iwata, and Vahab Mirrokni. Approximating submodular functions everywhere. In *Proc. 20th Symposium on Discrete algorithms*, pages 535–544, 2009.

[19] Sreenivas Gollapudi and Debmalya Panigrahi. Online algorithms for rent-or-buy with expert advice. In *International Conference on Machine Learning*, pages 2319–2327, 2019.

[20] Chen-Yu Hsu, Piotr Indyk, Dina Katabi, and Ali Vakilian. Learning-based frequency estimation algorithms. In *Proc. Conference on Learning Representations*, 2019.

[21] Satoru Iwata, Lisa Fleischer, and Satoru Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *Journal of the ACM (JACM)*, 48(4):761–777, 2001.

[22] Rishabh K Iyer, Stefanie Jegelka, and Jeff A Bilmes. Curvature and optimal algorithms for learning and minimizing submodular functions. In *Advances in Neural Information Processing Systems*, pages 2742–2750, 2013.

[23] Tim Kraska, Alex Beutel, Ed H Chi, Jeffrey Dean, and Neoklis Polyzotis. The case for learned index structures. In *Proc. Conference on Management of Data*, pages 489–504, 2018.

[24] Andreas Krause and Daniel Golovin. Submodular function maximization. In *Tractability: Practical Approaches to Hard Problems*, pages 71–104. Cambridge University Press, 2014.

[25] Ravi Kumar, Manish Purohit, and Zoya Svitkina. Improving online algorithms via ml predictions. In *Proc. 32nd Conference on Neural Information Processing Systems*, pages 9684–9693, 2018.

[26] Silvio Lattanzi, Thomas Lavastida, Benjamin Moseley, and Sergei Vassilvitskii. Online scheduling via learned weights. In *Proc. Symposium on Discrete Algorithms*, pages 1859–1877, 2020.

[27] Thodoris Lykouris and Sergei Vassilvtiskii. Competitive caching with machine learned advice. In *International Conference on Machine Learning*, pages 3296–3305, 2018.

[28] Michael Mitzenmacher. A model for learned bloom filters, and optimizing by sandwiching. In *Proc. Conference on Neural Information Processing Systems*, pages 464–473, 2018.

[29] Michael Mitzenmacher. Scheduling with predictions and the price of misprediction. In *Proc. 11th Innovations in Theoretical Computer Science Conference*, 2020.

[30] Michael Mitzenmacher and Sergei Vassilvitskii. *Beyond the Worst-Case Analysis of Algorithms*, chapter Algorithms with Predictions. Cambridge University Press, 2020.

[31] Dhruv Rohatgi. Near-optimal bounds for online caching with machine learned advice. In *Proc. Symposium on Discrete Algorithms*, pages 1834–1845, 2020.

[32] Tim Roughgarden. Intrinsic robustness of the price of anarchy. *Journal of the ACM*, 62(5): 32, 2015.

[33] Tim Roughgarden. Beyond worst-case analysis. *Communications of the ACM*, 62(3):88–96, 2019.

[34] Tim Roughgarden. *Beyond the Worst-Case Analysis of Algorithms*. Cambridge University Press, 2020.

[35] Maxim Sviridenko, Jan Vondrák, and Justin Ward. Optimal approximation for submodular and supermodular optimization with bounded curvature. *Mathematics of Operations Research*, 2017.

[36] Nguyen Kim Thang. Online primal-dual algorithms with configuration linear programs. In *Proc. 31st International Symposium on Algorithms and Computation*, 2020.

[37] Jan Vondrák. Submodularity and curvature: The optimal algorithm. *RIMS Kokyuroku Bessatsu*, 2010.

[38] David P Williamson and David B Shmoys. *The design of approximation algorithms*. Cambridge University Press, 2011.