Energy-Efficient Scheduling with Predictions

Eric Balkanski
Columbia University
eb3224@columbia.edu

Noemie Perivier
Columbia University
np2708@columbia.edu

Clifford Stein
Columbia University
cliff@ieor.columbia.edu

Hao-Ting Wei
Columbia University
hw2738@columbia.edu

Abstract

An important goal of modern scheduling systems is to efficiently manage power usage. In energy-efficient scheduling, the operating system controls the speed at which a machine is processing jobs with the dual objective of minimizing energy consumption and optimizing the quality of service cost of the resulting schedule. Since machine-learned predictions about future requests can often be learned from historical data, a recent line of work on learning-augmented algorithms aims to achieve improved performance guarantees by leveraging predictions. In particular, for energy-efficient scheduling, Bamas et. al. [NeurIPS '20] and Antoniadis et. al. [SWAT '22] designed algorithms with predictions for the energy minimization with deadlines problem and achieved an improved competitive ratio when the prediction error is small while also maintaining worst-case bounds even when the prediction error is arbitrarily large.

In this paper, we consider a general setting for energy-efficient scheduling and provide a flexible learning-augmented algorithmic framework that takes as input an offline and an online algorithm for the desired energy-efficient scheduling problem. We show that, when the prediction error is small, this framework gives improved competitive ratios for many different energy-efficient scheduling problems, including energy minimization with deadlines, while also maintaining a bounded competitive ratio regardless of the prediction error. Finally, we empirically demonstrate that this framework achieves an improved performance on real and synthetic datasets.

1 Introduction

Large data centers and machine learning models are important contributors to the growing impact that computing systems have on climate change. An important goal is thus to efficiently manage power usage in order to not only complete computing tasks in a timely manner but to also minimize energy consumption. In many operating systems, this tradeoff can be controlled by carefully scaling the speed at which jobs run. An extensive area of scheduling has studied such online (and offline) speed scaling problems (see, e.g., [1]). Since the speed of many processors is approximately the cube root of their power [26, 15], these works assume that the power of a processor is equal to speed to some power $\alpha \geq 1$, where α is thought of as being approximately 3 [29, 10] and the total energy consumption is power integrated over time.

Online energy-efficient scheduling algorithms have mostly been evaluated using competitive analysis, which provides robust guarantees that hold for any instance. However, since competitive analysis evaluates algorithms over worst-case instances, it can often be pessimistic. In particular, it ignores the fact that, in the context of scheduling, future computation requests to computing systems can often be

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

estimated from historical data. A recent line of work on algorithms with predictions aims to address this limitation by assuming that the algorithm designer is given access to machine-learned predictions about the input. In the context of online algorithms, where this line of work has been particularly active, the predictions are about future requests and the goal is to achieve improved competitive ratios when the predictions are accurate (consistency), while also maintaining the benefits of worst-case analysis with guarantees that hold even when the predictions are arbitrarily wrong (robustness).

In this framework with predictions, Bamas et al. [7] and Antoniadis et al. [4] recently studied the energy minimization with deadlines problem, which is a classical setting for energy-efficient scheduling (see, e.g., [29, 10]). However, there are many scenarios where the jobs do not have strict deadlines and the goal is instead to minimize the job response time. In energy plus flow time minimization problems, which are another family of energy-efficient scheduling problems that have been extensively studied in the setting without predictions, the objective is to minimize a combination of the energy consumption and the flow time of the jobs, which is the difference between their release date and completion time (see, e.g., [2, 12, 14, 3]).

In this paper, we study a general energy-efficient scheduling problem that we augment with predictions. This general problem includes both energy minimization with deadlines, which has been previously studied with predictions, and energy plus flow time minimization, which has not been previously studied with predictions, as well as many other variants and generalizations. In particular, the flow time problem with predictions introduces challenges that require novel learning-augmented scheduling algorithms (see Section 3 for additional discussion).

1.1 Our results

An instance of the General Energy-efficient Scheduling (GES) problem is described by a collection $\mathcal{J}$ of n jobs and an arbitrary quality of service cost function F. Each job $(j, r_j, p_j) \in \mathcal{J}$ consists of a release time r_j, a processing time p_j, and an identifier j (and potentially other parameters such as weights v_j or deadlines d_j). A schedule S is specified by the speeds $s_j(t)$ at which job j is processed by the machine at time t. The goal is to find a schedule of minimum cost $E(S) + F(S, \mathcal{J})$, where the energy consumption of a schedule is $E(S) = \int_{t \geq 0} (\sum_j s_j(t))^\alpha \mathrm{dt}$, for some constant $\alpha > 0$. In the general energy-efficient scheduling with predictions (GESP) problem, the algorithm is given at time $t = 0$ a collection $\hat{\mathcal{J}}$ of $\hat{n}$ predicted jobs $(j, \hat{r}_j, \hat{p}_j)$, which is a similar prediction model as in [7]. For all our results, we assume that the quality cost function F is monotone and subadditive, which are two mild conditions that are satisfied for the problems with flow times and with deadlines.

Near-optimal consistency and bounded robustness. Our first goal is to design an algorithm for the GESP problem that achieves a good tradeoff between its consistency (competitive ratio when the predictions are exactly correct) and robustness (competitive ratio when the predictions are arbitrarily wrong). Our first main result is that for any instance of the GES problem for which there exists a constant competitive algorithm and an optimal offline algorithm, there is an algorithm with predictions that is $1 + \epsilon$ consistent and $O(1)$ robust for any constant $\epsilon \in (0, 1]$ (Corollary 3.5). Since problems with the flow time and the problem with deadlines admit constant-competitive algorithms, we achieve a consistency that is arbitrarily close to optimal while also maintaining constant robustness for these problems (see Table 1 for a summary of problem-specific upper bounds). We complement this result by showing that there is a necessary trade-off between consistency and robustness for the flow time problem: for any $\lambda > 0$, there is no $1 + \lambda$-consistent algorithm that is $o(\sqrt{1 + 1/\lambda})$-robust (Appendix A.2).

The competitive ratio as a function of the prediction error. The second main result is that our algorithm achieves a competitive ratio that smoothly interpolates from the $1 + \epsilon$ consistency to the constant robustness as a function of the prediction error (Theorem 3.4). To define the prediction error, we denote by $\mathcal{J}^+ = \mathcal{J} \cap \hat{\mathcal{J}}$ the jobs that are correctly predicted. We define the prediction error $\eta = \frac{1}{\mathtt{OPT}(\hat{\mathcal{J}})} \max\{\mathtt{OPT}(\mathcal{J} \setminus \mathcal{J}^+), \mathtt{OPT}(\hat{\mathcal{J}} \setminus \mathcal{J}^+)\}$, which is the maximum between the optimal cost of scheduling the jobs $\mathcal{J} \setminus \mathcal{J}^+$ that arrived but were not predicted to arrive and the cost of the jobs $\hat{\mathcal{J}} \setminus \mathcal{J}^+$ that were predicted to arrive but did not arrive. This prediction error is upper bounded by the prediction error in [7] for the problem with uniform deadlines.

Problem	Previous results		Our results with predictions	
	without predictions	with predictions	Consistency	Robustness
Flow time	2 [3]	None	$(1+(2\lambda)^{\frac{1}{\alpha}})^{\alpha}$	$\frac{1+2^{2\alpha+1}\lambda^{\frac{1}{\alpha}}}{\lambda}$
Fractional weighted flow time	2 [14]			
Integral weighted flow time	$O((\frac{\alpha}{\log \alpha})^2)$ [12]		$(1+((\frac{\alpha}{\log \alpha})^2\lambda)^{\frac{1}{\alpha}})^{\alpha}$	$\frac{1+(\frac{\alpha}{\log \alpha})^2 2^{2\alpha}\lambda^{\frac{1}{\alpha}}}{\lambda}$
Deadlines	e^{α} [10]	[7, 4]	$1+\lambda$	$O(\frac{4^{\alpha^2}}{\lambda^{\alpha-1}})$

Table 1: The best-known competitive ratios for 4 energy-efficient scheduling problems, previous work studying these problems in the algorithms with predictions framework, and our consistency and robustness results, for any $\lambda \in (0, 1]$. Note that when λ is sufficiently small, the consistency improves over the best-known competitive ratios, while also maintaining bounded robustness. A detailed comparison with the results of [7, 4] for deadlines is provided in Section 1.2.

Extension to jobs that are approximately predicted correctly. We generalize our algorithm and the previous result to allow the correctly predicted jobs $\mathcal{J}^+$ to include jobs that are approximately predicted correctly, where the tolerable approximation is parameterized by a parameter chosen by the algorithm designer. The result for this extension requires an additional smoothness condition on the quality cost $F(S, \mathcal{J})$ of a schedule. This condition is satisfied for the flow time problem, but not by the one with deadlines.[1]

Experiments. In Section 5, we show that when the prediction error is small, our algorithm empirically outperforms on both real and synthetic datasets the online algorithm that achieves the optimal competitive ratio for energy plus flow time minimization without predictions.

1.2 Related work

Energy-efficient scheduling. Energy-efficient scheduling was initiated by Yao et al. [29], who studied the energy minimization with deadlines problem in both offline and online settings. These offline and online algorithms were later improved in [13, 10]. Over the last two decades, energy-efficient scheduling has been extended to several other objective functions. In particular, Albers and Fujiwara [2] proposed the problem of energy plus flow time minimization, which has been studied extensively (see, e.g., [3, 8, 14, 12, 18, 11, 9]).

Learning-augmented algorithms. Algorithms with predictions is a recent and extremely active area, especially in online algorithms, where it was initiated in [23, 28]. Many different scheduling problems have been studied with predictions (see, e.g., [19, 25, 17, 21, 6, 5, 16, 22]).

Learning-augmented energy-efficient scheduling. Energy-efficient scheduling with predictions has been studied by Antoniadis et al. [4] and Bamas et al. [7], who focus on the problem with deadlines, which is a special case of our setting. The prediction model in Bamas et al. [7] is the closest to ours. For the problem with deadlines, the algorithm in [7] achieves a better consistency-robustness tradeoff than our algorithm, but their algorithm and prediction model do not extend to more general energy-efficient scheduling problems such as the flow time problem. In addition, the competitive ratio as a function of the prediction error is only obtained in [7] in the case of uniform deadlines where the difference between the deadline and release date of a job is equal for all jobs (the authors mention that defining algorithms for general deadlines becomes complex and notationally heavy when aiming for bounds as a function of the prediction error). Thanks to our algorithmic framework and definition of prediction error, our bound generalizes to the non-uniform deadlines without complicating our algorithm. Antoniadis et al. [4] propose a significantly different prediction model that requires an equal number of jobs in both the prediction $\hat{\mathcal{J}}$ and true set of jobs $\mathcal{J}$. Consequently, their results are incomparable to ours and those by Bamas et al. [7].

[1] We note that Bamas et al. [7] give an alternate approach to transform an arbitrary algorithm with predictions for the problem with uniform deadlines to an algorithm that allows small deviations in the release time of the jobs. This approach can be applied to our algorithm for the problem with uniform deadlines.

Finally, we note that Lee et al. [20] also study energy scheduling with predictions, but with the different challenge of deciding if demand should be covered by local generators or the external grid.

2 Preliminaries

In the *General Energy-Efficient Scheduling (GES)* problem, an instance is described by a collection $\mathcal{J}$ of n jobs and a real-valued cost function $F(S, \mathcal{J})$ that takes as input the instance $\mathcal{J}$ and a schedule S for $\mathcal{J}$, and returns some quality evaluation of the schedule. Each job $(j, r_j, p_j) \in \mathcal{J}$ consists of a release time r_j, a processing time p_j, and an identifier j (and potentially other parameters such as weights v_j and deadlines d_j). We often abuse notation and write $j \in \mathcal{J}$ instead of $(j, r_j, p_j) \in \mathcal{J}$. For any time interval I, we let $\mathcal{J}_I = \{j \in \mathcal{J} : r_j \in I\}$ be the subset of jobs of $\mathcal{J}$ with release time in I. For intervals $I = [0, t]$ or $I = [t, \infty]$, we write $\mathcal{J}_{\leq t}$ and $\mathcal{J}_{\geq t}$.

A *feasible schedule* for a set of jobs $\mathcal{J}$ is specified by $S = \{s_j(t)\}_{t \geq 0, j \in \mathcal{J}_{\leq t}}$, where $s(t) := \sum_{j \in \mathcal{J}_{\leq t}} s_j(t)$ is the speed at which the machine runs at time t. Thus, $s_j(t)/s(t)$ is the fraction of the processing power of the machine allocated to job j at time t.[2] During a time interval I, there are $\int_I s_j(t)\text{dt}$ units of work for job j that are completed and we let S_I be the sub-schedule $\{s_j(t)\}_{t \in I, j \in \mathcal{J}_I}$. The cost function we consider is a combination of energy consumption and quality cost for the output schedule. The energy consumption incurred by a schedule is $E(S) = \int_{t \geq 0} s(t)^\alpha \text{dt}$, where $\alpha > 1$ is a problem-dependent constant, chosen so that the power at time t is $s(t)^\alpha$. To define the quality of a schedule, we introduce the *work profile* $W_j^S := \{w_j^S(t)\}_{t \geq r_j}$ of schedule S for job j, where $w_j^S(t) := p_j - \int_{r_j}^t s_j(u)\text{du}$ is the amount of work for j remaining at time t.

We consider general objective functions of the form $\text{cost}(S, \mathcal{J}) = E(S) + F(S, \mathcal{J})$ and the goal is to compute a feasible schedule of minimum cost. $F(S, \mathcal{J}) = f((W_1^S, j_1), \ldots, (W_n^S, j_n))$ is an arbitrary *quality cost* function that is a function of the work profiles and the jobs' parameters. In the energy minimization with deadlines problem, $F(S, \mathcal{J}) = \infty$ if there is a job j with completion time c_j^S such that $c_j^S > d_j$, and $F(S, \mathcal{J}) = 0$ otherwise. In the energy plus flow time minimization problem, we have $F(S, \mathcal{J}) = \sum_{j \in \mathcal{J}} c_j^S - r_j$ (see Section 3.3 for additional functions F). A function $F(S, \mathcal{J})$ is subadditive if for all sets of jobs $\mathcal{J}_1$ and $\mathcal{J}_2$, we have $F(S, \mathcal{J}_1 \cup \mathcal{J}_2) \leq F(S, \mathcal{J}_1) + F(S, \mathcal{J}_2)$. F is monotone if for all sets of jobs $\mathcal{J}$ and schedules S and S' such that $w_j^S(t) \leq w_j^{S'}(t)$ for all $j \in \mathcal{J}$ and $t \geq r_j$, we have that $F(S, \mathcal{J}) \leq F(S', \mathcal{J})$. We assume throughout the paper that F is monotone subadditive, which holds for the deadlines and flow time problems. We let $S^*(\mathcal{J})$ and $\texttt{OPT}(\mathcal{J}) := \text{cost}(S^*(\mathcal{J}), \mathcal{J})$ be an optimal offline schedule and the optimal objective value.

The general energy-efficient scheduling with predictions problem. We augment the GES problem with predictions regarding future job arrivals and call this problem the General Energy-Efficient Scheduling with Predictions problem (GESP). In this problem, the algorithm is given at time $t = 0$ a prediction $\hat{\mathcal{J}} = \{(j, \hat{r}_j, \hat{p}_j)\}$ regarding the jobs $\mathcal{J} = \{(j, r_j, p_j)\}$ that arrive online. An important feature of our prediction model is that the number of predicted jobs $|\hat{\mathcal{J}}|$ can differ from the number of true jobs $|\mathcal{J}|$.

Next, we define a measure for the prediction error which generalizes the prediction error in [7] for the problem with uniform deadlines to any GES problem. With $\mathcal{J}^+ = \mathcal{J} \cap \hat{\mathcal{J}}$ being the correctly predicted jobs, we define the prediction error as

$$\eta(\mathcal{J}, \hat{\mathcal{J}}) = \frac{1}{\texttt{OPT}(\hat{\mathcal{J}})} \max\{\texttt{OPT}(\mathcal{J} \setminus \mathcal{J}^+), \texttt{OPT}(\hat{\mathcal{J}} \setminus \mathcal{J}^+)\},$$

where $\texttt{OPT}(\mathcal{J} \setminus \mathcal{J}^+)$ is the optimal cost of scheduling the true jobs (j, r_j, p_j) such that either the prediction for j was wrong or there was no prediction for j and that $\texttt{OPT}(\hat{\mathcal{J}} \setminus \mathcal{J}^+)$ is the optimal cost of scheduling the predicted jobs $(j, \hat{r}_j, \hat{p}_j)$ such that either the prediction for j was wrong or j never arrived. The prediction error $\eta(\mathcal{J}, \hat{\mathcal{J}})$ is then the maximum of these costs, normalized by the optimal cost $\texttt{OPT}(\hat{\mathcal{J}})$ of scheduling the predicted jobs. We assume that $\hat{\mathcal{J}} \neq \emptyset$ to ensure that $\eta(\mathcal{J}, \hat{\mathcal{J}})$

[2] For ease of notation, we allow the machine to split its processing power at every time step t over multiple jobs. In practice, this is equivalent to partitioning time into arbitrarily small time periods and splitting each time period into smaller subperiods such that the machine is processing one job during each subperiod.

is well-defined. This prediction error is upper bounded by the prediction error $||w^{\text{true}} - w^{\text{pred}}||_\alpha^\alpha$ considered in [7] for the problem with uniform deadlines, which we prove in Appendix F.1. Here w^{true} and w^{pred} are the true and predicted workload at each time step t, i.e., the sum of the processing times of the jobs that arrive at t.

We note that in the above error model, a job j is in the set of correctly predicted jobs $\mathcal{J}^+$ only if all the parameters of j have been predicted exactly correctly. To overcome this limitation, we introduce in Section 4 a more general error model where some small deviations between the true and predicted parameters of a job j are allowed for the correctly predicted jobs $\mathcal{J}^+$. In Appendix F.1, we provide further discussion of this prediction model in comparison with [7, 4].

Performance metrics. The standard evaluation metrics for an online algorithm with predictions are its consistency, robustness, and competitive ratio as a function of the prediction error [24, 23]. The competitive ratio of an algorithm ALG as a function of a prediction error η is

$$c(\eta) = \max_{\mathcal{J}, \hat{\mathcal{J}}:\, \eta(\mathcal{J}, \hat{\mathcal{J}}) \leq \eta} \frac{\text{cost}_{\text{ALG}}(\mathcal{J}, \hat{\mathcal{J}})}{\texttt{OPT}(\mathcal{J})}.$$

ALG is ρ-robust if for all $\eta \geq 0$, $c(\eta) \leq \rho$ (competitive ratio when the error is arbitrarily large) and μ-consistent if $c(0) \leq \mu$ (competitive ratio when the prediction is exactly correct). The competitive ratio of ALG is called smooth if it smoothly degrades from μ to ρ as the prediction error η grows.

3 The Algorithm

In this section, we develop a simple and general algorithmic framework for GESP and analyze the resulting consistency, robustness, and competitive ratio as a function of the prediction error. We first note that the algorithm with predictions from [7] for the problem with deadlines does not easily generalize to some of the other problems that we consider, including the flow time problem (see Appendix F.3 for additional discussion). A major difference is that our algorithm consists of two distinct phases.

