Stochastic Online Scheduling on Unrelated Machines

Varun Gupta[1], Benjamin Moseley[2] ⋆, Marc Uetz[3], and Qiaomin Xie[4]

[1] University of Chicago, varun.gupta@chicagobooth.edu
[2] Washington University in St. Louis, bmoseley@wustl.edu
[3] University of Twente, m.uetz@utwente.nl
[4] University of Illinois at Urbana-Champaign, qxie3@illinois.edu

Abstract. We derive the first performance guarantees for a combinatorial online algorithm that schedules stochastic, nonpreemptive jobs on unrelated machines to minimize the expectation of the total weighted completion time. Prior work on unrelated machine scheduling with stochastic jobs was restricted to the offline case, and required sophisticated linear or convex programming relaxations for the assignment of jobs to machines. Our algorithm is purely combinatorial, and therefore it also works for the online setting. As to the techniques applied, this paper shows how the dual fitting technique can be put to work for stochastic and nonpreemptive scheduling problems.

1 Introduction

The scheduling of jobs on multiple, parallel machines is a fundamental problem both in combinatorial optimization and systems theory. There is a vast amount of different model variants as well as applications, which is testified by the existence of the handbook [18]. A well studied class of problems is scheduling a set of n nonpreemptive jobs that arrive over time on m unrelated machines with the objective of minimizing the total weighted completion time. Here, unrelated machines refers to the fact that the matrix that describes the processing times of all jobs on all machines can have any rank larger than 1. The offline version of that problem is denoted $\mathrm{R} \,|\, r_j \,|\, \sum w_j C_j$ in the three-field notation of Graham et al. [8], and it has always been a cornerstone problem for the development of new techniques in the design of (approximation) algorithms, e.g. [4, 11, 17, 29].

We here address the online version of that problem with stochastic jobs. Online means that jobs arrive over time, and the set of jobs is unknown a priori. With respect to online models in scheduling, we refer to [13, 26] for pointers to relevant work. In many systems, the scheduler may not know the exact processing times of jobs upon arrival. Different approaches have been introduced to cope with this uncertainty. If jobs can be preempted, then non-clairvoyant schedulers have been studied that do not know the processing time of the jobs until the job

⋆ Supported in part by a Google Research Award, a Yahoo Research Award and NSF Grant CCF-1617724.

is completed [25, 5, 15, 9, 12]. Unfortunately, if preemption is not allowed then any algorithm has poor performance in the non-clairvoyant model, as the lower bound for approximability is $\Omega(n)$.

That suggests that the non-clairvoyant model is perhaps too pessimistic. Even though exact processing times may be unknown, it is not unrealistic to assume that at least an an estimate of the true processing times is available. For such systems, a model that is used is *stochastic scheduling*. In the stochastic scheduling model the job's processing times are given by random variables. A *non-anticipatory* scheduler only knows this random variable P_j that encodes the possible realizations of job j's processing time. If the scheduler starts a job on a machine, then that job must be run to completion *non-preemptively*, and it is only when the job completes that the scheduler learns the actual processing time. Both the scheduler and the optimal solution are non-anticipatory, which roughly means that the future is uncertain for both, the scheduler and the adversary. Stochastic scheduling has been well-studied, including fundamental work such as [22, 23] and approximation algorithms, e.g. [24, 31, 20, 30, 28].

This paper considers online scheduling of non-preemptive, stochastic jobs in an unrelated machine environment to minimize the total weighted completion time.This paper addresses the same problem as [20], however not for identical machines, but for the most general, *unrelated* machines model. In the stochastic unrelated machine setting, that means that the scheduler is given a probability distribution of a job's processing time which is machine-dependent, and there need not be any correlation between the jobs' processing time distributions on different machines.

When all machines are identical, perhaps the most natural algorithm is Weighted Shortest Expected Processing Time (WSEPT) first, which always assigns a job with the maximum ratio of weight over expected size when a machine is free. With unit weights, this boils down to greedily scheduling jobs according to smallest expected size, or SEPT. When there is a single machine, WSEPT is optimal [27]. Further, in the case were job sizes are deterministic and arrive at the same time, SEPT is optimal [10]. In the identical machines setting, SEPT is optimal if job sizes are exponentially distributed [6, 34], or more generally, are stochastically comparable in pairs [33]. Some extensions of these optimality results to the problem with weights exist as well [16]. However for more general distributions, simple solutions fail [32], and our knowledge of optimal scheduling policies is limited.

For this reason, approximation algorithms have been studied. With the notable exception of [14], all approximation algorithms have performance guarantees that depend on an upper bound Δ on the squared coefficient of variation of the underlying random variables. Möhring, Schulz and Uetz [24] established the first approximation algorithms for the problem via the first linear programming relaxation for stochastic scheduling. Their work gave a $(3 + \Delta)$-approximation when jobs are released over time (yet offline), and they additionally showed that WSEPT is a $\frac{(3+\Delta)}{2}$-approximation when jobs arrive together[5]. These results have

[5] The ratio is slightly better, but for simplicity we ignore the additive $\Theta(1/m)$ term.

been built on and generalized in several settings [31, 21, 20, 32, 30, 28], notably in [20] for the online setting. The currently best known result when jobs are released over time (yet offline) is a $(2 + \Delta)$-approximation by Schulz [28]. In the online setting [28] gives a $(2.309 + 1.309\Delta)$-competitive algorithm. These results build on an idea from [7] to use a preemptive, fast single machine relaxation, next to the relaxation of [24]. The work of Im, Moseley and Pruhs[14] gave the first results independent of Δ showing that there exist poly-logarithmic approximation algorithms under some assumptions. All these papers address problems with identical machines.

For some 15 years after the results of [24] for the identical machines case, no non-trivial results were known for the *unrelated* machines case despite being a major target in the area. Recently Skutella et al. [30] gave a $\frac{3+\Delta}{2}$-approximation algorithm when jobs arrive at the same time, and a $(2 + \Delta)$-approximation when jobs are released over time (yet offline). Central to unlocking an efficient approximation algorithm for the unrelated machines case was the introduction of a time-indexed linear program that lower bounds the objective value of the optimal non-anticipatory scheduling policy. It is this LP that allows the authors to overcome the complexities of the unrelated machines setting.

The present paper targets the more realistic *online* setting for the scheduling of stochastic jobs on unrelated machines. A priori, it is not clear that there should exist an algorithm with small competitive ratio. Prior work for the offline problem requires sophisticated linear [30] or convex [3] programming relaxations. Good candidates for online algorithms are simple and combinatorial, but even discovering an offline algorithm that is combinatorial remains a target.

Results: This paper shows that there exists an online, $O(\Delta)$-competitive, *combinatorial* algorithm for stochastic scheduling on unrelated machines. We thereby (1) develop the first combinatorial algorithm for stochastic scheduling on unrelated machines, (2) give the first competitive combinatorial online algorithm for unrelated machines (even for the deterministic setting), and (3) introduce new techniques for bounding the performance of stochastic scheduling algorithms.

We address (1) and (2) by giving a simple greedy online algorithm for stochastic scheduling on unrelated machines. The algorithm rests on the simple idea to assign jobs to those machines where the expected increase of the objective is minimal, an idea that was used also before, e.g. in [2, 19, 20]. In the online-list model, where jobs arrive online (at time 0) and must be assigned to a machine immediately upon arrival, we establish a $(8 + 4\Delta)$-competitive algorithm. In the online-time model, where jobs arrive over time, we derive a $(144 + 72\Delta)$-competitive algorithm The $\Omega(\Delta)$ lower bound for fixed assignment policies in [30] yields that both these results are asymptotically tight in Δ.

As to (3), we develop how to use dual fitting techniques for stochastic and non-preemptive algorithm analysis. The technique has been used in [1] for deterministic and preemptive problems. This paper establishes that dual fitting is a powerful technique for bounding the performance of algorithms in stochastic settings and it is this technique that unlocks the ability to analyze a combinatorial algorithm. This is the first use of dual fitting for stochastic scheduling.

2 Notation & Preliminaries

We are given a set of unrelated parallel machines M of cardinality m. We consider two online models. In the first model, known as *online-list*, we are presented a sequence of jobs $j \in J$, which are presented to us one after the other, and whenever a job is presented we have to assign it to one of the machines. It is unknown how many jobs will arrive, but once all jobs in J have arrived, the jobs assigned to any one of the machines must be scheduled on that machine. In the other model, known as *online-time*, time progresses and jobs appear over time at their individual release times r_j. At the moment of arrival a job must be assigned to a machine, but can possibly wait on that machine until it is finally processed. Each job j needs to be executed on any one of the machines $i \in M$, and each machine can process at most one job at a time.

The jobs are nonpreemptive. That means that a job, once started, must not be interrupted until its completion. Moreover, the jobs are stochastic, meaning that each job j's processing time is only revealed in the form of a random variable P_{ij} for every machine $i \in M$. If job j is assigned to machine i, its processing time will be random according to P_{ij}. It is allowed that certain jobs $j \in J$ cannot be processed on certain machines $i \in M$, in which case $\mathbb{E}[P_{ij}] = \infty$.

In the stochastic scheduling model, the actual realization of the processing time of a job j becomes only known at the moment that the job completes. We are looking for a non-anticipatory scheduling policy Π which minimizes the expected total weighted completion time $\mathbb{E}\big[\sum_j w_j C_j\big]$, where C_j denotes the completion time of job j.

We will assume for simplicity that the random variables P_{ij} are discrete and integer valued. This assumption comes at the cost of a multiplicative factor $(1+\varepsilon)$ in the final approximation ratio, for any $\varepsilon > 0$ [30]. We will subsequently make use of the following facts about first and second moments of discrete random variables; they also appear in [30].

Lemma 1. *Let X be an integer-valued, nonnegative random variable. Then,*

$$\sum_{r\in\mathbb{Z}_{\geq 0}} \mathbb{P}[X > r] = \mathbb{E}[X] \quad \text{and} \quad \sum_{r\in\mathbb{Z}_{\geq 0}} (r + \tfrac{1}{2})\, \mathbb{P}[X > r] = \frac{1}{2}\mathbb{E}[X^2]\,.$$

Definition 1. *Let X be a nonnegative random variable. The* squared coefficient of variation *is defined as the scaled variance of X, that is,*

$$\mathbb{CV}[X]^2 := \mathbb{V}\text{ar}[X]/\mathbb{E}[X]^2\,,$$

where $\mathbb{V}\text{ar}[X] = \mathbb{E}[X^2] - E[X]^2$.

2.1 Stochastic Online Scheduling & Policies

The setting that we consider in this paper is that of stochastic online scheduling as defined also in [20]. That means that (the existence of) a jobs j is unknown before it arrives, and upon arrival, only the random variables P_{ij} for the possible processing times on machine $i = 1, \ldots, m$ are known. At any given time t, a non-anticipatory online scheduling policy is allowed to use only that information that is available at time t. In particular, it may anticipate the (so far) realized processing times of jobs up to time t. For example, a job that has possible sizes 1, 3 or 4 with probabilities 1/3 each, and has been running for 2 time units, will have processing times 3 or 4, each with probability 1/2. That adaptivity over time may be relevant in order to minimize the expectation of the total weighted completion times is well known even in the offline setting, e.g. [32]. We refer to [20] for a more thorough discussion of the stochastic online model.

For simplicity of notation, we denote by OPT the expected total weighted completion time of an optimal, non-anticipatory online scheduling policy for the problem. That is, OPT is our benchmark, and we seek to find a non-anticipatory online scheduling policy (an algorithm) with expected performance ALG close to OPT. Note that, for convenience we use the same notation for both algorithm and its expected performance.

We may assume w.l.o.g. that no pair of job and machine exists with $\mathbb{E}[P_{ij}] = 0$, as then we can always schedule such job j at machine i (whenever released) at minimum possible cost. That said, we may further assume that $\mathbb{E}[P_{ij}] \geq 1$ for all machines i and jobs j, by scaling.

3 Linear Programming Relaxations

As previously discussed also in [30, §8], we are going to use variables y_{ijs} that denote the probability that job j is being processed on machine i within time interval $[s, s+1]$, under some given and fixed scheduling policy. It is well known that y_{ijs} can be linearly expressed in terms of the variables x_{ijt}, which denote the probability that job j is started at time t on machine i, as follows

$$y_{ijs} = \sum_{t=0}^{s} x_{ijt}\, \mathbb{P}[P_{ij} > s - t] \ . \tag{1}$$

The fact that any machine can process at most one job at a time can be written as

$$\sum_{j \in J} y_{ijs} \leq 1 \qquad \text{for all } i \in M,\ s \in \mathbb{Z}_{\geq 0}. \tag{2}$$

Moreover, making use of (1) and the first part of Lemma 1, the fact that each job needs to be completely processed translates into the constraints

$$\sum_{i \in M} \sum_{s \in \mathbb{Z}_{\geq 0}} \frac{y_{ijs}}{\mathbb{E}[P_{ij}]} = 1 \qquad \text{for all } j \in J. \tag{3}$$

Finally, with the help of (1) and the second part of Lemma 1, the expected completion time of a job j can be expressed in y_{ijs} variables as

$$C_j^S := \sum_{i \in M} \sum_{s \in \mathbb{Z}_{\geq 0}} \left(\frac{y_{ijs}}{\mathbb{E}[P_{ij}]} \left(s + \tfrac{1}{2}\right) + \frac{1 - \mathbb{CV}[P_{ij}]^2}{2}\, y_{ijs} \right) \qquad \text{for all } j \in J\,, \tag{4}$$

where we labeled the expected completion time variables with a superscript S for "stochastic", for reasons that will become clear shortly.