Predictions cannot be completely trusted. We also note that a first natural approach is to assume that the predictions are exactly correct and aim for a 1-consistent algorithm. For the problem with deadlines, Bamas et al. [7] showed that there is no 1-consistent algorithm with bounded robustness. In Appendix A.1, we show that this approach would also fail for the flow time problem because the algorithm might start by processing jobs too fast and consume too much energy when trusting the predictions. More generally, in Appendix A.2, we show that there is a necessary trade-off between consistency and robustness for the flow time problem by proving that any $1 + \lambda$-consistent algorithm must be $O(\sqrt{1 + 1/\lambda})$-robust.

3.1 Description of the algorithm

The algorithm, called TPE, takes as input an arbitrary quality of service cost function F, predictions $\hat{\mathcal{J}}$, a confidence level $\lambda \in (0, 1]$ in the predictions, an offline algorithm OFFLINEALG for F, and an online algorithm ONLINEALG for F (without predictions). We denote by $\texttt{OFF}(\mathcal{J}) := \text{cost}(\text{OFFLINEALG}(\mathcal{J}), \mathcal{J})$ the objective value achieved by OFFLINEALG over $\mathcal{J}$.

The algorithm proceeds in two phases. In the first phase (Lines 1-5), TPE ignores the predictions and runs the auxiliary online algorithm ONLINEALG over the true jobs $\mathcal{J}_{\leq t}$ that have been released by time t. More precisely, during the first phase of the algorithm, $s_j(t)$ is the speed according to the online algorithm ONLINEALG for all jobs. The first phase ends at the time t_λ when the cost of the offline schedule computed by running OFFLINEALG on jobs $\mathcal{J}_{\leq t}$ reaches the threshold value $\lambda \cdot \texttt{OFF}(\hat{\mathcal{J}})$. As we will detail in the analysis section, this first phase guarantees a bounded robustness since we ensure that the offline cost for the true jobs reaches some value before starting to trust the predictions (hence, TPE does not initially 'burn' too much energy compared to the optimal offline cost, unlike the example described in Appendix A.1).

Algorithm 1 Two-Phase Energy Efficient Scheduling (TPE)

Input: predicted and true sets of jobs $\hat{\mathcal{J}}$ and $\mathcal{J}$, quality of cost function F, offline and online algorithms (without predictions) OFFLINEALG and ONLINEALG for problem F, confidence level $\lambda \in (0, 1]$.

1: **for** $t \geq 0$ **do**
2: **if** $\texttt{OFF}(\mathcal{J}_{\leq t}) > \lambda \cdot \texttt{OFF}(\hat{\mathcal{J}})$ **then**
3: $t_\lambda \leftarrow t$
4: **break**
5: $\{s_j(t)\}_{j \in \mathcal{J}_{\leq t}} \leftarrow \text{ONLINEALG}(\mathcal{J}_{\leq t})(t)$
6: $\{\hat{s}_j(t)\}_{t \geq t_\lambda, j \in \hat{\mathcal{J}}_{\geq t_\lambda}} \leftarrow \text{OFFLINEALG}(\hat{\mathcal{J}}_{\geq t_\lambda})$
7: **for** $t \geq t_\lambda$ **do**
8: $\{s_j(t)\}_{j \in \mathcal{J}_{\leq t} \setminus \hat{\mathcal{J}}_{\geq t_\lambda}} \leftarrow \text{ONLINEALG}(\mathcal{J}_{\leq t} \setminus \hat{\mathcal{J}}_{\geq t_\lambda})(t)$
9: $\{s_j(t)\}_{j \in \mathcal{J}_{[t_\lambda, t]} \cap \hat{\mathcal{J}}_{\geq t_\lambda}} \leftarrow \{\hat{s}_j(t)\}_{j \in \mathcal{J}_{[t_\lambda, t]} \cap \hat{\mathcal{J}}_{\geq t_\lambda}}$
10: **return** $\{s_j(t)\}_{t \geq 0, j \in \mathcal{J}}$

In the second phase (Lines 6-9), TPE starts leveraging the predictions. More precisely, TPE needs to set the speeds for three different types of jobs: (1) the remaining jobs that were correctly predicted (i.e., $\mathcal{J}_{\geq t_\lambda} \cap \hat{\mathcal{J}}_{\geq t_\lambda}$) (2) the remaining jobs that were not predicted (i.e., $\mathcal{J}_{\geq t_\lambda} \setminus \hat{\mathcal{J}}_{\geq t_\lambda}$) (3) the jobs that were not correctly scheduled in the first phase and still have work remaining at the switch point t_λ (which are a subset of $\mathcal{J}_{<t_\lambda}$). To schedule these jobs, TPE combines two different schedules. The first one is the offline schedule $\hat{S} := \text{OFFLINEALG}(\hat{\mathcal{J}}_{\geq t_\lambda})$ for the jobs $\hat{\mathcal{J}}_{\geq t_\lambda}$ that are predicted to arrive in the second phase. Each future job in the true set that was correctly predicted (i.e., $j \in \mathcal{J}_{[t_\lambda, t]} \cap \hat{\mathcal{J}}_{\geq t_\lambda}$ on Line 9) will then be scheduled by following $\hat{S}$. The second schedule is an online schedule for the set of jobs $\mathcal{J} \setminus \hat{\mathcal{J}}_{\geq t_\lambda} = \mathcal{J}_{<t_\lambda} \cup \mathcal{J}_{\geq t_\lambda} \setminus \hat{\mathcal{J}}_{\geq t_\lambda}$, which includes all jobs that have not been completed during the first phase ($\subseteq \mathcal{J}_{<t_\lambda}$) and the incorrectly predicted jobs that are released during the second phase ($\mathcal{J}_{[t_\lambda, t]} \setminus \hat{\mathcal{J}}_{\geq t_\lambda}$). This online schedule is constructed by running ONLINEALG on the set $\mathcal{J} \setminus \hat{\mathcal{J}}_{\geq t_\lambda}$ (Line 8). Note that the total speed of the machine at each time step is the sum of the speeds of these two online and offline schedules.

3.2 Analysis of the algorithm

We analyze the competitive ratio of TPE as a function of the prediction error η, from which the consistency and robustness bounds follow. Missing proofs are provided in Appendix B. We separately bound the cost of the algorithm due to jobs in $\mathcal{J}_{<t_\lambda}$, $\mathcal{J}_{\geq t_\lambda} \setminus \hat{\mathcal{J}}_{\geq t_\lambda}$ and $\mathcal{J}_{\geq t_\lambda} \cap \hat{\mathcal{J}}_{\geq t_\lambda}$. We do this by analyzing the costs of schedules $S^{on} := \text{ONLINEALG}(\mathcal{J} \setminus \hat{\mathcal{J}}_{\geq t_\lambda})$ and $\hat{\bar{S}} := \text{OFFLINEALG}(\hat{\mathcal{J}}_{\geq t_\lambda})$. In the next lemma, we first analyze the cost of combining, i.e., summing, two arbitrary schedules.

Lemma 3.1. *Let $\mathcal{J}_1$ be a set of jobs and S_1 be a feasible schedule for $\mathcal{J}_1$, let $\mathcal{J}_2$ be a set of jobs and S_2 be a feasible schedule for $\mathcal{J}_2$. Consider the schedule $S := S_1 + S_2$ for $\mathcal{J}_1 \cup \mathcal{J}_2$ which, at each time t, runs the machine at total speed $s(t) = s_1(t) + s_2(t)$ and processes each job $j \in \mathcal{J}_1$ at speed $s_{1,j}(t)$ and each job $j \in \mathcal{J}_2$ at speed $s_{2,j}(t)$. Then, cost$(S, \mathcal{J}_1 \cup \mathcal{J}_2) \leq \left(\text{cost}(S_1, \mathcal{J}_1)^{\frac{1}{\alpha}} + \text{cost}(S_2, \mathcal{J}_2)^{\frac{1}{\alpha}}\right)^\alpha$.*

We next upper bound the cost of the schedule output by TPE as a function of the prediction error η, which we decompose into $\eta_1 = \frac{\texttt{OPT}(\mathcal{J} \setminus \hat{\mathcal{J}})}{\texttt{OPT}(\hat{\mathcal{J}})}$ and $\eta_2 = \frac{\texttt{OPT}(\hat{\mathcal{J}} \setminus \mathcal{J})}{\texttt{OPT}(\hat{\mathcal{J}})}$. The proof uses the previous lemma repeatedly, first to analyze the cost of the schedule $S^{on} := \text{ONLINEALG}(\mathcal{J} \setminus \hat{\mathcal{J}}_{\geq t_\lambda})$ for the set of jobs $\mathcal{J} \setminus \hat{\mathcal{J}}_{\geq t_\lambda} = (\mathcal{J}_{<t_\lambda}) \cup (\mathcal{J}_{\geq t_\lambda} \setminus \hat{\mathcal{J}}_{\geq t_\lambda})$, then to analyze the cost of the final schedule, which combines S^{on} and $\hat{S} := \text{OFFLINEALG}(\hat{\mathcal{J}}_{\geq t_\lambda})$.

Lemma 3.2. *Assume that* OFFLINEALG *is γ_{off}-competitive and that* ONLINEALG *is γ_{on}-competitive. Then, for all $\lambda \in (0, 1]$, the schedule S output by* TPE *run with confidence parameter λ satisfies cost$(S, \mathcal{J}) \leq \texttt{OPT}(\hat{\mathcal{J}}) \left(\gamma_{off}^{\frac{1}{\alpha}} + \gamma_{on}^{\frac{1}{\alpha}}((\lambda\gamma_{off})^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}})\right)^\alpha$.*

Proof. We start by upper bounding $\text{cost}(S^{on}, \mathcal{J} \setminus \hat{\mathcal{J}}_{\geq t_\lambda})$. First, by the algorithm, we have that $\texttt{OFF}(\mathcal{J}_{<t_\lambda}) \leq \lambda \cdot \texttt{OFF}(\hat{\mathcal{J}})$. Since OFFLINEALG is γ_{off}-competitive, we get

$$\texttt{OPT}(\mathcal{J}_{<t_\lambda}) \leq \texttt{OFF}(\mathcal{J}_{<t_\lambda}) \leq \lambda \cdot \texttt{OFF}(\hat{\mathcal{J}}) \leq \lambda\gamma_{\text{off}} \cdot \texttt{OPT}(\hat{\mathcal{J}}).$$

We also have that $\texttt{OPT}(\mathcal{J}_{\geq t_\lambda} \setminus \hat{\mathcal{J}}_{\geq t_\lambda}) \leq \texttt{OPT}(\mathcal{J} \setminus \hat{\mathcal{J}}) \leq \eta_1 \texttt{OPT}(\hat{\mathcal{J}})$ where the first inequality is since $\mathcal{J}_{\geq t_\lambda} \setminus \hat{\mathcal{J}}_{\geq t_\lambda} \subseteq \mathcal{J} \setminus \hat{\mathcal{J}}$ and the second is by definition of η_1. Recall that $S^*(.)$ denotes the optimal offline schedule for the problem and consider the schedule $S' = S^\star(\mathcal{J}_{<t_\lambda}) + S^\star(\mathcal{J}_{\geq t_\lambda} \setminus \hat{\mathcal{J}}_{\geq t_\lambda})$ for $\mathcal{J} \setminus \hat{\mathcal{J}}_{\geq t_\lambda} = \mathcal{J}_{<t_\lambda} \cup (\mathcal{J}_{\geq t_\lambda} \setminus \hat{\mathcal{J}}_{\geq t_\lambda})$. We obtain that

$$\begin{aligned}
\texttt{OPT}(\mathcal{J} \setminus \hat{\mathcal{J}}_{\geq t_\lambda}) &\leq \text{cost}(S', \mathcal{J} \setminus \hat{\mathcal{J}}_{\geq t_\lambda}) \leq \left(\texttt{OPT}(\mathcal{J}_{<t_\lambda})^{\frac{1}{\alpha}} + \texttt{OPT}(\mathcal{J}_{\geq t_\lambda} \setminus \hat{\mathcal{J}}_{\geq t_\lambda})^{\frac{1}{\alpha}}\right)^\alpha \\
&\leq \left((\lambda\gamma_{\text{off}}\texttt{OPT}(\hat{\mathcal{J}}))^{\frac{1}{\alpha}} + (\eta_1 \texttt{OPT}(\hat{\mathcal{J}}))^{\frac{1}{\alpha}})\right)^\alpha \\
&= \texttt{OPT}(\hat{\mathcal{J}}) \left((\lambda\gamma_{\text{off}})^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}}\right)^\alpha,
\end{aligned}$$

where the second inequality is by Lemma 3.1. Since we assumed that ONLINEALG is γ_{on}-competitive,

$$\text{cost}(S^{on}, \mathcal{J} \setminus \hat{\mathcal{J}}_{\geq t_\lambda}) \leq \gamma_{\text{on}} \cdot \texttt{OPT}(\mathcal{J} \setminus \hat{\mathcal{J}}_{\geq t_\lambda}) \leq \gamma_{\text{on}} \cdot \texttt{OPT}(\hat{\mathcal{J}}) \left((\lambda\gamma_{\text{off}})^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}}\right)^\alpha.$$

We now bound the cost of schedule S. First, note that $\text{cost}(\hat{S}, \hat{\mathcal{J}}_{\geq t_\lambda}) = \text{OFFLINEALG}(\hat{\mathcal{J}}_{\geq t_\lambda}) \leq \gamma_{\text{off}} \cdot \texttt{OPT}(\hat{\mathcal{J}}_{\geq t_\lambda}) \leq \gamma_{\text{off}} \cdot \texttt{OPT}(\hat{\mathcal{J}})$, where the first inequality is since OFFLINEALG is γ_{off}-competitive and the last one since $\hat{\mathcal{J}}_{\geq t_\lambda} \subseteq \hat{\mathcal{J}}$. Therefore, by applying again Lemma 3.1, we get:

$$\begin{aligned}
\text{cost}(S, \mathcal{J}) &\leq \left(\text{cost}(\hat{S}, \hat{\mathcal{J}}_{\geq t_\lambda})^{\frac{1}{\alpha}} + \text{cost}(S^{on}, \mathcal{J} \setminus \hat{\mathcal{J}}_{\geq t_\lambda})^{\frac{1}{\alpha}}\right)^\alpha \\
&\leq \left((\gamma_{\text{off}} \cdot \texttt{OPT}(\hat{\mathcal{J}}))^{\frac{1}{\alpha}} + \left(\gamma_{\text{on}} \cdot \texttt{OPT}(\hat{\mathcal{J}}) \left((\lambda\gamma_{\text{off}})^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}}\right)^\alpha\right)^{\frac{1}{\alpha}}\right)^\alpha \\
&= \texttt{OPT}(\hat{\mathcal{J}}) \left(\gamma_{\text{off}}^{\frac{1}{\alpha}} + \gamma_{\text{on}}^{\frac{1}{\alpha}}((\lambda\gamma_{\text{off}})^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}})\right)^\alpha. \qquad \square
\end{aligned}$$

We next state a simple corollary of Lemma 3.1.

Corollary 3.3. *$\texttt{OPT}(\mathcal{J} \cap \hat{\mathcal{J}}) \geq \left(1 - \eta_2^{\frac{1}{\alpha}}\right)^\alpha \texttt{OPT}(\hat{\mathcal{J}})$, and, assuming that* OFFLINEALG *is γ_{off}-competitive, we have: if $\texttt{OFF}(\mathcal{J}) \leq \lambda \texttt{OFF}(\hat{\mathcal{J}})$, then $\eta_2 \geq \left(1 - (\lambda\gamma_{off})^{\frac{1}{\alpha}}\right)^\alpha$.*

We are ready to state the main result of this section, which is our upper bound on the competitive ratio of TPE.

Theorem 3.4. *For any $\lambda \in (0, 1]$,* TPE *with a γ_{on}-competitive algorithm* ONLINEALG *and a γ_{off}-competitive offline algorithm* OFFLINEALG *achieves a competitive ratio of*

$$\begin{cases}
\gamma_{on} & \text{if } \texttt{OFF}(\mathcal{J}) \leq \lambda \texttt{OFF}(\hat{\mathcal{J}}) \\
\dfrac{\left(\gamma_{off}^{\frac{1}{\alpha}} + \gamma_{on}^{\frac{1}{\alpha}}((\lambda\gamma_{off})^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}})\right)^\alpha}{\max\left\{\frac{\lambda}{\gamma_{off}}, \eta_1 + \left(1 - \eta_2^{\frac{1}{\alpha}}\right)^\alpha\right\}} & \text{otherwise.}
\end{cases}$$

The consistency and robustness immediately follow (for simplicity, we present the results in the case where OFFLINEALG is optimal). Additional discussion on this competitive ratio is provided in Appendix 3.4.

Corollary 3.5. *For any $\lambda \in (0, 1)$,* TPE *with a γ_{on}-competitive algorithm* ONLINEALG *and an optimal offline algorithm* OFFLINEALG *is $1 + \gamma_{on} 2^\alpha \lambda^{\frac{1}{\alpha}}$ competitive if $\eta_1 = \eta_2 = 0$ (consistency) and $\max\{\gamma_{on}, \frac{1 + \gamma_{on} 2^{2\alpha} \lambda^{\frac{1}{\alpha}}}{\lambda}\}$-competitive for all η_1, η_2 (robustness). In particular, for any constant $\epsilon > 0$, with $\lambda = (\frac{\epsilon}{\gamma_{on} 2^\alpha})^\alpha$,* TPE *is $1 + \epsilon$-consistent and $O(1)$-robust.*

3.3 Results for well-studied GES problems

We apply the general framework detailed in Section 3 to derive smooth, consistent and robust algorithms for a few classically studied objective functions.

Energy plus flow time minimization. Recall that c_S^j denote the completion time of job j. The quality cost function is defined as: $F(S, \mathcal{J}) = \sum_{j \in \mathcal{J}} (c_S^j - r_j)$, with total objective cost$(S, \mathcal{J}) = F(S, \mathcal{J}) + E(S)$. By a direct application of Corollary 3.5, we get that for all $\lambda \in (0, 1]$, Algorithm 1 run with the 2-competitive online algorithm from [3] and confidence parameter λ is $(1 + 2^{\frac{1}{\alpha}} \lambda^{\frac{1}{\alpha}})^\alpha$-consistent and $\frac{1+2 \cdot 2^{2\alpha} \lambda^{\frac{1}{\alpha}}}{\lambda}$-robust.

Energy plus fractional weighted flow time minimization. In this setting, each job has a weight v_j. The quality cost is $F(S, \mathcal{J}) = \sum_{j \in \mathcal{J}} v_j \int_{t \geq r_j} w_j^S(t) \mathrm{dt}$. We can use as ONLINEALG the 2-competitive algorithm from [14].

Energy plus integral weighted flow time minimization. In this setting, each job has a weight v_j. The quality cost function is defined as: $F(S, \mathcal{J}) = \sum_{j \in \mathcal{J}} v_j (c_S^j - r_j)$. We can use as ONLINEALG the $O((\alpha / \log \alpha)^2)$-competitive algorithm from [12].

Energy minimization with deadlines. In this setting, there is also a deadline d_j for the completion of each job. By writing the quality cost as $F(S, \mathcal{J}) = \sum_{j \in \mathcal{J}} \delta_{c_S^j > d_j}$, where $\delta_{c_S^j > d_j} = +\infty$ if $c_S^j > d_j$ and 0 otherwise, the total objective can be written as cost$(S, \mathcal{J}) = E(S) + F(S, \mathcal{J})$. We can use as ONLINEALG the AVERAGE RATE heuristic [29] (which is 2^α-competitive for uniform deadlines [7]). In particular, for uniform deadlines, and for all $\epsilon \in (0, 1]$, by setting $\lambda = (\frac{\epsilon}{C 2^\alpha})^\alpha$, we obtain a consistency of $(1 + \epsilon)$ for a robustness factor of $O(4^{\alpha^2} / \epsilon^{\alpha - 1})$.

3.4 Discussion on the competitive ratio

We assume in this section that OFFLINEALG is optimal. Note that for small η_1 and η_2, the competitive ratio is upper bounded as $\frac{\left(1 + \gamma_{\text{on}}^{\frac{1}{\alpha}} (\lambda^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}})\right)^\alpha}{\eta_1 + \left(1 - \eta_2^{\frac{1}{\alpha}}\right)^\alpha}$, which smoothly goes to $(1 + \gamma_{\text{on}}^{\frac{1}{\alpha}} \lambda^{\frac{1}{\alpha}})^\alpha$ (consistency case) when η_1, η_2 go to 0. Moreover, our upper bound distinguishes the effect of two possible sources of errors on the algorithm: (1) when removing jobs from the prediction ($\eta_1 = 0$ and η_2 goes to 1), the upper bound degrades monotonically to $O(\frac{1}{\lambda})$. (2) when adding jobs to the prediction ($\eta_2 = 0$ and η_1 goes to $+\infty$), the upper bound first degrades, then improves again, with an optimal asymptotic rate of γ_{on}. This is since our algorithm mostly follows the online algorithm when the cost of the additional jobs dominates.

4 The Extension to Small Deviations

Note that in the definition of the prediction error η, a job j is considered to be correctly predicted only if $r_j = \hat{r}_j$ and $p_j = \hat{p}_j$. In this extension, we consider that a job is correctly predicted even if its release time and processing time are shifted by a small amount. We also allow each job to have some weight $v_j > 0$, that can be shifted as well. Assuming an additional smoothness condition on the quality cost function $F(.,.)$, which is satisfied for the energy plus flow time minimization problem and its variants, we propose and analyze an algorithm that generalizes the algorithm from the previous section.

The algorithm, called TPE-S and formally described in Appendix C, takes the same input parameters as Algorithm TPE, with some additional shift tolerance parameter $\eta^{\text{shift}} \in [0, 1)$ that is chosen by the algorithm designer. Two main ideas are to artificially increase the predicted processing time $\hat{p}_j$ of each job j (because the true processing time p_j of job j could be shifted and be slightly larger than $\hat{p}_j$) and to introduce small delays for the job speeds (because the true release time r_j of some jobs j could be shifted and be slightly later than $\hat{r}_j$). Details can be found in Appendix C.

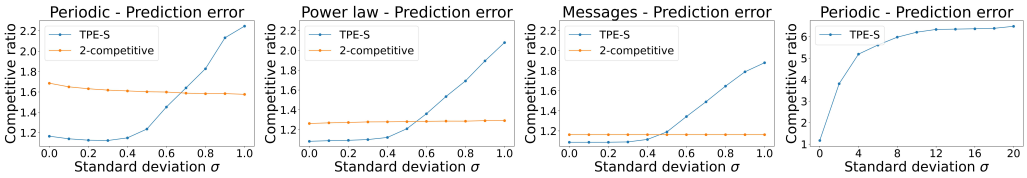

Figure 1: The competitive ratio achieved by our algorithm, TPE-S, and the benchmark algorithm, as a function of the error parameter σ (from left-most to the second from the right), and the competitive ratio of TPE-S for a larger range of σ, as a function of σ (right-most).

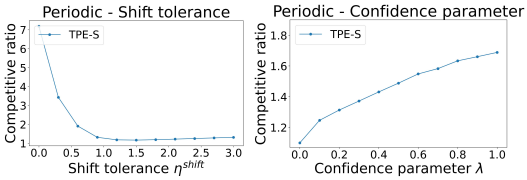

Figure 2: The competitive ratio achieved by our algorithm, TPE-S, as a function of the shift tolerance parameter η^{shift} (left) and as a function of the confidence parameter λ (right).

5 Experiments

We empirically evaluate the performance of Algorithm TPE-S on both synthetic and real datasets. Specifically, we consider the energy plus flow time minimization problem where $F(S, \mathcal{J}) = \sum_{j \in \mathcal{J}} c_j - r_j$ and consider unit-work jobs (i.e., $p_j = 1$ for all j) and fix $\alpha = 3$.

5.1 Experimental setting

Benchmarks. **TPE-S** is Algorithm 2 with the default setting $\lambda = 0.02$, $\eta^{\text{shift}} = 1$ and $\sigma = 0.4$, where σ is a parameter that controls the level of prediction error, that we call the error parameter. **2-COMPETITIVE** is the 2-competitive online algorithm from [3] that sets the speed at each time t to $n(t)^{\frac{1}{\alpha}}$, where $n(t)$ is the number of jobs with $r_j \leq t$ unfinished at time t, and uses the Shortest Remaining Processing Time rule.

Data sets. We consider two synthetic datasets and a real dataset. For the synthetic data, we first generate a predicted set of jobs $\hat{\mathcal{J}}$ and we fix the value of the error parameter $\sigma > 0$. To create the true set of jobs $\mathcal{J}$, we generate, for each job $j \in \hat{\mathcal{J}}$, some error $err(j)$ sampled i.i.d. from $\mathcal{N}(0, \sigma)$. The true set of jobs is then defined as $\mathcal{J} = \{(j, \hat{r}_j + err(j)) : j \in \hat{\mathcal{J}}\}$, which is the set of all predicted jobs, shifted according to $\{err(j)\}$. Note that for all $j \in \mathcal{J}$, $j \in \mathcal{J}^{\text{shift}}$ only if $|\hat{r}_j - r_j| = |err(j)| < \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})} \cdot \frac{\texttt{OPT}(\hat{\mathcal{J}})}{|\hat{\mathcal{J}}|}$. Hence, a larger $\sigma > 0$ corresponds to a larger prediction error η^g. For the first synthetic dataset, called the periodic dataset, the prediction is a set of $n = 300$ jobs, with i^{th} job's arrival $r_i = i/\alpha$. For the second synthetic dataset, we generate the prediction by using a power-law distribution. More precisely, for each time $t \in \{1, \ldots, T\}$, where we fix $T = 75$, the number of jobs' arrivals at time t is set to $M(1 - p(a))$, where $p(a)$ is sampled from a power law distribution of parameter a, and M is some scaling parameter. In all experiments, we use the values $a = 100$, $M = 500$.

We also evaluate the two algorithms on the College Message dataset from the SNAP database [27], where the scheduler must process messages that arrive over 9 days, each with between 300 and 500 messages. We first fix the error parameter $\sigma > 0$, then, for each day, we define the true set $\mathcal{J}$ as the arrivals for this day, and we create the predictions $\hat{\mathcal{J}}$ by adding some error $err(i)$ to the release time of each job i, where $err(i)$ is sampled i.i.d. from $\mathcal{N}(0, \sigma)$.

5.2 Experiment results

For each of the synthetic datasets, the competitive ratio achieved by the different algorithms is averaged over 10 instances generated i.i.d., and for the real dataset, it is averaged over the arrivals for each of the 9 days.

Experiment set 1. We first evaluate the performance of the algorithms as a function of the error parameter σ. In Figure 1, we observe that TPE-S outperforms 2-COMPETITIVE when the error parameter is small. In the right-most figure of Figure 1, the competitive ratio of TPE-S plateaus when the value of σ increases, which is consistent with our bounded robustness guarantee.

Experiment set 2. In the second set of experiments, we study the impact of the parameters η^{shift} and λ of the algorithm for the periodic dataset (results for the other datasets can be found in Appendix E) and fix $\sigma = 0.4$. In the left plot of Figure 2, we observe the importance of allowing some shift in the predictions as the performance of our algorithms first rapidly improves as a function of η^{shift} and then slowly deteriorates. The rapid improvement is because an increasing number of jobs are treated by the algorithm as being correctly predicted when η^{shift} increases. Next, in the right plot, we observe that the competitive ratio deteriorates as a function of λ, which implies that the algorithm can completely skip the first phase that ignores the predictions and run the second phase that combines the offline and online schedules when the prediction error is not too large. Note, however, that a larger value of λ leads to a better competitive ratio when the predictions are incorrect. Hence, there is a general trade-off here.

6 Limitations

The results in Section 3 and Section 4 require the quality cost function F to be monotone subadditive, which holds for the flow time problem and the problem with deadlines but might not hold for some other energy-efficient scheduling problems. The results in Section 4 require an additional smoothness assumption on F, which holds for the flow time problem but not for the problem with deadlines. Finally, we have only tested our algorithm on the three datasets described in Section 5.

Acknowledgements

Eric Balkanski was supported by NSF grants CCF-2210502 and IIS-2147361. Clifford Stein was supported in part by NSF grant CCF-2218677 and ONR grant ONR-13533312, and by the Wai T. Chang Chair in Industrial Engineering and Operations Research.

References

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

[2] Susanne Albers and Hiroshi Fujiwara. Energy-efficient algorithms for flow time minimization. *ACM Transactions on Algorithms (TALG)*, 3(4):49–es, 2007.

[3] Lachlan LH Andrew, Adam Wierman, and Ao Tang. Optimal speed scaling under arbitrary power functions. *ACM SIGMETRICS Performance Evaluation Review*, 37(2):39–41, 2009.

[4] Antonios Antoniadis, Peyman Jabbarzade Ganje, and Golnoosh Shahkarami. A novel prediction setup for online speed-scaling. In Artur Czumaj and Qin Xin, editors, *18th Scandinavian Symposium and Workshops on Algorithm Theory, SWAT 2022, June 27-29, 2022, Tórshavn, Faroe Islands*, volume 227 of *LIPIcs*, pages 9:1–9:20, 2022. doi: 10.4230/LIPIcs.SWAT.2022.9. URL `https://doi.org/10.4230/LIPIcs.SWAT.2022.9`.

[5] Eric Balkanski, Vasilis Gkatzelis, and Xizhi Tan. Strategyproof scheduling with predictions. *arXiv preprint arXiv:2209.04058*, 2022.

[6] Eric Balkanski, Tingting Ou, Clifford Stein, and Hao-Ting Wei. Scheduling with speed predictions. *arXiv preprint arXiv:2205.01247*, 2022.

[7] Étienne Bamas, Andreas Maggiori, Lars Rohwedder, and Ola Svensson. Learning augmented energy minimization via speed scaling. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/af94ed0d6f5acc95f97170e3685f16c0-Abstract.html`.

[8] Nikhil Bansal and Ho-Leung Chan. Weighted flow time does not admit o (1)-competitive algorithms. In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*, pages 1238–1244. SIAM, 2009.

[9] Nikhil Bansal and Kirk Pruhs. The geometry of scheduling. *SIAM J. Comput.*, 43(5):1684–1698, 2014. doi: 10.1137/130911317. URL `https://doi.org/10.1137/130911317`.

[10] Nikhil Bansal, Tracy Kimbrel, and Kirk Pruhs. Speed scaling to manage energy and temperature. *Journal of the ACM (JACM)*, 54(1):1–39, 2007.

[11] Nikhil Bansal, Ho-Leung Chan, Tak-Wah Lam, and Lap-Kei Lee. Scheduling for speed bounded processors. In *International Colloquium on Automata, Languages, and Programming*, pages 409–420. Springer, 2008.

[12] Nikhil Bansal, Kirk Pruhs, and Cliff Stein. Speed scaling for weighted flow time. *SIAM Journal on Computing*, 39(4):1294–1308, 2010.

[13] Nikhil Bansal, David P Bunde, Ho-Leung Chan, and Kirk Pruhs. Average rate speed scaling. *Algorithmica*, 60(4):877–889, 2011.

[14] Nikhil Bansal, Ho-Leung Chan, and Kirk Pruhs. Speed scaling with an arbitrary power function. *ACM Transactions on Algorithms (TALG)*, 9(2):1–14, 2013.

[15] David M Brooks, Pradip Bose, Stanley E Schuster, Hans Jacobson, Prabhakar N Kudva, Alper Buyuktosunoglu, John Wellman, Victor Zyuban, Manish Gupta, and Peter W Cook. Power-aware microarchitecture: Design and modeling challenges for next-generation microprocessors. *IEEE Micro*, 20(6):26–44, 2000.

[16] Woo-Hyung Cho, Shane Henderson, and David Shmoys. Scheduling with predictions. *arXiv preprint arXiv:2212.10433*, 2022.

[17] Sungjin Im, Ravi Kumar, Mahshid Montazer Qaem, and Manish Purohit. Non-clairvoyant scheduling with predictions. In *Proceedings of the 33rd ACM Symposium on Parallelism in Algorithms and Architectures*, pages 285–294, 2021.

[18] Tak-Wah Lam, Lap-Kei Lee, Isaac KK To, and Prudence WH Wong. Speed scaling functions for flow time scheduling based on active job count. In *European Symposium on Algorithms*, pages 647–659. Springer, 2008.

[19] 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)*, pages 1859–1877, 2020.

[20] Russell Lee, Jessica Maghakian, Mohammad Hajiesmaili, Jian Li, Ramesh Sitaraman, and Zhenhua Liu. Online peak-aware energy scheduling with untrusted advice. *ACM SIGENERGY Energy Informatics Review*, 1(1):59–77, 2021.

[21] Alexander Lindermayr and Nicole Megow. Permutation predictions for non-clairvoyant scheduling. In *Proceedings of the 34th ACM Symposium on Parallelism in Algorithms and Architectures*, pages 357–368, 2022.

[22] Alexander Lindermayr, Nicole Megow, and Martin Rapp. Speed-oblivious online scheduling: Knowing (precise) speeds is not necessary. *arXiv preprint arXiv:2302.00985*, 2023.

[23] Thodoris Lykouris and Sergei Vassilvitskii. Competitive caching with machine learned advice. *J. ACM*, 68(4):24:1–24:25, 2021. doi: 10.1145/3447579. URL `https://doi.org/10.1145/3447579`.

[24] Mohammad Mahdian, Hamid Nazerzadeh, and Amin Saberi. Allocating online advertisement space with unreliable estimates. In *Proceedings of the 8th ACM conference on Electronic commerce*, pages 288–294, 2007.

[25] Michael Mitzenmacher. Scheduling with Predictions and the Price of Misprediction. In *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*, volume 151 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 14:1–14:18, 2020. ISBN 978-3-95977-134-4.

[26] Trevor Mudge. Power: A first-class architectural design constraint. *Computer*, 34(4):52–58, 2001.

[27] Pietro Panzarasa, Tore Opsahl, and Kathleen M Carley. Patterns and dynamics of users' behavior and interaction: Network analysis of an online community. *Journal of the American Society for Information Science and Technology*, 60(5):911–932, 2009.

[28] Manish Purohit, Zoya Svitkina, and Ravi Kumar. Improving online algorithms via ml predictions. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2018.

[29] Frances Yao, Alan Demers, and Scott Shenker. A scheduling model for reduced cpu energy. In *Proceedings of IEEE 36th annual foundations of computer science*, pages 374–382. IEEE, 1995.

Appendix

A Consistent algorithms are not robust

In this section, we show that any learning-augmented algorithm for the (GSSP) problem must incur some trade-off between robustness and consistency. Note that some impossibility results for general objective functions of the form $\text{cost}(S, \mathcal{J}) = E(S) + F(S, \mathcal{J})$ given in Section 2 follow immediately from [7], since the problem of speed scaling with deadline constraints was studied there is a special case of (GSSP) (see Section 3.3).

We prove here some impossibility results for a different family of objective functions, where the objective is to maximize the total energy plus flow time. This is one of the most widely studied objectives of the form in $\text{cost}(S, \mathcal{J}) = E(S) + F(S, \mathcal{J})$ given in Section 2 (see for instance[2, 3, 12, 14]). Here, for all $j \in \mathcal{J}$, we let c_S^j denote the completion time of job j while following schedule S. The quality cost function studied in the remainder of this section is defined as: $F(S, \mathcal{J}) = \sum_{j \in \mathcal{J}} (c_S^j - r_j)$ and the total objective is $\text{cost}(S, \mathcal{J}) = F(S, \mathcal{J}) + E(S)$. We recall that for this problem, the best possible online algorithm is the 2-competitive algorithm from [3].

A.1 Warm-up: no 1-consistent algorithm is robust

We show in this section that no algorithm that is perfectly consistent (i.e., achieves an optimal cost when the prediction is totally correct) can have a bounded competitive ratio in the case the prediction is incorrect. To show this property, we build an instance where a lot of jobs are predicted, but only one of them arrives. To achieve consistency, any online algorithm must 'burn' a lot of energy during the first few time steps; however, in the case where only one job arrives, the algorithm ends up having wasted too much energy. This illustrates the necessity of a trade-off between robustness and consistency.

Proposition A.1. *For the objective of minimizing total energy plus (non-weighted) flow time, there is no algorithm that is* 1*-consistent and* $o(\sqrt{n})$*-robust, even if all jobs have unit-size work and if* $\mathcal{J} \subseteq \hat{\mathcal{J}}$*.*

Proof. Set $\alpha = 2$ and consider an instance $(\hat{\mathcal{J}}, \mathcal{J}')$ where $\hat{\mathcal{J}}$ contains n jobs of unit-size work such that the first job arrives at time $t = 0$ and the remaining $n - 1$ jobs arrive at time $t = \frac{1}{\sqrt{n}}$, and $\mathcal{J}'$ contains only the job that arrives at time $t = 0$.

By using results from [2], the optimal offline schedule for $\hat{\mathcal{J}}$ is to schedule each job $i \in [n]$ at speed $\sqrt{n - i + 1}$. Moreover, processing the first job any slower leads to a strictly worse cost. Hence, any algorithm that is 1-consistent (i.e, achieves an optimal competitive ratio when the realization is exactly $\hat{\mathcal{J}}$) must process the first job at speed $s_1(t) = \sqrt{n}$ for all $t \in [0, \frac{1}{\sqrt{n}}]$. In this case, the total objective is at least $\sqrt{n}^2 \cdot 1/\sqrt{n} + 1/\sqrt{n} = \sqrt{n} + 1/\sqrt{n}$.

However, by using results from [2], the optimal objective for $\mathcal{J}'$ is 2 (with the speed of the single job arriving at time $t = 0$ being set to 1). Hence, in the case where the realization is $\mathcal{J}'$, any algorithm that schedules the first job at speed $s_1(t) = \sqrt{n}$ has a competitive ratio at least $\frac{\sqrt{n}+1/\sqrt{n}}{2}$.