For the analysis to follow, we also need to express the fact that the expected completion time of a job cannot be smaller than its expected processing time

$$C_j^S \geq \sum_{i\in M}\sum_{s\in\mathbb{Z}_{\geq 0}} y_{ijs} \qquad \text{for all } j \in J. \tag{5}$$

That said, we can write down the following LP relaxation for the unrelated machine scheduling problem, which extends the one given in [30] by the additional constraints (5).

$$\begin{aligned} \min \quad & z^S = \sum_{j\in J} w_j\, C_j^S \\ \text{s.t.} \quad & (2),\ (3),\ (4),\ (5) \\ & y_{ijs} \geq 0 \qquad\qquad \text{for all } j\in J,\ i\in M,\ s\in\mathbb{Z}_{\geq 0}. \end{aligned} \tag{S}$$

Subsequently, we want to work with the dual of this relaxation. However the term $-\mathbb{CV}[P_{ij}]^2$ in the primal objective would appear in the dual constrains. As we do not know how to deal with this negative term in the analysis that is to follow, we are going to factor it out.

To that end, we first define a simpler, i.e., deterministic version for the expected completion times (4), labeled with "P" to distinguish it from the previous formulation, by letting

$$C_j^P = \sum_{i\in M}\sum_{s\in\mathbb{Z}_{\geq 0}} \frac{y_{ijs}}{\mathbb{E}[P_{ij}]}\left(s+\frac{1}{2}\right) + \frac{y_{ijs}}{2} \qquad \text{for all } j\in J. \tag{6}$$

Now consider the following linear programming problem

$$\begin{aligned} \min \quad & z^P = \sum_{j\in J} w_j\, C_j^P \\ \text{s.t.} \quad & (2),\ (3),\ (6) \\ & y_{ijs} \geq 0 \qquad\qquad \text{for all } j\in J,\ i\in M,\ s\in\mathbb{Z}_{\geq 0}\,. \end{aligned} \tag{P}$$

This corresponds to a time-indexed linear programming relaxation for a purely deterministic, unrelated machine scheduling problem where the random processing times are fixed at their expected values $\mathbb{E}[P_{ij}]$.

We are now going to establish a relation between these two relaxations. To do that, let us define an upper bound on the squared coefficient of variation by

$$\Delta := \max_{i,j} \mathbb{CV}[P_{ij}]^2\,.$$

Next, for any given solution y of (S) or (P), we define

$$H(y) := \sum_{j\in J} w_j \sum_{i\in M}\sum_{s\in\mathbb{Z}_{\geq 0}} y_{ijs}\,.$$

Now let y^S denote an optimal solution to (S), and note that by constraints (5),

$$H(y^S) = \sum_{j\in J} w_j \sum_{i\in M} \sum_{s\in\mathbb{Z}_{\geq 0}} y^S_{ijs} \leq \sum_{j\in J} w_j C^S_j = z^S(y^S) \leq \mathsf{OPT}\,.$$

The next lemma is crucial for our analysis and establishes the relation between the two relaxations.

Lemma 2. *The optimal solution values z^P and z^S of the linear programming relaxations* (P) *and* (S) *fulfill*

$$z^P \leq \big(1 + \frac{\Delta}{2}\big) z^S\,.$$

Recalling that (S) is a relaxation for the stochastic scheduling problem, we conclude the following.

Corollary 1. *The optimal solution value z^P of the linear programming relaxation* (P) *is bounded by the expected performance of an optimal scheduling policy by*

$$z^P \leq \big(1 + \frac{\Delta}{2}\big)\mathsf{OPT}\,.$$

Just like [1], we now consider the dual of (P), which will have unconstrained variables α_j for all $j \in J$ and nonnegative variables β_{is} for all $i \in M$ and $s \in \mathbb{Z}_{\geq 0}$. The dual is

$$\begin{aligned} \max \quad & z^D = \sum_{j\in J} \alpha_j \; - \sum_{i\in M} \sum_{s\in\mathbb{Z}_{\geq 0}} \beta_{is} & & \\ \text{s.t.} \quad & \frac{\alpha_j}{\mathbb{E}[P_{ij}]} - \beta_{is} \leq w_j \left(\frac{s+\frac{1}{2}}{\mathbb{E}[P_{ij}]} + \frac{1}{2} \right) & & \text{for all } i \in M, j \in J, s \in \mathbb{Z}_{\geq 0}\,, \\ & \beta_{is} \geq 0 & & \text{for all } i \in M, s \in \mathbb{Z}_{\geq 0}\,. \end{aligned} \tag{D}$$

We are going to define a feasible solution for the dual (D) by means of a simple online algorithm for the original scheduling problem. The same type of greedy algorithm has been used before, both in deterministic and stochastic scheduling on parallel machines, e. g. in [2, 19, 20].

4 Greedy Algorithm & Analysis

Let us assume w.l.o.g. that the jobs are presented in the order $1, 2 \ldots, |J|$. On any machine i, denote by $H(j, i)$ the set of all jobs that have higher priority according to their order in non-increasing order of ratios $w_j/\mathbb{E}[P_{ij}]$, breaking ties by index. That is, $H(j, i) := \{k \in J \mid w_k/\mathbb{E}[P_{ik}] > w_j/\mathbb{E}[P_{ij}]\} \cup \{k \in J \mid k \leq j, w_k/\mathbb{E}[P_{ik}] = w_j/\mathbb{E}[P_{ij}]\}$. Also, let $L(j, i) := J \setminus H(j, i)$. Further, denote by $k \to i$ the fact that a job k has been assigned to a machine i.

Greedy Algorithm: Whenever a new job $j \in J$ is presented to the algorithm, we compute for each of the machines $i \in M$ the *instantaneous expected* *increase* if the jobs already present on each machine where to be scheduled in non-increasing order of the ratios weight over expected processing time,

$$\mathrm{EI}(j \to i) := w_j \left(\mathbb{E}[P_{ij}] + \sum_{k \to i, k<j, k \in H(j,i)} \mathbb{E}[P_{ik}] \right) \;+\; \mathbb{E}[p_{ij}] \sum_{k \to i, k<j, k \in L(j,i)} w_k \,.$$

We assign the job to one of the machines where this quantity is minimal, that is, a job is assigned to machine $i(j) := \mathrm{argmin}_{i \in M}\{\mathrm{EI}(j \to i)\}$; ties broken arbitrarily. Once all jobs have arrived and are assigned, they will be sequenced in non-increasing order of ratios weight over expected processing time, which is optimal conditioned on the given assignment [27].

Now we define the dual solution (α, β) in a similar same way as it has been done in [1]. We let

$$\alpha_j := \mathrm{EI}(j \to i(j)) \quad \text{for all } j \in J \,.$$

That is, α_j is defined as the instantaneous expected increase on the machine to which it is assigned by the greedy algorithm. Moreover, let

$$\beta_{is} := \sum_{j \in A_i(s)} w_j \,,$$

where $A_i(s)$ is defined as the total set of jobs assigned to machine i by the greedy algorithm, but restricted to those that have not yet been completed by time s if the jobs' processing times were their expected values $\mathbb{E}[P_{ij}]$. In other words, β_{is} is exactly the expected total weight of yet unfinished jobs on machine i at time s, given the assignment (and sequencing) of the greedy algorithm.