Therefore, any 1-consistent algorithm must have a robustness factor of at least $\frac{\sqrt{n}+1/\sqrt{n}}{2}$. □

A.2 Consistency-robustness trade-off

In this section, we quantify more precisely the necessary trade-off between robustness and consistency. More precisely, we prove that there is a constant $C > 0$ such that for any λ small enough, any algorithm that is at most $(1 + \lambda)$ consistent must be at least $C\sqrt{\frac{1}{\lambda} + 1}$ robust. Moreover, letting $n_{\leq t}(\mathcal{J}) = |\{j \in \mathcal{J} : r_j \leq t\}|$ be the number of jobs of $\mathcal{J}$ that arrived before time t, we show that for the following natural notion of error:

$$\tilde{\eta}(\mathcal{J}, \mathcal{J}') = \frac{1}{\max\{|\mathcal{J}|, |\mathcal{J}'|\}} \max_{t \geq 0} \{|n_{\leq t}(\mathcal{J}) - n_{\leq t}(\mathcal{J}')|\},$$

which mimics the probability density function of the predicted and realized jobs, this property remains true even if we assume a small prediction error $\tilde{\eta}(\mathcal{J}, \hat{\mathcal{J}})$. Hence, one cannot obtain a smooth algorithm relatively to this notion of error. This motivates the introduction of the more refined notion of error from Section 2. More specifically, we show the following lemma.

Lemma A.2. *For the objective of minimizing total energy plus (non-weighted) flow time, there are $\lambda' \in (0, 1]$ and $C > 0$ such that for any $\epsilon > 0$, there is $M \in \mathbb{N}$ such that for all $\lambda \leq \lambda'$ and $n \geq M$, there is an instance $(\hat{\mathcal{J}}_{n,\lambda,\epsilon}, \mathcal{J}_{n,\lambda,\epsilon})$ such that $|\hat{\mathcal{J}}_{n,\lambda,\epsilon}| = n$ and $\tilde{\eta}(\hat{\mathcal{J}}_{n,\lambda,\epsilon}, \mathcal{J}_{n,\lambda,\epsilon}) \leq \epsilon$, and such that for any algorithm $\mathcal{A}$,*

- *either $\text{cost}_{\mathcal{A}}(\hat{\mathcal{J}} = \hat{\mathcal{J}}_{n,\lambda,\epsilon}, \mathcal{J} = \hat{\mathcal{J}}_{n,\lambda,\epsilon}) > (1 + \lambda) \cdot \texttt{OPT}(\hat{\mathcal{J}}_{n,\lambda,\epsilon})$ **(large consistency factor)***
- *or $\text{cost}_{\mathcal{A}}(\hat{\mathcal{J}} = \hat{\mathcal{J}}_{n,\lambda,\epsilon}, \mathcal{J} = \mathcal{J}_{n,\lambda,\epsilon}) \geq C\sqrt{\frac{1}{\lambda} + 1} \cdot \texttt{OPT}(\mathcal{J}_{n,\lambda,\epsilon})$ **(large robustness factor)**.*

The rest of this section is dedicated to the proof of Lemma A.2.

We first describe our lower bound instance. In the remainder of this section, we set $\alpha = 2$.

Lower bound instance. Let $\lambda \in (0, 1], n \in \mathbb{N}$ and $\epsilon > 0$. We construct an instance $(\hat{\mathcal{J}}_{n,\lambda,\epsilon}, \mathcal{J}_{n,\lambda,\epsilon})$ where the jobs in $\hat{\mathcal{J}}_{n,\lambda,\epsilon}$ can be organized in three different groups.

1. Group $A_{n,\lambda,\epsilon}$ is composed of $\frac{4}{3}\lambda\epsilon n$ jobs that all arrive at time 0.
2. Group $B_{n,\lambda,\epsilon}$ is composed of ϵn jobs that all arrive at time $t_A := \frac{4\lambda\epsilon n}{3\sqrt{\epsilon n(1+\frac{4}{3}\lambda)}}$.
3. Group $C_{n,\lambda,\epsilon}$ consists of n dummy jobs, where, for some $t' >> 0$, each job $j \in [n]$ arrives at time $t' + j$.

Next, we define $\mathcal{J}_{n,\lambda,\epsilon}$ as the union of jobs in $A_{n,\lambda,\epsilon}$ and $C_{n,\lambda,\epsilon}$. Note that by construction, we have $\tilde{\eta}(\hat{\mathcal{J}}_{n,\lambda,\epsilon}, \mathcal{J}_{n,\lambda,\epsilon}) \leq \epsilon$.

We now state and prove a few useful lemmas.

Lemma A.3. *Let $\mathcal{K}$ be a set of n jobs that all arrive at some time $t \geq 0$. Then, we have*

$$\frac{4}{3}n^{3/2} \leq \texttt{OPT}(\mathcal{K}) \leq \frac{4}{3}n^{3/2} + o(n^{3/2}).$$

Proof. By [2], the optimal schedule is to run each job i at speed $s_i = \sqrt{n - i + 1}$. The total cost is as follows:

$$\begin{aligned}
\text{cost}(S^*(\mathcal{K})) &= F(S^*(\mathcal{K})) + E(S^*(\mathcal{K})) \\
&= \sum_{i=1}^{n}\sum_{j=1}^{i}\frac{1}{\sqrt{n-j+1}} + \sum_{i=1}^{n}\frac{1}{\sqrt{n-i+1}}\sqrt{n-i+1}^2 \\
&= \sum_{j=1}^{n}\frac{1}{\sqrt{n-j+1}}\sum_{i=j}^{n}1 + \sum_{i=1}^{n}\sqrt{n-i+1} \\
&= 2\sum_{i=1}^{n}\sqrt{n-i+1}
\end{aligned}$$

Hence we have

$$\begin{aligned}
&2\int_0^n \sqrt{x}dx \leq \text{cost}(S^*(\mathcal{K})) \leq 2\int_1^{n+1}\sqrt{x}dx \\
\Rightarrow &\frac{4}{3}n^{3/2} \leq \text{cost}(S^*(\mathcal{K})) \leq \frac{4}{3}[(n+1)^{3/2} - 1] = \frac{4}{3}[n^{3/2} + o(n^{3/2})]
\end{aligned}$$

□

Lemma A.4. *Let $\mathcal{K} = \{j_1, \ldots, j_{|\mathcal{K}|}\}$ be a set of $|\mathcal{K}|$ jobs such that for all $i \in [|\mathcal{K}|]$, $|r_{i+1} - r_i| \geq 1$. Then, we have*

$$\texttt{OPT}(\mathcal{K}) = 2|\mathcal{K}|.$$

Proof. By using [2], the optimal solution is to run each job at speed 1. The result follows immediately. □

Lemma A.5. *Let $\lambda \in (0, 1], n \in \mathbb{N}$. Then, the optimal cost for jobs in $\hat{\mathcal{J}}_{n,\lambda,\epsilon}$ is upper bounded as follows:*

$$\texttt{OPT}(\hat{\mathcal{J}}_{n,\lambda,\epsilon}) \leq \frac{4}{3}(\epsilon n)^{3/2}(1 + \lambda + o(\lambda) + o(1))$$
$$\text{as } \lambda \to 0 \text{ (independently of } \epsilon), \ n \to +\infty \text{ (for a fixed } \epsilon).$$

Proof. Consider the schedule S which runs jobs in $A_{n,\lambda,\epsilon}$ at speed $\sqrt{\epsilon n(1 + \frac{4}{3}\lambda)}$, and jobs in $B_{n,\lambda,\epsilon}$ at the optimal speeds for ϵn jobs arriving at the same time, and jobs in $C_{n,\lambda,\epsilon}$ at speed 1. Consider the cost of S for all jobs in $A_{n,\lambda,\epsilon}$. Note that all the jobs in $A_{n,\lambda,\epsilon}$ are finished by time $t_A = \frac{4\lambda\epsilon n}{3\sqrt{\epsilon n(1+\frac{4}{3}\lambda)}}$. Hence, we have

$$\begin{aligned}
\text{cost}(S_{[0,t_A]}, A_{n,\lambda,\epsilon}) &\leq F(S_{[0,t_A]}, A_{n,\lambda,\epsilon}) + E(S_{[0,t_A]}) \\
&\leq t_A \frac{4}{3}\lambda\epsilon n + t_A \left(\sqrt{\epsilon n(1 + \frac{4}{3}\lambda)}\right)^2 \\
&\leq \frac{\frac{4}{3}\lambda\epsilon n \cdot \frac{4}{3}\lambda\epsilon n}{\sqrt{\epsilon n(1 + \frac{4}{3}\lambda)}} + \frac{4}{3}\lambda\epsilon n\sqrt{\epsilon n(1 + \frac{4}{3}\lambda)} \\
&= (\epsilon n)^{3/2} o(\lambda) + (\epsilon n)^{3/2}(\frac{4}{3}\lambda\sqrt{(1 + \frac{4}{3}\lambda)}) \\
&= (\epsilon n)^{3/2}[\frac{4}{3}\lambda + o(\lambda)].
\end{aligned}$$

Let t_B be the time at which S finishes all jobs in $B_{n,\lambda,\epsilon}$. Recall that all n jobs in $B_{n,\lambda,\epsilon}$ arrive at time t_A. By Lemma A.3, we have

$$\text{cost}(S_{[t_A,t_B]}, B_{n,\lambda,\epsilon}) \leq \frac{4}{3}(\epsilon n)^{3/2} + o((\epsilon n)^{3/2}),$$

Since all jobs in $C_{n,\lambda,\epsilon}$ arrive at time $t' >> t_B$, we have

$$\text{cost}(S_{\geq t_B}, C_{n,\lambda,\epsilon}) = 2n = o((\epsilon n)^{3/2}).$$

Therefore, when λ goes to 0 and n goes to $+\infty$, the total cost of S is upper bounded as follows:

$$\begin{aligned}
\text{cost}(S, \hat{\mathcal{J}}_{n,\lambda,\epsilon}) &= \text{cost}(S_{[0,t_A]}, A_{n,\lambda,\epsilon}) + \text{cost}(S_{[t_A,t_B]}, B_{n,\lambda,\epsilon}) + \text{cost}(S_{\geq t_B}, C_{n,\lambda,\epsilon}) \\
&\leq \frac{4}{3}(\epsilon n)^{3/2}[1 + \lambda + o(\lambda) + o(1)].
\end{aligned}$$

□

Lemma A.6. *There is $\lambda' \in (0, 1]$ such that for any $\epsilon > 0$, there is $M \in \mathbb{N}$ such that for all $\lambda \leq \lambda'$ and $n \geq M$, and for any schedule S for $\hat{\mathcal{J}}_{n,\lambda,\epsilon}$ which has at least $\lambda\epsilon n$ units of jobs from $A_{n,\lambda,\epsilon}$ remaining at time t_A, we have*

$$\frac{\textit{cost}(S, \hat{\mathcal{J}}_{n,\lambda,\epsilon})}{\texttt{OPT}(\hat{\mathcal{J}}_{n,\lambda,\epsilon})} > \left(1 + \frac{1}{4}\lambda\right).$$

Proof. Let $\lambda \in (0,1]$ and $\epsilon > 0$, and let S be a schedule for $\hat{\mathcal{J}}_{n,\lambda,\epsilon}$ which has at least $\lambda\epsilon n$ units of jobs from $A_{n,\lambda,\epsilon}$ remaining at time t_A.

Note that the cost of S for times $t \geq t_A$ is at least the cost of an optimal schedule for the remaining $\lambda\epsilon n$ units of jobs from $A_{n,\lambda,\epsilon}$ and the ϵn units of job from $B_{n,\lambda,\epsilon}$. By Lemma A.3, we thus get that:

$$\text{cost}(S, \hat{\mathcal{J}}_{n,\lambda,\epsilon}) \geq \text{cost}(S, A_{n,\lambda,\epsilon} \cup B_{n,\lambda,\epsilon}) \geq \frac{4}{3}((1+\lambda)\epsilon n)^{3/2}.$$

Now, by Lemma A.5, we get that when λ goes to 0 (independently of ϵ) and n goes to $+\infty$,

$$\text{cost}(S^*(\hat{\mathcal{J}}_{n,\lambda,\epsilon})) \leq \frac{4}{3}(\epsilon n)^{3/2}(1+\lambda+o(\lambda)+o(1)).$$

Hence,

$$\begin{aligned}\frac{\text{cost}(S, \hat{\mathcal{J}}_{n,\lambda,\epsilon})}{\text{cost}(S^*(\hat{\mathcal{J}}_{n,\lambda,\epsilon}))} &\geq \frac{\frac{4}{3}((1+\lambda)\epsilon n)^{3/2}}{\frac{4}{3}(\epsilon n)^{3/2}(1+\lambda+o(\lambda)+o(1))} \\ &= \left(1+\frac{3}{2}\lambda+o(\lambda)\right)\cdot(1-\lambda-o(\lambda)-o(1)) \\ &= 1+\frac{1}{2}\lambda-o(\lambda)-o(1).\end{aligned}$$

Hence, there is $\lambda' \in (0,1]$ and $M \in \mathbb{N}$ (note that $\lambda' \in (0,1]$ is independent of ϵ while M depends on it) such that if $\lambda \leq \lambda'$ and $n \geq M$, then

$$\frac{\text{cost}(S, \hat{\mathcal{J}}_{n,\lambda,\epsilon})}{\text{cost}(S^*(\hat{\mathcal{J}}_{n,\lambda,\epsilon}))} > \left(1+\frac{1}{4}\lambda\right).$$

□

Lemma A.7. *Let $\lambda \in (0,1], n \in \mathbb{N}$. Assume that S schedules at least $\frac{1}{3}\lambda\epsilon n$ units of jobs from $A_{n,\lambda,\epsilon}$ from time 0 to t_A. Then, there is a constant $C > 0$ such that*

$$\textit{cost}(S, \mathcal{J}_{n,\lambda,\epsilon}) \geq C(\epsilon n)^{3/2}\lambda\sqrt{1+\lambda}.$$

Proof. For convenience of exposition, assume that S schedules exactly $\frac{1}{3}\lambda\epsilon n$ units of jobs from $A_{n,\lambda,\epsilon}$ from time 0 to t (note that if S schedules more work from $A_{n,\lambda,\epsilon}$, then the cost can only be higher). By using [2], the optimal solution is to schedule each job i at speed $s_i = \sqrt{\frac{\epsilon\lambda n}{3} - i + c + 1}$, where c is the unique constant such that

$$\sum_{i=1}^{\frac{\epsilon\lambda n}{3}} \frac{1}{\sqrt{\frac{\epsilon\lambda n}{3} - i + c + 1}} = t_A.$$

To lower bound c, note that we then have

$$\begin{aligned}t_A &\geq \sum_{i=\frac{1}{2}\frac{\epsilon\lambda n}{3}}^{\frac{\epsilon\lambda n}{3}} \frac{1}{\sqrt{\frac{\epsilon\lambda n}{3} - \frac{1}{2}\frac{\epsilon\lambda n}{3} + c + 1}} \\ &= \frac{\epsilon\lambda n}{6}\frac{1}{\sqrt{\frac{\epsilon\lambda n}{6} + c + 1}}.\end{aligned}$$

By definition of t_A, we get

$$\begin{aligned}&\frac{4\lambda\epsilon n}{3\sqrt{\epsilon n(1+\frac{4}{3}\lambda)}} \geq \frac{\epsilon\lambda n}{6}\frac{1}{\sqrt{\frac{\epsilon\lambda n}{6} + c + 1}} \\ &\iff (\frac{4}{3}6)^2(\frac{\epsilon\lambda n}{6} + c + 1) \geq (1+\frac{4}{3}\lambda)\epsilon n \\ &\iff c \geq c_2\epsilon n - c_3\epsilon\lambda n - 1. \qquad \text{with } c_2 = \tfrac{1}{64}, c_3 = -\tfrac{1}{48} + \tfrac{1}{6}\end{aligned}$$

And the corresponding energy consumption is:

$$\begin{aligned}
&\sum_{i=1}^{\frac{\epsilon\lambda n}{3}} \sqrt{\frac{\epsilon\lambda n}{3} - i + c + 1} \\
&\geq \sum_{i=1}^{\frac{\epsilon\lambda n}{6}} \sqrt{\frac{\epsilon\lambda n}{3} - \frac{\epsilon\lambda n}{6} + c_2\epsilon n - c_3\epsilon\lambda n} \\
&\geq \frac{\epsilon\lambda n}{6} \sqrt{\frac{\epsilon\lambda n}{3} - \frac{\epsilon\lambda n}{6} + c_2\epsilon n + (\frac{1}{48} - \frac{1}{6})\epsilon\lambda n} \\
&= \frac{\epsilon\lambda n}{6} \sqrt{c_2\epsilon n + \frac{1}{48}\epsilon\lambda n} \\
&\geq C(\epsilon n)^{3/2}\lambda\sqrt{1+\lambda}. && \text{(for some constant } C > 0)
\end{aligned}$$

Therefore, any schedule S that completes $\lambda\epsilon n$ jobs before time t has cost lower bounded as:

$$\text{cost}(S, \mathcal{J}_{n,\lambda,\epsilon}) \geq C(\epsilon n)^{3/2}\lambda\sqrt{1+\lambda}.$$

□

We are now ready to present the proof of Lemma A.2.