Fact 1 *The solution* $(\alpha/2, \beta/2)$ *is feasible for* (D).

Moreover, we have the following observations which follow more or less directly from the definition of the dual variables (α, β). Let us denote by ALG the total expected value achieved by the greedy algorithm.

Lemma 3. *The total expected value of the greedy algorithm is*

$$\mathsf{ALG} = \sum_{j \in J} \alpha_j = \sum_{i \in M} \sum_{s \in \mathbb{Z}_{\geq 0}} \beta_{is} \,.$$

5 Speed Augmentation & Analysis

The previous analysis of the dual feasible solution $(\alpha/2, \beta/2)$ yields a dual objective value 0 by Lemma 3, which is of little help. However following [1], we can define another dual solution which has an interpretation in the model where all machines run at faster speed $f \geq 1$, meaning that all (expected) processing times get scaled down by a factor f^{-1}. This will yield something useful.

So let us define ALG^f as the expected solution value obtained by the same greedy algorithm, only when all the machine run at speed f. Note that $\mathsf{ALG} = f\mathsf{ALG}^f$, by definition. We denote by (α^f, β^f) the exact same dual solution that was defined before, only for the new instance with faster machines. We now claim the following.

Lemma 4. *The solution*

$$(\frac{1}{2}\alpha^f, \frac{1}{2f}\beta^f)$$

is a feasible solution for the dual (D) *in the* original *(unscaled) problem instance.*

Now we conclude with the first main theorem of the paper.

Theorem 1. *The greedy algorithm is a* $(8+4\Delta)$*-competitive algorithm for online scheduling of stochastic jobs to minimize the expectation of the total weighted completion times* $\mathbb{E}[\sum_j w_j C_j]$.

Proof. We know from Corollary 1 that

$$z^D(\frac{1}{2}\alpha^f, \frac{1}{2f}\beta^f) \le z^D = z^P \le \big(1 + \frac{\Delta}{2}\big)\mathsf{OPT}\,.$$

Next, recall that $\mathsf{ALG}^f = \sum_{j\in J} \alpha_j^f = \sum_{i\in M}\sum_{s\in\mathbb{Z}_{\ge 0}} \beta_{is}^f$ by Lemma 3, and $\mathsf{ALG} = f\mathsf{ALG}^f$. The theorem now follows from evaluating the objective value of the specifically chosen dual solution $(\frac{1}{2}\alpha^f, \frac{1}{2f}\beta^f)$ for (D), as

$$z^D(\frac{1}{2}\alpha^f, \frac{1}{2f}\beta^f) = \frac{1}{2}\sum_{j\in J}\alpha_j^f - \frac{1}{2f}\sum_{i\in M}\sum_{s\in\mathbb{Z}_{\ge 0}}\beta_{is}^f = \frac{f-1}{2f}\mathsf{ALG}^f = \frac{f-1}{2f^2}\mathsf{ALG}\,.$$

Putting together this equality with the previous inequality yields a performance bound of $\frac{2f^2}{f-1}(1+\frac{\Delta}{2})$, which is minimal for $f = 2$. □

6 The Online Time Model

We now consider the online time model where jobs arrive over time. A job j arrives at time r_j. We can assume w.l.o.g. that $r_j \le r_k$ for $j < k$. In the algorithm, which is the analogue of the one used in [20] for the parallel machine setting, each job will be irrevocably assigned to a machine upon arrival.

Modified Greedy Algorithm: *1. Assignment of jobs to machines:* At time r_j, we compute for each of the machines $\mathrm{EI}(j \to i)$ exactly in the same way as it has been done for the case without release times, and assign job j to one of the machines that minimizes $\mathrm{EI}(j \to i)$. *2. Scheduling:* For the case with release dates, it is well known that (long) jobs must be delayed in order to achieve competitive algorithms [19, 20]. We do the same here, but we insert a little more forced idleness than these papers. For any job j assigned to machine i at time r_j, we modify its release date to $r'_j = \max\{2r_j, \mathbb{E}[P_{ij}]\}$. Now if a machine i falls idle at a time t, among all unfinished jobs assigned to i and with $r'_j \geq t$, we schedule the job with the highest ratio $w_j/\mathbb{E}[P_{ij}]$, by first forcing the machine to remain idle for $\mathbb{E}[P_{ij}]$ units of time, and then beginning the actual processing of job j.

The main result of this section is:

Theorem 2. *For the stochastic online scheduling problem on unrelated parallel machines with release dates, if* $\max_{i,j} \mathbb{CV}[P_{ij}]^2 \leq \Delta$*, then the Modified Greedy Algorithm is* $(144 + 72\Delta)$*-competitive.*

Proof Sketch: The full proof is a bit intricate and presented in the appendix. Here we sketch the main steps in the analysis. Defining the expected cost of the modified greedy algorithm as ALG_S and of the optimal non-anticipative policy as OPT, our goal is to prove $\mathsf{ALG}_S \leq (144 + 72\Delta)\mathsf{OPT}$.
Step 1: As in the online-list model, the core of the argument proceeds via an instance with augmented machine speeds. Given instance $\{r_j, \{P_{ij}\}_{i\in M}\}_{j\in J}$, we define a family of instances parameterized by speed-up f with release times $r_j^f = r_j$ and processing times $P_{ij}^f = P_{ij}/f$. Denote by ALG_S^f the expected cost of a variant of the modified greedy algorithm where the scheduling rule is changed to use the modified release times as $R_j^f = \max\{r_j, \mathbb{E}[P_{ij}^f]\}$. Then a time scaling argument shows

$$\mathsf{ALG}_S = 2 \cdot \mathsf{ALG}_S^2.$$

In fact, the equality is even in distribution and not just for expectations.
Step 2: For the stochastic instance $\{r_j^f, \{P_{ij}^f\}\}$, we define the deterministic instance where processing time of job j on machine i is non-stochastic and equals $\mathbb{E}[P_{ij}^f]$. Further, we begin processing the jobs as soon as they are scheduled, without the idleness. Let ALG_D^f denote the cost of our algorithm on this instance. We show,

$$\mathsf{ALG}_S^f \leq 6 \cdot \mathsf{ALG}_D^f.$$

Step 3: As in Section 3, we define the LP relaxation of the online stochastic machine scheduling problem (with optimal solution z^{S_o}). The only difference is that y_{ijs} are forced to be 0 for $s \leq r_j$. Analogously, z^{P_o} denotes the optimal solution value of the corresponding deterministic version as in Section 3, giving us:

$$z^{P_o} \leq \left(1 + \frac{\Delta}{2}\right) z^{S_o} \leq \left(1 + \frac{\Delta}{2}\right) \mathsf{OPT}.$$

Finally, we use a dual fitting argument to prove, for any $f > 1$:

$$\mathsf{ALG}_D^f \leq \frac{6f}{f-1} z^{P_o}.$$

Now substituting $f = 2$,

$$\mathsf{ALG}_S = 2\mathsf{ALG}_S^2 \leq 2 \cdot 6 \cdot \mathsf{ALG}_D^2 \leq 2 \cdot 6 \cdot 12 \cdot z^{P_o} \leq 144 \left(1 + \frac{\Delta}{2}\right) \mathsf{OPT}.$$

7 Conclusions

The main result of this paper is to show that simple, combinatorial online algorithms *can* be worst-case analyzed even for the most general of all machine scheduling models and uncertain job sizes. Further, the performance bounds are $O(\Delta)$, asymptotically the same as the identical machines setting.