Proof of Lemma A.2. By Lemma A.6, we have that there is a constant $\lambda' \in (0, 1]$ such that for all $\epsilon > 0$, there is $M \in \mathbb{N}$ such that for any algorithm $\mathcal{A}$ and $n \geq M$, and when running $\mathcal{A}$ with predictions $\hat{\mathcal{J}} = \hat{\mathcal{J}}_{n,\lambda,\epsilon}$ and realization $\mathcal{J} \in \{\hat{\mathcal{J}}_{n,\lambda,\epsilon}, \mathcal{J}_{n,\lambda,\epsilon}\}$, then either $\mathcal{A}$ schedules at least $\frac{1}{3}\lambda\epsilon n$ units of jobs of $A_{n,\lambda,\epsilon}$ before time t, or the schedule S output by $\mathcal{A}$ satisfies:

$$\frac{\text{cost}(S, \hat{\mathcal{J}}_{n,\lambda,\epsilon})}{\text{cost}(S^*(\hat{\mathcal{J}}_{n,\lambda,\epsilon}))} > \left(1 + \frac{1}{4}\lambda\right).$$

Hence, if $\mathcal{A}$ achieves a consistency of at most $\left(1 + \frac{1}{4}\lambda\right)$, $\mathcal{A}$ must schedule at least $\frac{1}{3}\lambda\epsilon n$ units of jobs of $A_{n,\lambda,\epsilon}$ before time t. However, we then have, by Lemma A.7, that for some constant $C > 0$,

$$\text{cost}_{\mathcal{A}}(\hat{\mathcal{J}} = \hat{\mathcal{J}}_{n,\lambda,\epsilon}, \mathcal{J} = \mathcal{J}_{n,\lambda,\epsilon}) \geq C(\epsilon n)^{3/2}\lambda\sqrt{1+\lambda}.$$

On the other hand, assuming that $\mathcal{J} = \mathcal{J}_{n,\lambda,\epsilon}$, we get by Lemma A.3 and Lemma A.4 that

$$\texttt{OPT}(\mathcal{J}_{n,\lambda,\epsilon}) \leq \frac{4}{3}(\lambda\epsilon n)^{3/2} + o((\epsilon n)^{3/2}) + 2n.$$

Hence, we get that for some constant $C'' > 0$ and n large enough,

$$\frac{\text{cost}_{\mathcal{A}}(\hat{\mathcal{J}} = \hat{\mathcal{J}}_{n,\lambda,\epsilon}, \mathcal{J} = \mathcal{J}_{n,\lambda,\epsilon})}{\texttt{OPT}(\mathcal{J}_{n,\lambda,\epsilon})} \geq \frac{C(\epsilon n)^{3/2}\lambda\sqrt{1+\lambda}}{\frac{4}{3}(\lambda\epsilon n)^{3/2} + o((\epsilon n)^{3/2}) + 2n} \geq C''\sqrt{\frac{1}{\lambda} + 1}.$$

□

B Missing analysis from Section 3

Lemma 3.1. *Let $\mathcal{J}_1$ be a set of jobs and S_1 be a feasible schedule for $\mathcal{J}_1$, let $\mathcal{J}_2$ be a set of jobs and S_2 be a feasible schedule for $\mathcal{J}_2$. Consider the schedule $S := S_1 + S_2$ for $\mathcal{J}_1 \cup \mathcal{J}_2$ which, at each time t, runs the machine at total speed $s(t) = s_1(t) + s_2(t)$ and processes each job $j \in \mathcal{J}_1$ at speed $s_{1,j}(t)$ and each job $j \in \mathcal{J}_2$ at speed $s_{2,j}(t)$. Then, cost$(S, \mathcal{J}_1 \cup \mathcal{J}_2) \leq \left(\text{cost}(S_1, \mathcal{J}_1)^{\frac{1}{\alpha}} + \text{cost}(S_2, \mathcal{J}_2)^{\frac{1}{\alpha}}\right)^{\alpha}$.*

Proof. We first upper bound the quality cost $F(S, \mathcal{J}_1 \cup \mathcal{J}_2)$ of the proposed schedule S. In each infinitesimal time interval $[t, t+dt]$ and for all $j \in \mathcal{J}_1$, S processes $s_{1,j}(t)dt$ units of work of job j, and for each $j \in \mathcal{J}_2$, S processes $s_{2,j}(t)dt$ units of work of job j. Hence S processes exactly the same amount of work for each job $j \in \mathcal{J}_1$ (resp. $j \in \mathcal{J}_2$) as S_1 (resp. S_2). We thus get that for all $t \geq 0$,

$$w_j^S(t) = w_j^{S_1}(t) \text{ for all } j \in \mathcal{J}_1 \qquad \text{and} \qquad w_j^S(t) = w_j^{S_2}(t) \text{ for all } j \in \mathcal{J}_2. \tag{1}$$

Therefore,

$$\begin{aligned} F(S, \mathcal{J}_1 \cup \mathcal{J}_2) &\leq F(S, \mathcal{J}_1) + F(S, \mathcal{J}_2) && (F \text{ is sub-additive}) \\ &= f\left(\{(W_j^S, j)\}_{j \in \mathcal{J}_1}\right) + f\left(\{(W_j^S, j)\}_{j \in \mathcal{J}_2}\right) \\ &= f\left(\{(W_j^{S_1}, j)\}_{j \in \mathcal{J}_1}\right) + f\left(\{(W_j^{S_2}, j)\}_{j \in \mathcal{J}_2}\right) && (\text{by } (1)) \\ &= F(S_1, \mathcal{J}_1) + F(S_2, \mathcal{J}_2). \end{aligned}$$

Next, we upper bound the energy consumption $E(S)$ of the proposed schedule S.

$$\begin{aligned} E(S) &= \int (s_1(t) + s_2(t))^\alpha \text{dt} \\ &= \sum_{i=0}^{\alpha} \binom{\alpha}{i} \int (s_1(t)^\alpha)^{\frac{i}{\alpha}} (s_2(t)^\alpha)^{\frac{\alpha-i}{\alpha}} \text{dt} \\ &\leq \sum_{i=0}^{\alpha} \binom{\alpha}{i} \left(\int (s_1(t))^\alpha \text{dt}\right)^{\frac{i}{\alpha}} \left(\int (s_2(t))^\alpha \text{dt}\right)^{\frac{\alpha-i}{\alpha}} \quad \text{(Hölder's inequality)} \\ &= E(S_1) + E(S_2) + \sum_{i=1}^{\alpha-1} \binom{\alpha}{i} E(S_1)^{\frac{i}{\alpha}} E(S_2)^{\frac{\alpha-i}{\alpha}} \\ &\leq E(S_1) + E(S_2) + \sum_{i=1}^{\alpha-1} \binom{\alpha}{i} \text{cost}(S_1)^{\frac{i}{\alpha}} \text{cost}(S_2)^{\frac{\alpha-i}{\alpha}}. \end{aligned}$$

Therefore, the total cost of schedule S can be upper bounded as follows:

$$\begin{aligned} \text{cost}(S, \mathcal{J}_1 \cup \mathcal{J}_2) &= F(S, \mathcal{J}_1 \cup \mathcal{J}_2) + E(S) \\ &\leq F(S_1, \mathcal{J}_1) + E(S_1) + F(S_2, \mathcal{J}_2) + E(S_2) + \sum_{i=1}^{\alpha-1} \binom{\alpha}{i} \text{cost}(S_1)^{\frac{i}{\alpha}} \text{cost}(S_2)^{\frac{\alpha-i}{\alpha}} \\ &= \left(\text{cost}(S_1, \mathcal{J}_1)^{\frac{1}{\alpha}} + \text{cost}(S_2, \mathcal{J}_2)^{\frac{1}{\alpha}}\right)^\alpha. \end{aligned}$$

□

Corollary 3.3. *$\texttt{OPT}(\mathcal{J} \cap \hat{\mathcal{J}}) \geq \left(1 - \eta_2^{\frac{1}{\alpha}}\right)^\alpha \texttt{OPT}(\hat{\mathcal{J}})$, and, assuming that* OFFLINEALG *is γ_{off}-competitive, we have: if $\texttt{OFF}(\mathcal{J}) \leq \lambda \texttt{OFF}(\hat{\mathcal{J}})$, then $\eta_2 \geq \left(1 - (\lambda\gamma_{off})^{\frac{1}{\alpha}}\right)^\alpha$.*

Proof. We prove the first part of the Corollary by contradiction. Assume that $\texttt{OPT}(\mathcal{J} \cap \hat{\mathcal{J}}) < \left(1 - \eta_2^{\frac{1}{\alpha}}\right)^\alpha \texttt{OPT}(\hat{\mathcal{J}})$. Next, by definition of the error η_2, we have $\texttt{OPT}(\hat{\mathcal{J}} \setminus \mathcal{J}) = \eta_2 \cdot \texttt{OPT}(\hat{\mathcal{J}})$. Hence, by Lemma 3.1, there exists a schedule S for $(\mathcal{J} \cap \hat{\mathcal{J}}) \cup (\hat{\mathcal{J}} \setminus \mathcal{J}) = \hat{\mathcal{J}}$ such that

$$\begin{aligned} \text{cost}(S, \hat{\mathcal{J}}) &\leq \left(\texttt{OPT}(\hat{\mathcal{J}} \setminus \mathcal{J})^{\frac{1}{\alpha}} + \texttt{OPT}(\mathcal{J} \cap \hat{\mathcal{J}})^{\frac{1}{\alpha}}\right)^\alpha \\ &< \left((\eta_2 \texttt{OPT}(\hat{\mathcal{J}}))^{\frac{1}{\alpha}} + \left(\left(1 - \eta_2^{\frac{1}{\alpha}}\right)^\alpha \texttt{OPT}(\hat{\mathcal{J}})\right)^{\frac{1}{\alpha}}\right)^\alpha \\ &= \texttt{OPT}(\hat{\mathcal{J}}), \end{aligned}$$

which contradicts the definition of $\texttt{OPT}(\hat{\mathcal{J}})$ and ends the proof of the first result.

We now show the second part of the Corollary.

Assume that $\texttt{OFF}(\mathcal{J}) \leq \lambda\texttt{OFF}(\hat{\mathcal{J}})$. Then, since OFFLINEALG is γ_{off}-competitive, we have

$$\texttt{OPT}(\mathcal{J}) \leq \texttt{OFF}(\mathcal{J}) \leq \lambda\texttt{OFF}(\hat{\mathcal{J}}) \leq \lambda\gamma_{\text{off}}\texttt{OPT}(\hat{\mathcal{J}}).$$

In particular, $\texttt{OPT}(\mathcal{J} \cap \hat{\mathcal{J}}) \leq \lambda\gamma_{\text{off}}\texttt{OPT}(\hat{\mathcal{J}})$. Next, assume by contradiction, that $\eta_2 < \left(1 - (\lambda\gamma_{\text{off}})^{\frac{1}{\alpha}}\right)^{\alpha}$, which implies that $\texttt{OPT}(\hat{\mathcal{J}} \setminus \mathcal{J}) < \left(1 - (\lambda\gamma_{\text{off}})^{\frac{1}{\alpha}}\right)^{\alpha} \texttt{OPT}(\hat{\mathcal{J}})$. Then, by Lemma 3.1, there exists a schedule S for $(\mathcal{J} \cap \hat{\mathcal{J}}) \cup (\hat{\mathcal{J}} \setminus \mathcal{J}) = \hat{\mathcal{J}}$ such that

$$\begin{aligned}
\text{cost}(S, \hat{\mathcal{J}}) &\leq \left(\texttt{OPT}(\hat{\mathcal{J}} \setminus \mathcal{J})^{\frac{1}{\alpha}} + \texttt{OPT}(\mathcal{J} \cap \hat{\mathcal{J}})^{\frac{1}{\alpha}}\right)^{\alpha} \\
&< \left(((\lambda\gamma_{\text{off}})\texttt{OPT}(\hat{\mathcal{J}}))^{\frac{1}{\alpha}} + \left(\left(1 - (\lambda\gamma_{\text{off}})^{\frac{1}{\alpha}}\right)^{\alpha} \texttt{OPT}(\hat{\mathcal{J}})\right)^{\frac{1}{\alpha}}\right)^{\alpha} \\
&= \texttt{OPT}(\hat{\mathcal{J}}),
\end{aligned}$$

which contradicts the definition of $\texttt{OPT}(\hat{\mathcal{J}})$. Hence, $\eta_2 \geq \left(1 - (\lambda\gamma_{\text{off}})^{\frac{1}{\alpha}}\right)^{\alpha}$. □

Theorem 3.4. *For any $\lambda \in (0, 1]$,* TPE *with a γ_{on}-competitive algorithm* ONLINEALG *and a γ_{off}-competitive offline algorithm* OFFLINEALG *achieves a competitive ratio of*

$$\begin{cases} \gamma_{on} & \text{if } \texttt{OFF}(\mathcal{J}) \leq \lambda\texttt{OFF}(\hat{\mathcal{J}}) \\ \dfrac{\left(\gamma_{off}^{\frac{1}{\alpha}} + \gamma_{on}^{\frac{1}{\alpha}}((\lambda\gamma_{off})^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}})\right)^{\alpha}}{\max\left\{\frac{\lambda}{\gamma_{off}}, \eta_1 + \left(1 - \eta_2^{\frac{1}{\alpha}}\right)^{\alpha}\right\}} & \text{otherwise.} \end{cases}$$

Proof. First, assume that that for all $t \geq 0$, $\texttt{OFF}(\mathcal{J}_{\leq t}) \leq \lambda\texttt{OFF}(\hat{\mathcal{J}})$ (i.e., TPE never goes through lines 6-10). Then, the schedule S returned by the algorithm is obtained by running the γ_{on}-competitive algorithm ONLINEALG on $\mathcal{J}$, hence

$$\text{cost}(S, \mathcal{J}) \leq \gamma_{\text{on}} \cdot \texttt{OPT}(\mathcal{J}). \tag{2}$$

Next, assume that there is $t_\lambda \geq 0$ such that $\texttt{OFF}(\mathcal{J}_{\leq t_\lambda}) > \lambda\texttt{OFF}(\hat{\mathcal{J}})$. Since OFFLINEALG is γ_{off}-competitive, we immediately get:

$$\texttt{OPT}(\mathcal{J}) \geq \texttt{OPT}(\mathcal{J}_{\leq t_\lambda}) \geq \frac{\texttt{OFF}(\mathcal{J}_{\leq t_\lambda})}{\gamma_{\text{off}}} > \frac{\lambda}{\gamma_{\text{off}}}\texttt{OFF}(\hat{\mathcal{J}}) \geq \frac{\lambda}{\gamma_{\text{off}}}\texttt{OPT}(\hat{\mathcal{J}}).$$

By Corollary 3.3 and by definition of the error η_1, we also get the following lower bound on the optimal schedule:

$$\texttt{OPT}(\mathcal{J}) \geq \texttt{OPT}(\mathcal{J} \setminus \hat{\mathcal{J}}) + \texttt{OPT}(\mathcal{J} \cap \hat{\mathcal{J}}) \geq \eta_1\texttt{OPT}(\hat{\mathcal{J}}) + \left(1 - \eta_2^{\frac{1}{\alpha}}\right)^{\alpha} \texttt{OPT}(\hat{\mathcal{J}}).$$

Therefore,

$$\texttt{OPT}(\mathcal{J}) \geq \max\left\{\frac{\lambda}{\gamma_{\text{off}}}, \eta_1 + \left(1 - \eta_2^{\frac{1}{\alpha}}\right)^{\alpha}\right\} \texttt{OPT}(\hat{\mathcal{J}}). \tag{3}$$

Now, by Lemma 3.2, the cost of the schedule S output by TPE is always upper bounded as follows:

$$\text{cost}(S, \mathcal{J}) \leq \texttt{OPT}(\hat{\mathcal{J}}) \left(\gamma_{\text{off}}^{\frac{1}{\alpha}} + \gamma_{\text{on}}{}^{\frac{1}{\alpha}}((\lambda\gamma_{\text{off}})^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}})\right)^{\alpha}. \tag{4}$$

Hence, we get the following upper bound on the competitive ratio of TPE:

$$\begin{aligned}
&\frac{\text{cost}(S,\mathcal{J})}{\texttt{OPT}(\mathcal{J})} \\
&= \mathbf{1}_{\texttt{OFF}(\mathcal{J})\leq\lambda\texttt{OFF}(\hat{\mathcal{J}})}\frac{\text{cost}(S,\mathcal{J})}{\texttt{OPT}(\mathcal{J})} + \mathbf{1}_{\texttt{OFF}(\mathcal{J})>\lambda\texttt{OFF}(\hat{\mathcal{J}})}\frac{\text{cost}(S,\mathcal{J})}{\texttt{OPT}(\mathcal{J})} \\
&\leq \mathbf{1}_{\texttt{OFF}(\mathcal{J})\leq\lambda\texttt{OFF}(\hat{\mathcal{J}})}\gamma_{\text{on}} + \mathbf{1}_{\texttt{OFF}(\mathcal{J})>\lambda\texttt{OFF}(\hat{\mathcal{J}})}\frac{\left(\gamma_{\text{off}}^{\frac{1}{\alpha}} + \gamma_{\text{on}}^{\frac{1}{\alpha}}((\lambda\gamma_{\text{off}})^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}})\right)^{\alpha}}{\max\left\{\frac{\lambda}{\gamma_{\text{off}}}, \eta_1 + \left(1-\eta_2^{\frac{1}{\alpha}}\right)^{\alpha}\right\}}. \qquad \text{by (2),(3),(4)}
\end{aligned}$$

□

Corollary 3.5. *For any $\lambda \in (0,1)$,* TPE *with a γ_{on}-competitive algorithm* ONLINEALG *and an optimal offline algorithm* OFFLINEALG *is $1+\gamma_{on}2^{\alpha}\lambda^{\frac{1}{\alpha}}$ competitive if $\eta_1 = \eta_2 = 0$ (consistency) and $\max\{\gamma_{on}, \frac{1+\gamma_{on}2^{2\alpha}\lambda^{\frac{1}{\alpha}}}{\lambda}\}$-competitive for all η_1, η_2 (robustness). In particular, for any constant $\epsilon > 0$, with $\lambda = (\frac{\epsilon}{\gamma_{on}2^{\alpha}})^{\alpha}$,* TPE *is $1+\epsilon$-consistent and $O(1)$-robust.*

Proof. We start by the consistency. Since we assumed that OFFLINEALG is optimal, by Corollary 3.3, we have that for all $\lambda \in (0,1)$, $\texttt{OFF}(\mathcal{J}) > \lambda\texttt{OFF}(\hat{\mathcal{J}})$, when $\eta_2 = 0$. The result follows by an immediate upper bound on the competitive ratio in this case.

We now show the robustness. First note that if $\texttt{OFF}(\mathcal{J}) \leq \lambda\texttt{OFF}(\hat{\mathcal{J}})$, then $\frac{\text{cost}(S,\mathcal{J})}{\texttt{OPT}(\mathcal{J})} \leq \gamma_{\text{on}} \leq \max\{\gamma_{\text{on}}, \frac{1+\gamma_{\text{on}}2^{2\alpha}\lambda^{\frac{1}{\alpha}}}{\lambda}\}$ for any $\lambda \in [0,1]$. Now, assume that $\texttt{OFF}(\mathcal{J}) > \lambda\texttt{OFF}(\hat{\mathcal{J}})$. We then have

$$\frac{\text{cost}(S,\mathcal{J})}{\texttt{OPT}(\mathcal{J})} \leq \frac{\left(1+\gamma_{\text{on}}^{\frac{1}{\alpha}}(\lambda^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}})\right)^{\alpha}}{\max\left\{\lambda, \eta_1 + \left(1-\eta_2^{\frac{1}{\alpha}}\right)^{\alpha}\right\}} \leq \frac{\left(1+\gamma_{\text{on}}^{\frac{1}{\alpha}}(\lambda^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}})\right)^{\alpha}}{\max\{\lambda, \eta_1\}}.$$

If $\eta_1 \leq \lambda$, we get:

$$\frac{\text{cost}(S,\mathcal{J})}{\texttt{OPT}(\mathcal{J})} \leq \frac{\left(1+\gamma_{\text{on}}^{\frac{1}{\alpha}}(2\lambda^{\frac{1}{\alpha}})\right)^{\alpha}}{\lambda} \leq \frac{1+\gamma_{\text{on}}2^{2\alpha}\lambda^{\frac{1}{\alpha}}}{\lambda},$$

and if $\eta_1 \geq \lambda$, we get:

$$\frac{\text{cost}(S,\mathcal{J})}{\texttt{OPT}(\mathcal{J})} \leq \frac{\left(1+\gamma_{\text{on}}^{\frac{1}{\alpha}}(2\eta_1^{\frac{1}{\alpha}})\right)^{\alpha}}{\eta_1} \leq \frac{1+\gamma_{\text{on}}2^{2\alpha}\max\{\eta_1, \eta_1^{\frac{1}{\alpha}}\}}{\eta_1}.$$

Since the above function reaches its maximum value over $[\lambda, +\infty[$ for $\eta_1 = \lambda$, this immediately yields the result. □

C The Extension with Job Shifts (Full version)

Note that in the definition of the prediction error η, a job j is considered to be correctly predicted only if $r_j = \hat{r}_j$ and $p_j = \hat{p}_j$. In this section, we consider an extension where a job is considered to be correctly predicted even if the release time and processing time are shifted by a small amount. In this extension, we also allow each job to have some weight $v_j > 0$, that can be shifted as well. We propose and analyze an algorithm that generalizes the algorithm from the previous section.