Acknowledgements. This work was done while all four authors were with the Simons Institute for the Theory of Computing at UC Berkeley. The authors wish to thank the institute for the financial support and the organizers of the semester on "Algorithms & Uncertainty" for providing a very stimulating atmosphere.

References

1. S. Anand, N. Garg, and A. Kumar. Resource augmentation for weighted flow-time explained by dual fitting. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 1228–1241, 2012.
2. N. Avrahami and Y. Azar. Minimizing total flow time and total completion time with immediate dispatching. In *Proc. 15th Symp. on Parallelism in Algorithms and Architectures (SPAA 2003)*, pages 11–18. ACM, 2003.
3. S. Balseiro, D. Brown, and C. Chen. Static routing in stochastic scheduling: Performance guarantees and asymptotic optimality. Tech. Rep., 2016.
4. N. Bansal, A. Srinivasan, and O. Svensson. Lift-and-round to improve weighted completion time on unrelated machines. In *Proc. 48th Ann. ACM Symp. Theory Computing (STOC)*, pages 156–167. ACM, 2016.
5. L. Becchetti and S. Leonardi. Non-clairvoyant scheduling to minimize the average flow time on single and parallel machines. In *STOC*, pages 94–103, 2001.
6. J. Bruno, P. J. Downey, and G. Frederickson. Sequencing tasks with exponential service times to minimize the expected flowtime or makespan. *Journal of the ACM*, 28:100–113, 1981.
7. J. Correa and M. Wagner. LP-based online scheduling: From single to parallel machines. *Mathematical Programming*, 119:109–136, 2008.
8. R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5:287–326, 1979.
9. A. Gupta, S. Im, R. Krishnaswamy, B. Moseley, and K. Pruhs. Scheduling heterogeneous processors isn't as easy as you think. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 1242–1253, 2012.
10. W. Horn. Minimizing average flowtime with parallel machines. *Operations Research*, 21:846– 847, 1973.
11. E. Horowitz and S. Sahni. Exact and approximate algorithms for scheduling nonidentical processors. *Journal of the ACM*, 23(2):317–327, 1976.
12. S. Im, J. Kulkarni, K. Munagala, and K. Pruhs. Selfishmigrate: A scalable algorithm for non-clairvoyantly scheduling heterogeneous processors. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 531–540, 2014.

13. S. Im, B. Moseley, and K. Pruhs. A tutorial on amortized local competitiveness in online scheduling. *SIGACT News*, 42(2):83–97, 2011.
14. S. Im, B. Moseley, and K. Pruhs. Stochastic scheduling of heavy-tailed jobs. In *STACS*, 2015.
15. B. Kalyanasundaram and K. Pruhs. Speed is as powerful as clairvoyance. *J. ACM*, 47(4):617–643, 2000.
16. T. Kämpke. On the optimality of static priority policies in stochastic scheduling on parallel machines. *Journal of Applied Probability*, 24:430–448, 1987.
17. J. Lenstra, D. B. Shmoys, and É. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming*, 46:259–271, 1990.
18. J. Y.-T. Leung, editor. *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*. Chapman & Hall/CRC, 2004.
19. N. Megow and A. Schulz. On-line scheduling to minimize average completion time revisited. *Operations Research Letters*, 32:485–490, 2004.
20. N. Megow, M. Uetz, and T. Vredeveld. Models and algorithms for stochastic online scheduling. *Mathematics of Operations Research*, 31(3):513–525, 2006.
21. N. Megow and T. Vredeveld. A tight 2-approximation for preemptive stochastic scheduling. *Mathematics of Operations Research*, 39:1297 – 1310, 2011.
22. R. H. Möhring, F. J. Radermacher, and G. Weiss. Stochastic scheduling problems I: General strategies. *ZOR - Zeitschrift für Operations Research*, 28:193–260, 1984.
23. R. H. Möhring, F. J. Radermacher, and G. Weiss. Stochastic scheduling problems II: Set strategies. *ZOR - Zeitschrift für Operations Research*, 29:65–104, 1985.
24. R. H. Möhring, A. S. Schulz, and M. Uetz. Approximation in stochastic scheduling: The power of LP-based priority policies. *Journal of the ACM*, 46:924–942, 1999.
25. R. Motwani, S. Phillips, and E. Torng. Non-clairvoyant scheduling. *Theor. Comput. Sci.*, 130(1):17–47, 1994.
26. K. Pruhs, J. Sgall, and E. Torng. *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, chapter Online Scheduling. 2004.
27. M. H. Rothkopf. Scheduling with random service times. *Management Science*, 12:703–713, 1966.
28. A. S. Schulz. Stochastic online scheduling revisited. In B. Yang, D.-Z. Du, and C. Wang, editors, *Combinatorial Optimization and Applications*, volume 5165 of *Lecture Notes in Computer Science*, pages 448–457. Springer, 2008.
29. M. Skutella. Convex quadratic and semidefinite programming relaxations in scheduling. *Journal of the ACM*, 48:206–242, 2001.
30. M. Skutella, M. Sviridenko, and M. Uetz. Stochastic scheduling on unrelated machines. *Mathematics of Operations Research*, 41(3):851–864, 2016.
31. M. Skutella and M. Uetz. Stochastic machine scheduling with precedence constraints. *SIAM Journal on Computing*, 34:788–802, 2005.
32. M. Uetz. When greediness fails: Examples from stochastic scheduling. *Operations Research Letters*, 31:413–419, 2003.
33. R. Weber, P. Varaiya, and J. Walrand. Scheduling jobs with stochastically ordered processing times on parallel machines to minimize expected owtime. *Journal of Applied Probability*, 23:841–847, 1986.
34. G. Weiss and M. Pinedo. Scheduling tasks with exponential service times on non-identical processors to minimize various cost functions. *Journal of Applied Probability*, 17:187–202, 1980.

A Omitted Proofs

Proof of Lemma 2

Proof. Let y^P be an optimal solution to (P) and y^S be an optimal solution to (S). Clearly, y^S is a feasible solution also for (P) which is less constrained. Hence we get that

$$\begin{aligned} z^P = z^P(y^P) &\le z^P(y^S) \\ &= z^S(y^S) + \sum_{j\in J} w_j \sum_{i\in M} \sum_{s\in\mathbb{Z}_{\ge 0}} \frac{\mathbb{CV}[P_{ij}]^2}{2} y^S_{ijs} \\ &\le z^S(y^S) + \frac{\Delta}{2} H(y^S) \quad \le \quad \left(1+\frac{\Delta}{2}\right) z^S(y^S)\,. \end{aligned} \tag{7}$$

Note that the second-to-last inequality only uses the definitions of Δ and $H(\cdot)$. The last inequality holds because $H(y^S) \le z^S(y^S)$. □

Proof of Fact 1

Proof. In fact we can show that

$$\frac{\alpha_j}{\mathbb{E}[P_{ij}]} \le \beta_{is} + w_j \left(\frac{s}{\mathbb{E}[P_{ij}]} + 1 \right) \tag{8}$$

holds for all $i \in M$, $j \in J$, and $s \in \mathbb{Z}_{\ge 0}$. This implies the feasibility of $(\alpha/2, \beta/2)$ for (D). Let us fix job j and machine i, and recall that $k \to i$ describes the fact that a job k got assigned to machine i. First, by definition of α_j and by choice of $i(j)$ as the minimizer of $\mathrm{EI}(j \to i)$, for all i we have

$$\frac{\alpha_j}{\mathbb{E}[P_{ij}]} \le \frac{\mathrm{EI}(j\to i)}{\mathbb{E}[P_{ij}]} = w_j + w_j \sum_{k\to i, k<j, k\in H(j,i)} \frac{\mathbb{E}[P_{ik}]}{\mathbb{E}[P_{ij}]} + \sum_{k\to i, k<j, k\in L(j,i)} w_k\,. \tag{9}$$

We are going to argue that the right-hand-side of (9) is upper bounded by the right-hand side of (8), from which the claim follows. First observe that the term w_j cancels. Next observe that any job $k \to i$, $k \ne j$, can appear in the right-hand side of (9) at most once, either with value w_k, namely when $k \in L(j,i)$, or with value $w_j \mathbb{E}[P_{ik}]/\mathbb{E}[P_{ij}] \le w_k$ when $k \in H(j,i)$. We show that each of these values in the right-hand-side of (9) is accounted for in the right-hand side of (8), for any $s \ge 0$.

Therefore, let us fix any such job $k \to i$. First consider the case that the time s is small enough so that our job $k \to i$ is still alive at time s, so $s < \sum_{\ell\to i, \ell\in H(k,i)} \mathbb{E}[P_{i\ell}]$. Then, w_k is accounted for in the definition of β_{is}, and we are done.

So now consider the case that $s \ge \sum_{\ell\to i,\ell\in H(k,i)} \mathbb{E}[P_{i\ell}]$, which means that job k is already finished at time s. In that case, we distinguish two cases.

Case 1 is $k \in L(j,i)$: In that case, job k contributes to the right-hand side of (9) a value of w_k, but as $s \geq \sum_{\ell \to i, \ell \in H(k,i)} \mathbb{E}[P_{i\ell}]$, the term $w_j(s/\mathbb{E}[P_{ij}])$ in the right-hand side of (8) contains the term $w_j(\mathbb{E}[P_{ik}]/\mathbb{E}[P_{ij}]) \geq w_k$.

Case 2 is $k \in H(j,i)$: In that case, job k contributes to the right-hand side of (9) a value of $w_j(\mathbb{E}[P_{ik}]/\mathbb{E}[P_{ij}])$, which is exactly what is also contained in the term $w_j(s/\mathbb{E}[P_{ij}])$, because $s \geq \sum_{\ell \to i, \ell \in H(k,i)} \mathbb{E}[P_{i\ell}]$. □

Proof of Lemma 3

Proof. For the first equality, recall that α_j is the total expected instantaneous increase of the expected value that ALG achieves. Summing this over all jobs gives exactly the total expected value for ALG. For a formal proof of this, see for example [20, Lemma 4.1] for the case of parallel identical machines. That lemma and its proof can directly be extended to the case of unrelated machines. The second equality follows from the fact that the (expected) total weighted completion time of any schedule can be alternatively expressed by weighting each period of time by the total weight of yet unfinished jobs. The equality is true here, because β was defined on the basis of the same distribution of jobs over machines as given by ALG, and because each job k's weight w_k, given $k \to i$, appears in β_{is} for all s up to a job k's expected completion time, given jobs' processing times are fixed to their expected values. This is exactly what happens also in computing the expected completion times under the greedy algorithm, because it is a "fixed assignment" algorithm that assigns all jobs to machines at time 0, and sequences the jobs per machine thereafter. □

Proof of Lemma 4

Proof. By definition of $(\frac{1}{2}\alpha^f, \frac{1}{2f}\beta^f)$, to show feasibility for (D) it suffices to show

$$\frac{\alpha_j^f}{\mathbb{E}[P_{ij}]} \leq \frac{1}{f}\beta_{is}^f + w_j\left(\frac{s}{\mathbb{E}[P_{ij}]} + 1\right)$$

for all i, j, s. By definition of α we have $\alpha_j = f\alpha_j^f$. So the above is equivalent to

$$\frac{\alpha_j}{\mathbb{E}[P_{ij}]} \leq \beta_{is}^f + w_j\left(\frac{f \cdot s}{\mathbb{E}[P_{ij}]} + f\right) . \tag{10}$$

As $f \geq 1$, (10) is implied by

$$\frac{\alpha_j}{\mathbb{E}[P_{ij}]} \leq \beta_{is}^f + w_j\left(\frac{f \cdot s}{\mathbb{E}[P_{ij}]} + 1\right) . \tag{11}$$

To see that this is true, observe that we can use the exact same argument as before in the proof of the feasibility of $(\alpha/2, \beta/2)$ in Fact 1, because the left-hand side of (11) is identical to the left-hand side of (9), and the right-hand side of (11) is identical to the right-had side of (9), because $\beta_{is}^f = \beta_{i(f \cdot s)}$. □

Proof of Theorem 2

We will follow the outline described in the main body.

Step 1. Bound on ALG_S versus stochastic augmented instance. Recall that we defined the speed augmented instance with release times r_j and stochastic processing times $P_{ij}^f = P_{ij}/f$. Denote by ALG_S^f the expected solution value obtained by the greedy algorithm where if a job is assigned to machine i upon arrival, it will become available at time $R_j^f = \max\{r_j, \mathbb{E}[P_{ij}]/f\}$. By noting that for the original stochastic instance, the time at which job j is made available to machine i for scheduling is $r_j' = \max\{2r_j, \mathbb{E}[P_{ij}]\}$ and that the assignment decisions of jobs to machines depend only on the order of release, and are invariant to scaling of processing times, we see that the schedule of the speed augmented instance with $f = 2$ is *exactly* the schedule of the original stochastic instance but with the time axis compressed by a factor of 2. Therefore, $\mathsf{ALG}_S = 2\mathsf{ALG}_S^2$.

Step 2. Upper bound on ALG_S^f. For the stochastic instance $\{r_j, \{P_{ij}\}\}$, we first establish an upper bound on the expected completion time of a job j if it is assigned to machine i under the modified greedy algorithm, and the scheduling rule is changed to use the modified release times as $R_{ij} = \max\{r_j, \mathbb{E}[P_{ij}]\}$.