Motivating example. Consider the objective of minimizing energy plus flow time with $\alpha = 2$. Let $(\hat{\mathcal{J}}, \mathcal{J})$ be an instance where $\mathcal{J}$ has n jobs with weight $w = 1.01$ and processing time $p = 0.99$, all released at time $r = 0.1$, and $\hat{\mathcal{J}}$ has n jobs with weight $w = 1$ and processing time $p = 1$, all released at time $r = 0$. Since $\hat{\mathcal{J}} \setminus \mathcal{J} = \hat{\mathcal{J}}$, we have here that $\eta_1 = \texttt{OPT}(\hat{\mathcal{J}} \setminus \mathcal{J}) = \texttt{OPT}(\hat{\mathcal{J}}) = \Omega(n^{3/2})$ (by Lemma A.3), whereas it seems reasonable to say that $\hat{\mathcal{J}}$ was a 'good' prediction for instance $\mathcal{J}$, since it accurately represents the pattern of the jobs in $\mathcal{J}$.

In this section, we assume that the quality of cost function F is such that the total cost function $E+F$ satisfies a smoothness condition, which we next define.

Smooth objective function. Let $\mathbb{J}$ denote the collection of all sets of jobs. We say that a function $\beta : \mathbb{J} \longrightarrow \mathbb{R}$ is smooth if for all $\mathcal{J} \in \mathbb{J}$, $\{r'_j\}_{j\in\mathcal{J}} \geq 0$ and $\{\eta_j\}_{j\in\mathcal{J}} \geq 0$, we have $\beta(\mathcal{J}') \leq (1+\max_j \eta_j)\beta(\mathcal{J})$, where $\mathcal{J}' = \{(j, r'_j, p_j(1+\eta_j), v_j(1+\eta_j)\}$. β is monotone if for all $\mathcal{J}'' \subseteq \mathcal{J}$, we have $\beta(\mathcal{J}'') \leq \beta(\mathcal{J})$.

We say that a cost function cost(.,.) is β-smooth if there is a smooth monotone function $\beta(.) \geq 1$ such that for all $\eta, \eta' \in [0,1]$, $\mathcal{J}_1, \mathcal{J}_2$ with $|\mathcal{J}_1| = |\mathcal{J}_2|$ and bijection $\pi : \mathcal{J}_1 \longrightarrow \mathcal{J}_2$, and for all S_1 and S_2 feasible for $\mathcal{J}_1$ and $\mathcal{J}_2$:

- **(smoothness of optimal cost).** If for all $j \in \mathcal{J}_1$, $|r_j - r_{\pi(j)}| \leq \eta'$, $p_j \leq p_{\pi(j)}(1+\eta)$ and $v_j \leq v_{\pi(j)}(1+\eta)$, then
$$\texttt{OPT}(\mathcal{J}_1) \leq (1+\beta(\mathcal{J}_1)\eta)\texttt{OPT}(\mathcal{J}_2) + \beta(\mathcal{J}_1)|\mathcal{J}_1|\eta'.$$
- **(shifted work profile for dominated schedule.)** If for all $j \in \mathcal{J}$, $p_j \leq p_{\pi(j)}$, $v_j \leq v_{\pi(j)}$, $r_j \geq r_{\pi(j)} - \eta'$ and for all $t \geq r_{\pi(j)} + \eta'$, $w^j_{S_1}(t) \leq w^{\pi(j)}_{S_2}(t-\eta')$, where $w^j_{S_1}(t)$ (resp. $w^j_{S_2}(t)$) denotes the remaining amount of work for j a time t for S_1 (resp. S_2), then
$$F(S_1, \mathcal{J}_1) \leq F(S_2, \mathcal{J}_2) + \beta(\mathcal{J}_1)|\mathcal{J}_1|\eta'.$$

In other words, if $\mathcal{J}_1$ and $\mathcal{J}_2$ are close to each other, then the optimal costs for $\mathcal{J}_1$ and $\mathcal{J}_2$ are close, and if schedules S_1, S_2 induce similar but slightly shifted work profiles for $\mathcal{J}_1$ and $\mathcal{J}_2$, then the quality costs for S_1 and S_2 are close.

We show in Appendix D that for the classically studied energy plus weighted flow time minimization problem with $\alpha \geq 1$, the cost function is $\max(4\max_j v_j, 2^\alpha - 1)$-smooth. Note that for energy minimization with deadlines, the objective introduced in Section 3.3 is not smooth for any bounded function $\beta(.)$, since a small shift in the work profiles can induce a large increase in the objective function (in the case we miss a job's hard deadline). However, [7] (Section F.2) shows that it is also possible to transform any prediction-augmented algorithm for the energy plus deadline problem into a shift-tolerant algorithm.

Shift tolerance and error definition. In this extension, we allow each job in the prediction to be perturbed by a small amount. Past this tolerance threshold, the perturbed job is treated as a distinct job. We assume here that when a job arrives, it is always possible to identify which job of the prediction (if any) it corresponds to. More specifically, for each job j, we write (j, r_j, p_j, v_j) for the real values of the parameters associated with j and $(j, \hat{r}_j, \hat{p}_j, \hat{v}_j)$ for their predicted values (with the convention that $(j, r_j, p_j, v_j) = \emptyset$ if the job didn't arrive and $(j, \hat{r}_j, \hat{p}_j, \hat{v}_j) = \emptyset$ if the job was not predicted).

Next, we let $\eta^{\text{shift}} \in [0,1)$ be a shift tolerance parameter, that is initially set by the decision-maker, and we assume that the objective function is β-smooth for some smooth monotone function $\beta(.) \geq 1$. We now define the set of 'correctly predicted' jobs as $\mathcal{J}^{\text{shift}} = \{(j, r_j, p_j, v_j) : |r_j - \hat{r}_j| \leq \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})} \cdot \frac{\texttt{OPT}(\hat{\mathcal{J}})}{|\hat{\mathcal{J}}|}, |p_j - \hat{p}_j| \leq \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})} \cdot \hat{p}_j, |v_j - \hat{v}_j| \leq \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})} \cdot \hat{v}_j\}$, which is the set of jobs whose release time, weights and processing times have only been slightly shifted as compared to their predicted values. The amount of shift we tolerate depends on the smoothness function $\beta(.)$ of the objective function F and on the predicted instance $\hat{\mathcal{J}}$. In addition, note that the allowed shift in release time is proportional to the average cost per job (the intuition here is that for most objective functions, the average cost per job is at least the average completion time per job). We underscore the fact that $\mathcal{J} \setminus \mathcal{J}^{\text{shift}}$ contains both the predicted jobs that have past the shift tolerance and additional jobs in the realization. Finally, we let $\hat{\mathcal{J}}^{\text{shift}} = \{(j, \hat{r}_j, \hat{p}_j, \hat{v}_j) : (j, r_j, p_j, v_j) \in \mathcal{J}^{\text{shift}}\}$. The error $\eta^g = \frac{1}{\texttt{OPT}(\hat{\mathcal{J}})} \cdot \max\{\texttt{OPT}(\mathcal{J} \setminus \mathcal{J}^{\text{shift}}), \texttt{OPT}(\hat{\mathcal{J}} \setminus \hat{\mathcal{J}}^{\text{shift}})\}$ is now defined as the optimal cost for both the additional and missing jobs (similarly as in the previous sections) and the jobs that have past the shift tolerance, normalized by the optimal cost for the prediction.

Algorithm description. The Algorithm, called TPE-S and formally described in Algorithm 2, takes the same input parameters as Algorithm TPE, with some additional shift tolerance parameter $\eta^{\text{shift}} \in [0,1)$, from which we compute the maximum allowed shift in release time $\bar{\eta}$ (Line 8).

TPE-S globally follows the structure of TPE, with a few differences, that we now detail. First, we start by slightly increasing the predicted weight and processing time of each job to obtain the set of jobs $\hat{\mathcal{J}}^{\text{up}} := \{(j, \hat{r}_j, \hat{p}_j(1 + \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}), \hat{v}_j(1 + \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}))\}$ (Line 1). Note that by the first smoothness condition, the optimal schedule for $\hat{\mathcal{J}}^{\text{up}}$ has only a slightly higher cost than $\texttt{OPT}(\hat{\mathcal{J}})$.

Then, similarly as TPE, TPE-S starts with a first phase where it follows the online algorithm ONLINEALG until the time t_λ where the optimal offline has reached some threshold value $\lambda\texttt{OPT}(\hat{\mathcal{J}}^{\text{up}})$ (Lines 3-7). In the second phase (Lines 9-13), it again combines two schedules, this time, for (1) the jobs in $\mathcal{J}^{\text{shift}}_{\geq t_\lambda}$ that are within the shift tolerance (2) the jobs in $\mathcal{J} \setminus \mathcal{J}^{\text{shift}}_{\geq t_\lambda}$, which include the remaining jobs from phase 1 and the non-predicted jobs (or jobs that have past the shift-tolerance) that are released after time t_λ. To schedule the jobs in $\mathcal{J}^{\text{shift}}_{\geq t_\lambda}$, we first compute an offline schedule $\hat{S}$ for $\hat{\mathcal{J}}^{\text{up}}$ (Line 9). One small difference with TPE is that we will delay the schedule $\hat{S}$ by $\bar{\eta}$ time steps backwards when we schedule jobs in $\mathcal{J}^{\text{shift}}_{\geq t_\lambda}$. More precisely, each job $(j, r_j, p_j, v_j) \in \mathcal{J}^{\text{shift}}_{\geq t_\lambda}$ is scheduled on the same way the job with the same identifier j in $\hat{\mathcal{J}}^{\text{up}}$ is scheduled by $\hat{S}$ at time $t - \bar{\eta}$ (Line 12). The intuition here is that we need to wait a small delay of $\bar{\eta}$ in order to identify which jobs of the predictions indeed arrived. Finally, and similarly as TPE, the speeds for jobs in $\mathcal{J} \setminus \mathcal{J}^{\text{shift}}_{\geq t_\lambda}$ are set by running ONLINEALG on the set $\mathcal{J} \setminus \mathcal{J}^{\text{shift}}_{\geq t_\lambda}$ (Line 13).

Algorithm 2 Two-Phase Energy Efficient Scheduling with Shift Tolerance (TPE-S)

Input: predicted and true sets of jobs $\hat{\mathcal{J}}$ and $\mathcal{J}$, quality of cost function F, offline and online algorithms (without predictions) OFFLINEALG and ONLINEALG for problem F, confidence level $\lambda \in (0, 1]$, shift tolerance $\eta^{\text{shift}} > 0$.

1: $\hat{\mathcal{J}}^{\text{up}} \leftarrow \{(j, \hat{r}_j, \hat{p}_j(1 + \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}), \hat{v}_j(1 + \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}))\}$
2: $\hat{\texttt{OPT}} \leftarrow \texttt{OPT}(\hat{\mathcal{J}}^{\text{up}})$
3: **for** $t \geq 0$ **do**
4: **if** $\texttt{OPT}(\mathcal{J}_{\leq t}) > \lambda \cdot \hat{\texttt{OPT}}$ **then**
5: $t_\lambda \leftarrow t$
6: **break**
7: $\{s_j(t)\}_{j \in \mathcal{J}_{\leq t}} \leftarrow \text{ONLINEALG}(\mathcal{J}_{\leq t})(t)$
8: $\bar{\eta} \leftarrow \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})} \cdot \frac{\texttt{OPT}(\hat{\mathcal{J}})}{|\hat{\mathcal{J}}|}$
9: $\{\hat{s}_j(t)\}_{t \geq 0, j \in \hat{\mathcal{J}}^{\text{up}}} \leftarrow \text{OFFLINEALG}(\hat{\mathcal{J}}^{\text{up}})$
10: **for** $t \geq t_\lambda$ **do**
11: **for** $j : (j, r_j, p_j, v_j) \in \mathcal{J}^{\text{shift}}_{\geq t_\lambda}$ **do**
12: $s_j(t) \leftarrow \hat{s}_j(t - \bar{\eta})$
13: $\{s_j(t)\}_{j \in \mathcal{J}_{\leq t} \setminus \mathcal{J}^{\text{shift}}_{\geq t_\lambda}} \leftarrow \text{ONLINEALG}(\mathcal{J}_{\leq t} \setminus \mathcal{J}^{\text{shift}}_{\geq t_\lambda})(t)$
14: **return** $\{s_j(t)\}_{t \geq 0, j \in \mathcal{J}}$

Analysis. We now present the analysis of TPE-S. All missing proofs are provided in Appendix D. In the following lemma, we start by upper bounding the cost of the schedule output by TPE-S for the jobs that were released in the second phase and were correctly predicted (i.e., within the shift tolerance). The proof mainly exploits the two smoothness conditions of the cost function.

Lemma C.1. *Assume that cost$(.,.)$ is β-smooth. Consider the schedule S^{shift}, which, for all $t \geq t_\lambda$ and $(j, r_j, p_j, v_j) \in \mathcal{J}^{\text{shift}}_{\geq t_\lambda}$, processes job j at speed*

$$s_j(t) = \hat{s}_j\left(t - \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})} \cdot \frac{\texttt{OPT}(\hat{\mathcal{J}})}{|\hat{\mathcal{J}}|}\right).$$

Then,

$$\text{cost}(S^{\text{shift}}, \mathcal{J}^{\text{shift}}_{\geq t_\lambda}) \leq (1 + 2\eta^{\text{shift}}(1 + \eta^{\text{shift}}))\texttt{OPT}(\hat{\mathcal{J}}).$$

We now show some slightly modified version of Corollary 3.3. Similarly as before, we write $\eta_1 = \frac{\texttt{OPT}(\mathcal{J} \setminus \mathcal{J}^{\text{shift}})}{\texttt{OPT}(\hat{\mathcal{J}})}$ to denote the error corresponding to additional jobs in the prediction, and $\eta_2 = \frac{\texttt{OPT}(\hat{\mathcal{J}} \setminus \hat{\mathcal{J}}^{\text{shift}})}{\texttt{OPT}(\hat{\mathcal{J}})}$ for the error corresponding to missing jobs.

Corollary C.2. *Assume that cost$(.,.)$ is β-smooth, then*

$$\texttt{OPT}(\mathcal{J}^{\textit{shift}}) \geq \left[\left(1 - \eta_2^{\frac{1}{\alpha}}\right)^{\alpha} - \eta^{\textit{shift}} \right] \Big/ (1 + \eta^{\textit{shift}}) \texttt{OPT}(\hat{\mathcal{J}}).$$

Corollary C.3. *Assume that cost$(.,.)$ is β-smooth. If $\texttt{OPT}(\mathcal{J}) \leq \lambda \texttt{OPT}(\hat{\mathcal{J}})$, then*

$$\eta_2 \geq \left(1 - (\lambda(1 + \eta^{\textit{shift}}) + \eta^{\textit{shift}})^{\frac{1}{\alpha}}\right)^{\alpha}.$$

We now state the main result of this section, which is our upper bound on the competitive ratio of the shift-tolerant Algorithm TPE-S.

Theorem C.4. *Assume that cost$(.,.)$ is β-smooth. Then, for any $\lambda \in (0, 1], \eta^{\textit{shift}} \in [0, 1)$, the competitive ratio of* TPE-S *run with trust parameter λ, a γ-competitive algorithm* ONLINEALG*, an optimal offline algorithm* OFFLINEALG*, and shift tolerance $\eta^{\textit{shift}}$ is at most*

$$\begin{cases} \gamma & \text{if } \texttt{OPT}(\mathcal{J}) \leq \lambda \texttt{OPT}(\hat{\mathcal{J}}) \\ \dfrac{\left((1+2\eta^{\textit{shift}}(1+\eta^{\textit{shift}}))^{\frac{1}{\alpha}} + \gamma^{\frac{1}{\alpha}}\left(\lambda^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}}\right)\right)^{\alpha}}{\max\left\{\lambda, \eta_1 + \left(\left(1 - \eta_2^{\frac{1}{\alpha}}\right)^{\alpha} - \eta^{\textit{shift}}\right)(1+\eta^{\textit{shift}})^{-1}\right\}} & \textit{otherwise.} \end{cases}$$

In particular, we deduce the following consistency and robustness guarantees.

Corollary C.5. *(consistency) For any $\lambda \in (0, 1]$ and $\eta^{\textit{shift}} \in [0, 1)$, if $\eta_1 = \eta_2 = 0$ (all jobs are within the shift tolerance and there is no extra or missing jobs), then the competitive ratio of* TPE-S *run with trust parameter λ and shift tolerance parameter $\eta^{\textit{shift}}$ is upper bounded by* $\min(\frac{1}{\lambda}, \frac{(1+\eta^{\textit{shift}})}{(1-\eta^{\textit{shift}})}) \cdot \left((1 + 2\eta^{\textit{shift}}(1 + \eta^{\textit{shift}}))^{\frac{1}{\alpha}} + \gamma^{\frac{1}{\alpha}}\lambda^{\frac{1}{\alpha}}\right)^{\alpha} \leq \frac{(1+2\eta^{\textit{shift}}(1+\eta^{\textit{shift}}))^2}{(1-\eta^{\textit{shift}})} \cdot (1 + \gamma 2^{\alpha}\lambda^{\frac{1}{\alpha}})$.

Corollary C.6. *(robustness) For any $\lambda \in (0, 1]$ and $\eta^{\textit{shift}} \in [0, 1)$, the competitive ratio of Algorithm 1 run with trust parameter λ and shift tolerance parameter $\eta^{\textit{shift}}$ is upper bounded by* $\frac{(1+2\eta^{\textit{shift}}(1+\eta^{\textit{shift}}))(1+\gamma 2^{2\alpha}\lambda^{\frac{1}{\alpha}})}{\lambda}$.

D Missing analysis from Appendix C

Lemma D.1. *For the objective of minimizing total integral weighted flow time plus energy with $\alpha \geq 1$, the cost function is* $\max(4 \cdot \max_j v_j, 2^{\alpha} - 1)$*-smooth.*

Proof. Let $\mathcal{J}_1, \mathcal{J}_2$ with $|\mathcal{J}_1| = |\mathcal{J}_2|$, a bijection $\pi : \mathcal{J}_1 \longrightarrow \mathcal{J}_2$ and S_1 and S_2 feasible for $\mathcal{J}_1$ and $\mathcal{J}_2$.

We start with the first smoothness condition. Assume that for some $\eta, \eta' \in [0, 1]$ and for all $j \in \mathcal{J}_1$, $|r_j - r_{\pi(j)}| \leq \eta'$, $p_j \leq p_{\pi(j)}(1 + \eta)$ and $v_j \leq v_{\pi(j)}(1 + \eta)$. Let S^* be an optimal schedule for $\mathcal{J}_2$ and consider the schedule $S = \{s_j(t) := (1 + \eta) \cdot s^*_{\pi(j)}(t - \eta')\}_{j \in \mathcal{J}_1}$ for $\mathcal{J}_1$.