Lemma 5. *For the stochastic instance $\{r_j, \{P_{ij}\}\}$, if a job j is assigned to machine i under the modified greedy algorithm, its expected competition time is bounded by*

$$\mathbb{E}[C_j | j \to i] \leq 4R_{ij} + 2 \sum_{k \to i, k \in H(j,i)} \mathbb{E}[P_{ik}].$$

Proof. With the modified release times, job j becomes available on machine i at time R_{ij}. We use the random variable X to denote the remaining processing time of a job l being processed at time R_{ij}, if any such job exists. If machine i is idle at that time, X has value 0. Note that job j will not be started until job l and any available job with higher priority are completed. Thus we can bound the expected start time of job j by

$$\mathbb{E}[S_j | j \to i] \leq R_{ij} + \mathbb{E}[X] + 2 \sum_{k \to i, k \in H(j,i) \setminus \{j\}} \mathbb{E}[P_{ik}] + \mathbb{E}[P_{ij}]\,. \tag{12}$$

To bound $\mathbb{E}[X]$, we define a sequence of random variables $Y_1, Y_2, \cdots$ where Y_k measures the time interval between the completion time of the $(k-1)$st job and that of the kth job completed on machine i. Let I_k denote the idle time of machine i within this interval. Define A_k to be the sum of the expected processing time of the kth job completed and I_k. Under the modified greedy algorithm,

$$Y_k \geq A_k \;\; w.p.\ 1, \text{ and } \mathbb{E}[Y_k | Y_1, \ldots, Y_{k-1}] \leq 2A_k.$$

Define the $R_{ij}-$stopping time τ with respect to the sequence as

$$\tau := \min\{n' : Y_1 + Y_2 + \cdots + Y_{n'} \geq R_{ij}\}.$$

By Lemma 9 (see Appendix B), we have

$$R_{ij} + \mathbb{E}[X] = \mathbb{E}[Y_1 + Y_2 + \cdots + Y_\tau] \le 4R_{ij}.$$

Therefore,

$$\mathbb{E}[C_j | j \to i] = \mathbb{E}[S_j] + \mathbb{E}[P_{ij}] \le 4R_{ij} + 2 \sum_{k \to i, k \in H(j,i)} \mathbb{E}[P_{ik}].$$

□

Now we can derive an upper bound on the cost ALG_S^f of the modified greedy algorithm for the stochastic instance $\{r_j^f, \{P_{ij}^f\}\}$ in terms of the cost ALG_D^f for the corresponding deterministic instance.

Lemma 6.

$$\mathsf{ALG}_S^f \le 6\mathsf{ALG}_D^f.$$

Proof. Let C_{Sj}^f denote the completion time of job j for the stochastic instance. From Lemma 5, we know that

$$\begin{aligned}
\mathsf{ALG}_S^f = \mathbb{E}[\sum_j w_j C_{Sj}^f] &= \sum_i \sum_{j:j\to i} w_j \mathbb{E}[C_{Sj}^f | j \to i] \\
&\le \sum_i \sum_{j:j\to i} \left[4w_j R_{ij}^f + 2w_j \sum_{k\to i, k\in H(j,i)} \mathbb{E}[P_{ik}] \right] \\
&\le \sum_i \left[4 \sum_{j:j\to i} w_j (r_j^f + \mathbb{E}[P_{ij}^f]) + 2 \sum_{j:j\to i} w_j \sum_{k\to i, k\in H(j,i)} \mathbb{E}[P_{ik}] \right],
\end{aligned}$$

where the last inequality follows by the definition of the modified release dates $R_{ij}^f = \max\{r_j^f, \mathbb{E}[P_{ij}^f]\}$.

Note that the assignment of jobs to machines is independent of the release times $\{r_j^f\}$ and the realization of processing times. Hence job j is assigned to the same machine i for the stochastic instance $\{r_j^f, \{P_{ij}^f\}\}$ and deterministic instance $\{r_j^f, \{\mathbb{E}[P_{ij}^f]\}\}$. We denote by C_{Dj}^f the completion time of job j for the deterministic instance. Hence for each machine i, $\sum_{j:j\to i} w_j(r_j^f + \mathbb{E}[P_{ij}^f]) \le \sum_{j:j\to i} w_j C_{Dj}^f$. Moreover,

$$\sum_{j:j\to i} w_j \sum_{k\to i, k\in H(j,i)} \mathbb{E}[P_{ik}] \le \sum_{j:j\to i} w_j C_{Dj}^f \,,$$

as the left-hand-side is simply the value of the optimal solution on machine i without considering release times [27], which is clearly a lower bound. Therefore,

$$\mathsf{ALG}_S^f \le 6 \sum_i \sum_{j:j\to i} w_j C_{Dj}^f = 6\mathsf{ALG}_D^f \,.$$

□

Step 3.1 LP Relaxation Analogous to (S), we will define the LP relaxation for the online stochastic scheduling instance. We omit writing out the LP relaxation in detail as it it exactly the same as (S), except that the variables y_{ijs} are defined only for $s \geq r_j$. Let us call this LP "S_o" and its optimal solution value z^{S_o}. Similarly, analogous to (P) we can write the LP relaxation for the online deterministic version by dropping the $\mathbb{CV}[P_{ij}]^2$ terms from the objective function. Call this deterministic LP relaxation "P_o" with optimal solution value z^{P_o}. Lemma 2 and Corollary 1 apply to z^{P_o} and z^{S_o} in exactly the same way. That is,

$$z^{P_o} \leq \big(1 + \frac{\Delta}{2}\big) z^{S_o} \leq \big(1 + \frac{\Delta}{2}\big)\mathsf{OPT}\,.$$

Step 3.2. Lower bound on z^{P_o}. To lower bound the cost of the deterministic Primal LP relaxation P_o, we will define a feasible solution to its dual LP, which is:

$$\begin{aligned} \max \quad & z^{D_o} = \sum_{j\in J} \alpha_j \; - \sum_{i\in M} \sum_{s\in\mathbb{Z}_{\geq 0}} \beta_{is} \\ \text{s.t.} \quad & \frac{\alpha_j}{\mathbb{E}[P_{ij}]} - \beta_{is} \leq w_j \left(\frac{s+\frac{1}{2}}{\mathbb{E}[P_{ij}]} + \frac{1}{2} \right) \quad \text{for all } i \in M, j \in J, s \in \mathbb{Z}_{\geq r_j}\,, \\ & \beta_{is} \geq 0 \qquad\qquad\qquad\qquad\qquad\quad \text{for all } i \in M, s \in \mathbb{Z}_{\geq 0}\,. \end{aligned} \tag{D_o}$$

To define a feasible solution for (D_o), we consider the modified greedy algorithm for the speed augmented instance with speed up f. The release times of the speed augmented instance are unchanged: $r_j^f = r_j$, and the processing times (deterministic) are a factor of f smaller: $P_{ij}^f = \mathbb{E}[P_{ij}]/f$. To construct the dual solutions, we will use the schedule obtained by a variant of modified greedy run on the speed augmented instance where the machine assignment decisions are based on the values $\mathrm{EI}(j \to i)$, but when scheduling the jobs assigned to a machine i, we consider the modified release time as $R_{ij}^f = \max\{r_j^f, P_{ij}^f\}$. Note that the machine assignments for the speed augmented instance are identical to the assignments made by the Modified Greedy Algorithm for the original instance, since they do not depend on release times and are invariant under scaling all processing times. Further, note that the cost of the just defined variant of the modified greedy algorithm on the deterministic instance is precisely ALG_D^f.