Note that for all $j \in \mathcal{J}_1$ and $t \geq 0$, $s^*_{\pi(j)}(t - \eta') > 0$ only if $t - \eta' > r_{\pi(j)}$. Since we assumed $|r_j - r_{\pi(j)}| \leq \eta'$, we get that $s_j(t) > 0$ only if $t \geq r_j$, hence S is feasible for $\mathcal{J}_1$. Next, note that

$$E(S) = E(S^*)(1 + \eta)^{\alpha} \leq E(S^*)(1 + (2^{\alpha} - 1)\eta).$$

Recall that c_j^S denotes the completion time of j by S. Since $p_j \leq (1+\eta)p_{\pi(j)}$, and since we assumed that $|r_j - r_{\pi(j)}| \leq \eta'$, we have, by definition of S, that for all j, $c_j^S \leq \eta' + c^{S^*}_{\pi(j)}$.

Hence,

$$\begin{aligned}
F(S, \mathcal{J}_1) &= \sum_{j \in \mathcal{J}_1} v_j (c_j^S - r_j) \\
&\leq \sum_{j \in \mathcal{J}_1} v_{\pi(j)} (1+\eta)(c_{\pi(j)}^{S^*} + \eta' - (r_{\pi(j)} - \eta')) \\
&= (1+\eta) F(S^*, \mathcal{J}_2) + 2 \max_j v_j |\mathcal{J}_1| (1+\eta)\eta' \\
&\leq (1+\eta) F(S^*, \mathcal{J}_2) + 4 \max_j v_j |\mathcal{J}_1| \eta' && \eta \in [0,1] \\
&\leq (1 + (2^\alpha - 1)\eta) F(S^*, \mathcal{J}_2) + 4 \max_j v_j |\mathcal{J}_1| \eta' && \alpha \geq 1.
\end{aligned}$$

Therefore,

$$\begin{aligned}
\texttt{OPT}(\mathcal{J}_1) &\leq \text{cost}(S, \mathcal{J}_1) \\
&= E(S) + F(S, \mathcal{J}_1) \\
&\leq (E(S^*) + F(S^*, \mathcal{J}_2)) \cdot (1 + (2^\alpha - 1)\eta) + 4 \max_j v_j |\mathcal{J}_1| \eta' \\
&= \text{cost}(S^*, \mathcal{J}_2)(1 + (2^\alpha - 1)\eta) + 4 \max_j v_j |\mathcal{J}_1| \eta' \\
&= \texttt{OPT}(\mathcal{J}_2)(1 + (2^\alpha - 1)\eta) + 4 \max_j v_j |\mathcal{J}_1| \eta'.
\end{aligned}$$

We now show the second smoothness condition. Assume that for all $j \in \mathcal{J}_1$, $p_j \leq p_{\pi(j)}$, $v_j \leq v_{\pi(j)}$, $r_j \geq r_{\pi(j)} - \eta'$ and that for all $t \geq r_{\pi(j)} + \eta'$, $w_{S_1}^j(t) \leq w_{S_2}^{\pi(j)}(t - \eta')$. Then, in particular $w_{S_1}^j(c_{\pi(j)}^{S_2} + \eta') \leq w_{S_2}^{\pi(j)}(c_{\pi(j)}^{S_2}) = 0$, hence $c_j^{S_1} = \min\{t \geq r_j : w_{S_1}^j(t) = 0\} \leq c_{\pi(j)}^{S_2} + \eta'$. By a similar argument as above, we conclude that:

$$F(S, \mathcal{J}_1) \leq F(S_2, \mathcal{J}_2) + 4 \max_j v_j |\mathcal{J}_1| \eta'.$$

□

Lemma C.1. *Assume that cost(.,.) is β-smooth. Consider the schedule S^{shift}, which, for all $t \geq t_\lambda$ and $(j, r_j, p_j, v_j) \in \mathcal{J}_{\geq t_\lambda}^{shift}$, processes job j at speed*

$$s_j(t) = \hat{s}_j \left(t - \frac{\eta^{shift}}{\beta(\hat{\mathcal{J}})} \cdot \frac{\texttt{OPT}(\hat{\mathcal{J}})}{|\hat{\mathcal{J}}|} \right).$$

Then,

$$\text{cost}(S^{shift}, \mathcal{J}_{\geq t_\lambda}^{shift}) \leq (1 + 2\eta^{shift}(1 + \eta^{shift}))\texttt{OPT}(\hat{\mathcal{J}}).$$

Proof. For simplifying the exposition, in the remainder of the proof, we write $\bar{\eta}$ instead of $\frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})} \cdot \frac{\texttt{OPT}(\hat{\mathcal{J}})}{|\hat{\mathcal{J}}|}$.

We first analyse the energy cost.

$$\begin{aligned}
E(S^{\text{shift}}) = \int_{t \geq t_\lambda} \left(\sum_{j \in \mathcal{J}_{\geq t_\lambda}^{\text{shift}}} \hat{s}_j (t - \bar{\eta}) \right)^\alpha \text{dt} &= \int_{t \geq t_\lambda - \bar{\eta}} \left(\sum_{j \in \mathcal{J}_{\geq t_\lambda}^{\text{shift}}} \hat{s}_j (t) \right)^\alpha \text{dt} \\
&\leq \int_t \left(\sum_{j \in \hat{\mathcal{J}}^{\text{up}}} \hat{s}_j (t) \right)^\alpha \text{dt} = E(\hat{S}). \qquad (5)
\end{aligned}$$

Next, we analyze the quality cost. Note that by definition of S^{shift}, we have that for all $j \in \mathcal{J}^{\text{shift}}_{\geq t_\lambda}$ and $t \geq \hat{r}_j + \bar{\eta}$, the same amount of work for (j, r_j, p_j, v_j) has been processed by S^{shift} at time t as the amount of work for $(j, \hat{r}_j, \hat{p}_j\big(1 + \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\big), \hat{v}_j\big(1 + \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\big))$ processed by $\hat{S}$ at time $t - \bar{\eta}$. Hence, for all $t \geq \hat{r}_j + \bar{\eta}$,

$$w^{(j,r_j,p_j,v_j)}_{S^{\text{shift}}}(t) = w^{(j,\hat{r}_j,\hat{p}_j\left(1+\frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\right),\hat{v}_j\left(1+\frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\right))}_{\hat{S}}(t - \bar{\eta}).$$

By definition of $\mathcal{J}^{\text{shift}}_{\geq t_\lambda}$, we also have that for all $(j, r_j, p_j, v_j) \in \mathcal{J}^{\text{shift}}_{\geq t_\lambda}$, $|r_j - \hat{r}_j| \leq \bar{\eta}$, $p_j \leq \hat{p}_j\big(1 + \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\big)$ and $v_j \leq \hat{v}_j\big(1 + \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\big)$. Hence, we can apply the second smoothness condition with $\mathcal{J}_1 = \mathcal{J}^{\text{shift}}_{\geq t_\lambda}$ and $\mathcal{J}_2 = \{(j, \hat{r}_j, \hat{p}_j\big(1 + \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\big), \hat{v}_j\big(1 + \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\big)) : (j, r_j, p_j, v_j) \in \mathcal{J}^{\text{shift}}_{\geq t_\lambda}\} \subseteq \hat{\mathcal{J}}^{\text{up}}$. This gives:

$$\begin{aligned}
F(S^{\text{shift}}, \mathcal{J}^{\text{shift}}_{\geq t_\lambda}) &\leq F(\hat{S}, \mathcal{J}_2) + \bar{\eta}\beta(\mathcal{J}^{\text{shift}}_{\geq t_\lambda})|\mathcal{J}^{\text{shift}}_{\geq t_\lambda}| \\
&\leq F(\hat{S}, \hat{\mathcal{J}}^{\text{up}}) + \bar{\eta}\beta(\mathcal{J}^{\text{shift}}_{\geq t_\lambda})|\mathcal{J}^{\text{shift}}_{\geq t_\lambda}| \\
&= F(\hat{S}, \hat{\mathcal{J}}^{\text{up}}) + \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})} \cdot \frac{\texttt{OPT}(\hat{\mathcal{J}})}{|\hat{\mathcal{J}}|} \cdot \beta(\mathcal{J}^{\text{shift}}_{\geq t_\lambda})|\mathcal{J}^{\text{shift}}_{\geq t_\lambda}| \\
&\leq F(\hat{S}, \hat{\mathcal{J}}^{\text{up}}) + \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})} \cdot \frac{\texttt{OPT}(\hat{\mathcal{J}})}{|\hat{\mathcal{J}}|} \cdot \beta(\hat{\mathcal{J}}) \left(1 + \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\right) |\hat{\mathcal{J}}| \\
&= F(\hat{S}, \hat{\mathcal{J}}^{\text{up}}) + \eta^{\text{shift}} \left(1 + \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\right) \texttt{OPT}(\hat{\mathcal{J}}) \\
&\leq F(\hat{S}, \hat{\mathcal{J}}^{\text{up}}) + \eta^{\text{shift}}(1 + \eta^{\text{shift}})\texttt{OPT}(\hat{\mathcal{J}}), \qquad (6)
\end{aligned}$$

where the first inequality is by the second smoothness condition, the second one is is by monotonicity of F, the equality is by definition of $\bar{\eta}$, and the third inequality is by the smoothness and monotonicity of β. The last inequality is since $\beta \geq 1$.

Therefore, we get

$$\begin{aligned}
\text{cost}(S^{\text{shift}}, \mathcal{J}^{\text{shift}}_{\geq t_\lambda}) &= E(S^{\text{shift}}) + F(S^{\text{shift}}, \mathcal{J}^{\text{shift}}_{\geq t_\lambda}) \\
&\leq E(\hat{S}) + F(\hat{S}, \hat{\mathcal{J}}^{\text{up}}) + \eta^{\text{shift}}(1 + \eta^{\text{shift}})\texttt{OPT}(\hat{\mathcal{J}}) \\
&= \texttt{OPT}(\hat{\mathcal{J}}^{\text{up}}) + \eta^{\text{shift}}(1 + \eta^{\text{shift}})\texttt{OPT}(\hat{\mathcal{J}}) \\
&\leq \left(1 + \beta(\hat{\mathcal{J}}^{\text{up}})\frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\right) \texttt{OPT}(\hat{\mathcal{J}}) + \eta^{\text{shift}}(1 + \eta^{\text{shift}})\texttt{OPT}(\hat{\mathcal{J}}) \\
&\leq \left(1 + \beta(\hat{\mathcal{J}}) \left(1 + \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\right) \frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\right) \texttt{OPT}(\hat{\mathcal{J}}) + \eta^{\text{shift}}(1 + \eta^{\text{shift}})\texttt{OPT}(\hat{\mathcal{J}}) \\
&\leq (1 + \eta^{\text{shift}}(1 + \eta^{\text{shift}}))\texttt{OPT}(\hat{\mathcal{J}}) + \eta^{\text{shift}}(1 + \eta^{\text{shift}})\texttt{OPT}(\hat{\mathcal{J}}) \\
&= (1 + 2\eta^{\text{shift}}(1 + \eta^{\text{shift}}))\texttt{OPT}(\hat{\mathcal{J}}),
\end{aligned}$$

where the first inequality is by (5) and (6), the second inequality is by the first smoothness condition with $\mathcal{J}_1 = \hat{\mathcal{J}}^{\text{up}}$ and $\mathcal{J}_2 = \hat{\mathcal{J}}$, the third inequality is by smoothness of β and the last inequality since $\beta \geq 1$.

□

Corollary C.2. *Assume that cost$(.,.)$ is β-smooth, then*

$$\texttt{OPT}(\mathcal{J}^{\text{shift}}) \geq \Big[\Big(1 - \eta_2^{\frac{1}{\alpha}}\Big)^{\alpha} - \eta^{\text{shift}}\Big] \Big/ (1 + \eta^{\text{shift}})\texttt{OPT}(\hat{\mathcal{J}}).$$

Proof. We prove the result by contradiction. Assume that $\texttt{OPT}(\mathcal{J}^{\text{shift}}) < \left[\left(1-\eta_2^{\frac{1}{\alpha}}\right)^{\alpha} - \eta^{\text{shift}}\right] \Big/ (1+\eta^{\text{shift}})\texttt{OPT}(\hat{\mathcal{J}})$. Then, we have:

$$\begin{aligned}
\texttt{OPT}(\hat{\mathcal{J}}^{\text{shift}}) &\leq \left(1+\beta(\hat{\mathcal{J}}^{\text{shift}})\cdot\frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\right)\texttt{OPT}(\mathcal{J}^{\text{shift}}) + \beta(\hat{\mathcal{J}}^{\text{shift}})|\hat{\mathcal{J}}^{\text{shift}}|\cdot\frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\cdot\frac{\texttt{OPT}(\hat{\mathcal{J}})}{|\hat{\mathcal{J}}|} \\
&\leq \left(1+\beta(\hat{\mathcal{J}})\cdot\frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\right)\texttt{OPT}(\mathcal{J}^{\text{shift}}) + \beta(\hat{\mathcal{J}})|\hat{\mathcal{J}}^{\text{shift}}|\cdot\frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\cdot\frac{\texttt{OPT}(\hat{\mathcal{J}})}{|\hat{\mathcal{J}}|} \\
&\leq (1+\eta^{\text{shift}})\texttt{OPT}(\mathcal{J}^{\text{shift}}) + \eta^{\text{shift}}\texttt{OPT}(\hat{\mathcal{J}}) \\
&< \left(1-\eta_2^{\frac{1}{\alpha}}\right)^{\alpha}\texttt{OPT}(\hat{\mathcal{J}}).
\end{aligned}$$

where the first inequality is by the first smoothness condition with $\mathcal{J}_1 = \hat{\mathcal{J}}^{\text{shift}}$, $\mathcal{J}_2 = \mathcal{J}^{\text{shift}}$, and the second one is by monotonicity of β.

Next, by definition of the error η_2, we have $\texttt{OPT}(\hat{\mathcal{J}} \setminus \hat{\mathcal{J}}^{\text{shift}}) = \eta_2 \cdot \texttt{OPT}(\hat{\mathcal{J}})$. Hence, by Lemma 3.1, there exists a schedule S for $\hat{\mathcal{J}}^{\text{shift}} \cup (\hat{\mathcal{J}} \setminus \hat{\mathcal{J}}^{\text{shift}}) = \hat{\mathcal{J}}$ such that

$$\begin{aligned}
\text{cost}(S, \hat{\mathcal{J}}) &\leq \left(\texttt{OPT}(\hat{\mathcal{J}} \setminus \hat{\mathcal{J}}^{\text{shift}})^{\frac{1}{\alpha}} + \texttt{OPT}(\hat{\mathcal{J}}^{\text{shift}})^{\frac{1}{\alpha}}\right)^{\alpha} \\
&< \left((\eta_2\texttt{OPT}(\hat{\mathcal{J}}))^{\frac{1}{\alpha}} + \left(\left(1-\eta_2^{\frac{1}{\alpha}}\right)^{\alpha}\texttt{OPT}(\hat{\mathcal{J}})\right)^{\frac{1}{\alpha}}\right)^{\alpha} \\
&= \texttt{OPT}(\hat{\mathcal{J}}),
\end{aligned}$$

which contradicts the definition of $\texttt{OPT}(\hat{\mathcal{J}})$. □

Corollary C.3. *Assume that cost$(.,.)$ is β-smooth. If $\texttt{OPT}(\mathcal{J}) \leq \lambda\texttt{OPT}(\hat{\mathcal{J}})$, then*

$$\eta_2 \geq \left(1-(\lambda(1+\eta^{\text{shift}})+\eta^{\text{shift}})^{\frac{1}{\alpha}}\right)^{\alpha}.$$

Proof. Assume that $\texttt{OPT}(\mathcal{J}) \leq \lambda\texttt{OPT}(\hat{\mathcal{J}})$. Since $\mathcal{J}^{\text{shift}} \subseteq \mathcal{J}$, we get $\texttt{OPT}(\mathcal{J}^{\text{shift}}) \leq \lambda\texttt{OPT}(\hat{\mathcal{J}})$. Hence, we have:

$$\begin{aligned}
\texttt{OPT}(\hat{\mathcal{J}}^{\text{shift}}) &\leq \left(1+\beta(\hat{\mathcal{J}}^{\text{shift}})\cdot\frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\right)\texttt{OPT}(\mathcal{J}^{\text{shift}}) + \beta(\hat{\mathcal{J}}^{\text{shift}})|\hat{\mathcal{J}}^{\text{shift}}|\cdot\frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\cdot\frac{\texttt{OPT}(\hat{\mathcal{J}})}{|\hat{\mathcal{J}}|} \\
&\leq \left(1+\beta(\hat{\mathcal{J}})\cdot\frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\right)\texttt{OPT}(\mathcal{J}^{\text{shift}}) + \beta(\hat{\mathcal{J}})|\hat{\mathcal{J}}^{\text{shift}}|\cdot\frac{\eta^{\text{shift}}}{\beta(\hat{\mathcal{J}})}\cdot\frac{\texttt{OPT}(\hat{\mathcal{J}})}{|\hat{\mathcal{J}}|} \\
&\leq (1+\eta^{\text{shift}})\texttt{OPT}(\mathcal{J}^{\text{shift}}) + \eta^{\text{shift}}\texttt{OPT}(\hat{\mathcal{J}}) \\
&\leq (\lambda(1+\eta^{\text{shift}})+\eta^{\text{shift}})\texttt{OPT}(\hat{\mathcal{J}}).
\end{aligned}$$

where the first inequality is by the first smoothness condition with $\mathcal{J}_1 = \hat{\mathcal{J}}^{\text{shift}}$, $\mathcal{J}_2 = \mathcal{J}^{\text{shift}}$, and the second one is by monotonicity of β.

Next, assume by contradiction, that $\eta_2 < \left(1-(\lambda(1+\eta^{\text{shift}})+\eta^{\text{shift}})^{\frac{1}{\alpha}}\right)^{\alpha}$, which implies that $\texttt{OPT}(\hat{\mathcal{J}} \setminus \hat{\mathcal{J}}^{\text{shift}}) < \left(1-(\lambda(1+\eta^{\text{shift}})+\eta^{\text{shift}})^{\frac{1}{\alpha}}\right)^{\alpha}\texttt{OPT}(\hat{\mathcal{J}})$.

Then, by Lemma 3.1, there exists a schedule S for $\hat{\mathcal{J}}^{\text{shift}} \cup (\hat{\mathcal{J}} \setminus \hat{\mathcal{J}}^{\text{shift}}) = \hat{\mathcal{J}}$ such that

$$\begin{aligned}
&\text{cost}(S, \hat{\mathcal{J}}) \\
&\leq \left(\texttt{OPT}(\hat{\mathcal{J}} \setminus \hat{\mathcal{J}}^{\text{shift}})^{\frac{1}{\alpha}} + \texttt{OPT}(\hat{\mathcal{J}}^{\text{shift}})^{\frac{1}{\alpha}}\right)^{\alpha} \\
&< \left(((\lambda(1+\eta^{\text{shift}})+\eta^{\text{shift}})\texttt{OPT}(\hat{\mathcal{J}}))^{\frac{1}{\alpha}} + \left(\left(1-(\lambda(1+\eta^{\text{shift}})+\eta^{\text{shift}})^{\frac{1}{\alpha}}\right)^{\alpha}\texttt{OPT}(\hat{\mathcal{J}})\right)^{\frac{1}{\alpha}}\right)^{\alpha} \\
&= \texttt{OPT}(\hat{\mathcal{J}}),
\end{aligned}$$

which contradicts the definition of $\texttt{OPT}(\hat{\mathcal{J}})$. Hence, $\eta_2 \geq \left(1 - (\lambda(1+\eta^{\text{shift}}) + \eta^{\text{shift}})^{\frac{1}{\alpha}}\right)^{\alpha}$. □

Theorem C.4. *Assume that cost(.,.) is β-smooth. Then, for any $\lambda \in (0,1], \eta^{shift} \in [0,1)$, the competitive ratio of* TPE-S *run with trust parameter λ, a γ-competitive algorithm* ONLINEALG*, an optimal offline algorithm* OFFLINEALG*, and shift tolerance η^{shift} is at most*

$$\begin{cases} \gamma & \text{if } \texttt{OPT}(\mathcal{J}) \leq \lambda \texttt{OPT}(\hat{\mathcal{J}}) \\ \dfrac{\left((1+2\eta^{shift}(1+\eta^{shift}))^{\frac{1}{\alpha}} + \gamma^{\frac{1}{\alpha}}\left(\lambda^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}}\right)\right)^{\alpha}}{\max\left\{\lambda, \eta_1 + \left(\left(1-\eta_2^{\frac{1}{\alpha}}\right)^{\alpha} - \eta^{shift}\right)(1+\eta^{shift})^{-1}\right\}} & \text{otherwise.} \end{cases}$$

Proof. Similarly as in the proof of Theorem 3.4, we have that if $\texttt{OPT}(\mathcal{J}) \leq \lambda \texttt{OPT}(\hat{\mathcal{J}})$, then

$$\text{cost}(S, \mathcal{J}) \leq \gamma \cdot \texttt{OPT}(\mathcal{J}). \tag{7}$$

Next, we assume that there is $t_\lambda \geq 0$ such that $\texttt{OPT}(\mathcal{J}_{\leq t_\lambda}) > \lambda \texttt{OPT}(\hat{\mathcal{J}})$. Hence we immediately have:

$$\texttt{OPT}(\mathcal{J}) \geq \texttt{OPT}(\mathcal{J}_{\leq t_\lambda}) > \lambda \texttt{OPT}(\hat{\mathcal{J}}).$$

By Corollary C.2 and by definition of the error η_1, we also get the following lower bound on the optimal schedule:

$$\begin{aligned} \texttt{OPT}(\mathcal{J}) &\geq \texttt{OPT}(\mathcal{J} \setminus \mathcal{J}^{\text{shift}}) + \texttt{OPT}(\mathcal{J}^{\text{shift}}) \\ &\geq \eta_1 \texttt{OPT}(\hat{\mathcal{J}}) + \left[\left(1 - \eta_2^{\frac{1}{\alpha}}\right)^{\alpha} - \eta^{\text{shift}}\right] \Big/ (1+\eta^{\text{shift}}) \texttt{OPT}(\hat{\mathcal{J}}). \end{aligned}$$

Therefore,

$$\texttt{OPT}(\mathcal{J}) \geq \max\left\{\lambda, \eta_1 + \left[\left(1 - \eta_2^{\frac{1}{\alpha}}\right)^{\alpha} - \eta^{\text{shift}}\right] \Big/ (1+\eta^{\text{shift}})\right\} \texttt{OPT}(\hat{\mathcal{J}}). \tag{8}$$

We now upper bound the cost of the schedule output by our algorithm. By the same argument as in the proof of Lemma 3.2, we get:

$$\text{cost}(S^{on}, \mathcal{J} \setminus \mathcal{J}^{\text{shift}}_{\geq t_\lambda}) \leq \gamma \cdot \texttt{OPT}(\hat{\mathcal{J}}) \left(\lambda^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}}\right)^{\alpha}.$$

Now, from Lemma C.1, we have

$$\text{cost}(S^{\text{shift}}, \mathcal{J}^{\text{shift}}_{\geq t_\lambda}) \leq (1 + 2\eta^{\text{shift}}(1+\eta^{\text{shift}})) \texttt{OPT}(\hat{\mathcal{J}}).$$

Therefore, by applying Lemma 3.1, we get:

$$\begin{aligned} \text{cost}(S, \mathcal{J}) &\leq \left(\text{cost}(S^{\text{shift}}, \mathcal{J}^{\text{shift}}_{\geq t_\lambda})^{\frac{1}{\alpha}} + \text{cost}(S^{on}, \mathcal{J} \setminus S^{\text{shift}})^{\frac{1}{\alpha}}\right)^{\alpha} \\ &\leq \texttt{OPT}(\hat{\mathcal{J}}) \left((1 + 2\eta^{\text{shift}}(1+\eta^{\text{shift}}))^{\frac{1}{\alpha}} + \gamma^{\frac{1}{\alpha}}(\lambda^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}})\right)^{\alpha}. \end{aligned} \tag{9}$$

Hence, we get the following upper bound on the competitive ratio of Algorithm 2:

$$\begin{aligned} &\frac{\text{cost}(S, \mathcal{J})}{\texttt{OPT}(\mathcal{J})} \\ &= \mathbf{1}_{\texttt{OPT}(\mathcal{J}) \leq \lambda \texttt{OPT}(\hat{\mathcal{J}})} \frac{\text{cost}(S, \mathcal{J})}{\texttt{OPT}(\mathcal{J})} + \mathbf{1}_{\texttt{OPT}(\mathcal{J}) > \lambda \texttt{OPT}(\hat{\mathcal{J}})} \frac{\text{cost}(S, \mathcal{J})}{\texttt{OPT}(\mathcal{J})} \\ &\leq \mathbf{1}_{\texttt{OPT}(\mathcal{J}) \leq \lambda \texttt{OPT}(\hat{\mathcal{J}})} \cdot \gamma + \mathbf{1}_{\texttt{OPT}(\mathcal{J}) > \lambda \texttt{OPT}(\hat{\mathcal{J}})} \frac{\left((1 + 2\eta^{\text{shift}}(1+\eta^{\text{shift}}))^{\frac{1}{\alpha}} + \gamma^{\frac{1}{\alpha}}(\lambda^{\frac{1}{\alpha}} + \eta_1^{\frac{1}{\alpha}})\right)^{\alpha}}{\max\left\{\lambda, \eta_1 + \left[\left(1 - \eta_2^{\frac{1}{\alpha}}\right)^{\alpha} - \eta^{\text{shift}}\right] \Big/ (1+\eta^{\text{shift}})\right\}}. \end{aligned}$$

□

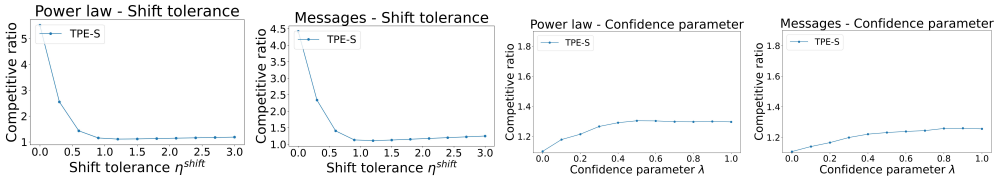

Figure 3: The competitive ratio achieved by our algorithm, TPE-S and the benchmark algorithm as a function of the shift tolerance η^{shift} (row 1) and as a function of the confidence parameter λ (row 2).

E Additional Experiments

Here, we also evaluate the impact of setting the parameters η^{shift} and λ on the two other datasets (power law and real datasets). The result are presented in Figure 3. We observe similar behaviors as for the periodic dataset.

F Comparison with [7, 4]

In [7, 4], the authors consider the energy minimization problem with deadlines, which, as detailed in Section 3.3, is a special case of our general framework. For this problem, they propose two different learning-augmented algorithms. We present here some elements of comparison with our algorithm for (GESP). We first show in Section F.1 and F.2 that our prediction model and results generalize the ones in [7]: they are similar in the case of *uniform* deadlines and generalize the ones in [7] for *general* deadlines. We also note that they are incomparable to those in [4]. In Section F.3, we then discuss the algorithmic differences with [7] for the special case of energy with uniform deadlines.

F.1 Discussion about the prediction and error model in comparison to [7, 4]

At a high level, the prediction models considered in [7] and [4] are qualified by [4] as 'orthogonal'. In [4], the number of jobs is known in advance, as well as the exact processing time for each job, however, the release time and deadlines are only revealed when a job arrives, and the error is proportional to the maximal shift in these values. On the contrary, in [7], the release times and deadlines are known in advance, and the prediction regards the total workload at each time step. The error is then defined as a function of the total variation of workload, which is the analog of additional and missing jobs in our setting. Note that in the model in [4], the predicted and true set of jobs need to contain exactly the same number of jobs, whereas the model in [7] and our model allow for extra or missing jobs.

Comparison with the prediction model and the error metrics in [7] for energy minimization with uniform deadlines. Note that the prediction model used in [7] for the energy minimization with deadlines problem is slightly different than ours: the prediction is the total workload w_i^{pred} that arrives at each time step i and needs to be scheduled before time $i + D$, and the error metric is defined as

$$\text{err}(w^{\text{real}}, w^{\text{pred}}) := \sum_i ||w_i^{\text{real}} - w_i^{\text{pred}}||^{\alpha}, \tag{10}$$

where w^{real} denotes the real workload at each time step.

However, in the specific case of energy minimization under *uniform* deadline constraints, our prediction model and error metric and the ones from [7] are comparable: a workload w_i^{real} that arrives at time i is equivalent in our setting to receiving w_i^{real} unit jobs with release time $r = i$ and a common deadline $d = i + D$. Moreover, we prove the following lemma, which shows that a small error in the sense of [7] induces a small error $\eta(\mathcal{J}, \hat{\mathcal{J}})$ in the sense defined in Section 2.

Lemma F.1. *For any constant $D > 0$, and any instance $(\mathcal{J}, \hat{\mathcal{J}})$, where at each time i, $\mathcal{J}$ is composed of w_i^{real} jobs of one time unit with deadline $i + D$ and $\hat{\mathcal{J}}$ is composed of w_i^{pred} jobs of one time unit with deadline $i + D$, we have:*

$$\eta(\mathcal{J}, \hat{\mathcal{J}}) \cdot \texttt{OPT}(\hat{\mathcal{J}}) = \max\{\texttt{OPT}(\mathcal{J} \setminus \hat{\mathcal{J}}), \texttt{OPT}(\hat{\mathcal{J}} \setminus \mathcal{J})\} \leq D \cdot \textit{err}(w^{real}, w^{pred}).$$

Proof. For convenience, we write $\Delta_i = ||w_i^{\text{real}} - w_i^{\text{pred}}||$. We can then write $(\mathcal{J} \setminus \hat{\mathcal{J}}) \cup (\hat{\mathcal{J}} \setminus \mathcal{J})$ as the instance which, at each time step i, is composed of Δ_i unit size jobs with a common deadline $i + D$.

We now upper bound the optimal cost for $(\mathcal{J} \setminus \hat{\mathcal{J}}) \cup (\hat{\mathcal{J}} \setminus \mathcal{J})$ by the cost obtained by the Average Rate heuristic (AVR) (first introduced in [29]). For each j, The AVR algorithm schedules uniformly the Δ_j units of work arriving at time j over the next D time steps. This is equivalent to setting the speed $s_j(t)$ for each workload Δ_j at time $t \in [j, \ldots, j + D]$ to $\frac{\Delta_j}{D}$ and set $s_j(t) = 0$ everywhere else. For all $t \geq 0$, the machine then runs at total speed $\sum_j s_j(t)$.

Letting E_{AVR} denote the total cost of the AVR heuristic, we get:

$$\begin{aligned}
\max\{\texttt{OPT}(\mathcal{J} \setminus \hat{\mathcal{J}}), \texttt{OPT}(\hat{\mathcal{J}} \setminus \mathcal{J})\} &\leq \texttt{OPT}((\mathcal{J} \setminus \hat{\mathcal{J}}) \cup (\hat{\mathcal{J}} \setminus \mathcal{J})) \\
&\leq E_{\text{AVR}}((\mathcal{J} \setminus \hat{\mathcal{J}}) \cup (\hat{\mathcal{J}} \setminus \mathcal{J})) \\
&= \sum_{t=1}^{\infty} \left(\sum_j \mathbf{1}_{s_j(t) \neq 0} s_j(t) \right)^{\alpha} \\
&\leq \sum_{t=1}^{\infty} |\{j : s_j(t) \neq 0\}|^{\alpha} \left(\max_{j : s_j(t) \neq 0} s_j(t) \right)^{\alpha} \\
&\leq \sum_{t=1}^{\infty} D^{\alpha} \left(\max_{j : s_j(t) \neq 0} s_j(t) \right)^{\alpha} \\
&\leq \sum_{t=1}^{\infty} D^{\alpha} \sum_{j : s_j(t) \neq 0} s_j(t)^{\alpha} \\
&= \sum_j D^{\alpha} s_j(t)^{\alpha} \sum_t \mathbf{1}_{s_j(t) \neq 0} \\
&\leq \sum_j D^{\alpha} s_j(t)^{\alpha} D \\
&= \sum_j \Delta_j^{\alpha} D \\
&= D \cdot \text{err}(w^{\text{real}}, w^{\text{pred}}),
\end{aligned}$$

where the fourth and sixth inequalities are since by definition of the AVR algorithm, each workload $\Delta_j > 0$ has only positive speed on time steps $[j, \ldots, j + D]$. □

Comparison of the error metrics for general objective functions. We illustrate here that for a more general GESP problem, the error metric we define can be tighter than the one in [7] (in the sense that there are instances $(\mathcal{J}, \hat{\mathcal{J}})$ and quality cost functions F such that $\eta(\mathcal{J}, \hat{\mathcal{J}}) << \text{err}(w^{\text{real}}, w^{\text{pred}})$) and that it may better adapt to the specific cost function under consideration.

To illustrate this point, consider an instance where the prediction is the realization plus an additional workload of k jobs that all arrive at time 0, and consider the objective of minimizing total energy plus flow time. In this case, the error computed in (10) is k^{α}, whereas the error $\eta(\mathcal{J}, \hat{\mathcal{J}})$ we define is the optimal cost for the k extra jobs. By using results from [3], this is equal to $k^{\frac{2\alpha - 1}{\alpha}}$ ($<< k^{\alpha}$ when α grows large). Hence our error metric is tighter in this case.

F.2 Comparison with the theoretical guarantees in [7]

We compare below the theoretical guarantees in Theorem 3.4 and the ones shown in [7] for the specific problem of energy minimization with *uniform* deadlines. We note that we also generalize these results to the case of *general* deadlines to obtain the first guarantee that smoothly degrades as a function of the prediction error in that setting. Note that for general deadlines, [7] only obtain consistency and robustness, but not smoothness.

Comparison in the case of uniform deadlines. For convenience of the reader, we first recall below the guarantee proven in [7].

Theorem F.2 (Theorem 8 in [7])**.** *For any given $\epsilon > 0$, algorithm LAS constructs, for the energy minimization with deadlines problem, a schedule of cost at most* $\min\{(1+\epsilon)\texttt{OPT} + O((\frac{\alpha}{\epsilon})^\alpha)\textit{err}, O((\frac{\alpha}{\epsilon})^\alpha)\texttt{OPT}\}$*, where*

$$\textit{err}(w^{\textit{real}}, w^{\textit{pred}}) := \sum_i ||w_i^{\textit{real}} - w_i^{\textit{pred}}||^\alpha,$$

which is a similar dependency in ϵ as the one proved in Theorem 3.4. In particular, for all $\epsilon > 0$, Algorithm LAS achieves a consistency of $(1+\epsilon)$ for a robustness factor of $O((\frac{\alpha}{\epsilon})^\alpha)$. On the other hand, when running Algorithm 1 with parameter $\lambda = (\frac{\epsilon}{C2^\alpha})^\alpha$ and the AVERAGE RATE heuristic [29] as ONLINEALG (which was proven to have a 2^α competitive ratio in [7]), we obtain, by plugging $\lambda = (\frac{\epsilon}{C2^\alpha})^\alpha$ in the bounds provided in Corollary 3.5, a consistency of $(1+\epsilon)$ for a robustness factor of $O(\frac{4^{\alpha^2}}{\epsilon^{\alpha-1}})$.

F.3 Comparison with the algorithm (LAS) in [7]

In this section, we discuss the technical differences with the algorithm (LAS) proposed in [7] for the energy with deadlines problem. We first note that [7] only shows smoothness, consistency and robustness in the uniform deadline case, where all jobs must be completed within D time steps from their release time. For the general deadline case, [7] presents a more complicated algorithm and only show consistency and robustness. The authors note that "one can also define smooth algorithms for general deadlines as [they] did in the uniform case. However, the prediction model and the measure of error quickly get complex and notation heavy". On the contrary, Algorithm 1 remains simple, captures the general deadline case, and is also endowed with smoothness guarantees. We now discuss more specifically the technical differences.

Robustification technique. [7] uses a convolution technique for the uniform deadline case, and a more complicated procedure that separates each interval into a base part and an auxiliary part for the general deadline case. On the other hand, our robustification technique is based on a simpler two-phase algorithm.

We now give some intuition about why a direct generalisation of the techniques in [7] to general objective functions does not seem straightforward. The main technical difficulty is that in [7], each job j must be completed before its deadline d_j, which is revealed to the decision maker at the time the job arrives and is used by the algorithm. For a general objective function, we do not have a deadline ; however, one could think about using the total completion time c_j of each job instead. The issue is that c_j may depend on all future job arrivals and is not known at the time the job arrives, hence it cannot be used directly by the algorithm.

To illustrate this point, consider the objective of minimizing total energy plus flow time with $\alpha = 2$. Consider two instances, where the first one has 1 job arriving at time 0 and the second one has 1 job arriving at time 0 and $n-1$ jobs arriving at time $\frac{1}{\sqrt{n}}$. In the first case, the optimal is to complete the first job in 1 unit of time, whereas it is completed in $\frac{1}{\sqrt{n}}$ unit of time in the second case. Furthermore, this can be only deduced after the $n-1$ other jobs have arrived. Hence, the completion times for the first job significantly differ in the two cases. Since it is not immediate how to generalize the technique in [7] without knowing c_j at the time each job j arrives, this motivated our choice of a different robustification technique.

Smoothness and consistency technique. To obtain smoothness and consistency guarantees, we use a similar technique as in [7] (summing the speeds obtained by computing an offline schedule for the predicted jobs and an online schedule for the extra jobs), with two main differences:
(1) In [7], the extra jobs arriving at each time i are scheduled uniformly over the next D time units. On the other hand, our algorithm computes the speeds for all extra jobs by following an auxiliary online algorithm given as an input to the decision maker. In fact, the technique from [7] can be interpreted as a special case of our algorithm, where the auxiliary algorithm is the AVERAGE RATE heuristic [29].
(2) The offline schedule we compute is conceptually identical to the one used in [7], however, our

online schedule differs, as it needs to integrate two different types of extra jobs: (1) the extra jobs that arrive during the second phase of the algorithm ($t \geq t_\lambda$), and (2) the jobs that were not finished during the first phase of the algorithm. [7] only needs to handle the first type of extra jobs. This results in a different analysis.