For job $j \in J$, define

$$\alpha_j^f := \min_i \left[2w_j(r_j^f + P_{ij}^f) + w_j \sum_{k\to i, k\leq j, k\in H(j,i)} P_{ik}^f \; + \; P_{ij}^f \sum_{k\to i, k<j, k\in L(j,i)} w_k \right].$$

For a machine i and time s, we denote by $A_i^f(s)$ the total set of jobs assigned to machine i by the modified greedy algorithm on the augmented deterministic instance, but restricted to those jobs that have not been completed by time s

(including those that are assigned but not available according to release times R^f_{ij}). We define β^f_{is} as the total weight of jobs in $A^f_i(s)$, i.e.,

$$\beta^f_{is} := \sum_{j \in A^f_i(s)} w_j.$$

Lemma 7. *The values* $(\{\alpha_j\}, \{\beta_{is}\}) := \left(\{\frac{\alpha^f_j}{6}\}, \{\frac{\beta^f_{is}}{6f}\}\right)$ *are feasible for* (D_o).

Proof. Fix job j and machine i. We need to show that

$$\frac{f\alpha^f_j}{\mathbb{E}[P_{ij}]} \leq \beta^f_{is} + 6fw_j\left(\frac{s+\frac{1}{2}}{\mathbb{E}[P_{ij}]} + \frac{1}{2}\right)$$

holds for all $i \in M$, $j \in J$, and $s \geq r_j$. By the definition of α^f_j, for any machine i, we have

$$\begin{aligned}\alpha^f_j \leq & 2w_j(r^f_{ij} + P^f_{ij}) + w_j \sum_{k\to i, k \leq j, k \in H(j,i)} P^f_{ik} \; + \; P^f_{ij} \sum_{k \to i, k<j, k \in L(j,i)} w_k \\ = & 2w_j\left(r_j + \frac{\mathbb{E}[P_{ij}]}{f}\right) + w_j \sum_{k\to i, k\leq j, k\in H(j,i)} \frac{\mathbb{E}[P_{ik}]}{f} \; + \; \frac{\mathbb{E}[P_{ij}]}{f} \sum_{k\to i, k<j, k\in L(j,i)} w_k\end{aligned}$$

Equivalently,

$$\begin{aligned}\frac{f\alpha^f_j}{\mathbb{E}[P_{ij}]} \leq & \frac{2w_j(fr_j + \mathbb{E}[P_{ij}])}{\mathbb{E}[P_{ij}]} + w_j \sum_{k\to i, k\leq j, k\in H(j,i)} \frac{\mathbb{E}[P_{ik}]}{\mathbb{E}[P_{ij}]} \; + \sum_{k\to i, k<j, k\in L(j,i)} w_k \\ \leq & w_j\left(2\frac{fs}{\mathbb{E}[P_{ij}]} + 3\right) + w_j \sum_{k\to i, k<j, k\in H(j,i)} \frac{\mathbb{E}[P_{ik}]}{\mathbb{E}[P_{ij}]} \; + \sum_{k\to i, k<j, k\in L(j,i)} w_k \, ,\end{aligned}$$

where the last inequality follows by the fact that $r_j \leq s$. Therefore, it suffices to show that

$$w_j \sum_{k\to i, k<j, k\in H(j,i)} \frac{\mathbb{E}[P_{ik}]}{\mathbb{E}[P_{ij}]} \; + \sum_{k\to i, k<j, k\in L(j,i)} w_k \; \leq \beta^f_{is} + 3fw_j \frac{s}{\mathbb{E}[P_{ij}]}. \quad (13)$$

Let $D^f_{ij}(s)$ denote the set of jobs $k < j$ assigned to machine i and completed by time s, and $U^f_{ij}(s)$ be the set of jobs $k < j$ assigned to machine i and still unfinished (alive) at time s (including those are assigned but not available according to modified release times R^f_{ij}). Observe that $U^f_{ij}(s) \subset A^f_i(s)$. Hence by the definition of β^f_{is},

$$\sum_{k \in U^f_{ij}(s)} w_k \leq \sum_{k \in A^f_i(s)} w_k = \beta^f_{is}. \quad (14)$$

Here

$$\sum_{k \in D_{ij}^f(s)} \frac{\mathbb{E}[P_{ik}]}{f} \le s. \tag{15}$$

Note that if $k \in H(j,i)$, $w_j \frac{\mathbb{E}[P_{ik}]}{\mathbb{E}[P_{ij}]} \le w_k$, and if $k \in L(j,i)$, $w_j \frac{\mathbb{E}[P_{ik}]}{\mathbb{E}[P_{ij}]} > w_k$. Then we can upper bound the left-hand side (LHS) of (13) as follows:

$$\begin{aligned}
\text{LHS of (13)} &= \sum_{k \in H(j,i) \cap D_{ij}^f(s)} w_j \frac{\mathbb{E}[P_{ik}]}{\mathbb{E}[P_{ij}]} + \sum_{k \in L(j,i) \cap D_{ij}^f(s)} w_k \\
&\quad + \sum_{k \in H(j,i) \cap U_{ij}^f(s)} w_j \frac{\mathbb{E}[P_{ik}]}{\mathbb{E}[P_{ij}]} + \sum_{k \in L(j,i) \cap U_{ij}^f(s)} w_k \\
&\le \sum_{k \in H(j,i) \cap D_{ij}^f(s)} w_j \frac{\mathbb{E}[P_{ik}]}{\mathbb{E}[P_{ij}]} + \sum_{k \in L(j,i) \cap D_{ij}^f(s)} w_j \frac{\mathbb{E}[P_{ik}]}{\mathbb{E}[P_{ij}]} \\
&\quad + \sum_{k \in H(j,i) \cap U_{ij}^f(s)} w_k + \sum_{k \in L(j,i) \cap U_{ij}^f(s)} w_k \\
&= \frac{w_j}{\mathbb{E}[P_{ij}]} \sum_{k \in D_{ij}^f(s)} \mathbb{E}[P_{ik}] + \sum_{k \in U_{ij}^f(s)} w_k \\
&\le \frac{w_j}{\mathbb{E}[P_{ij}]} s + \beta_{is}^f,
\end{aligned}$$

where the last inequality follows from (14)-(15). □

Corollary 2. *The optimal solution of the deterministic LP relaxation (P_o) is bounded by:*

$$z^{P_o} \ge \frac{1}{6} \left(\sum_{j \in J} \alpha_j^f - \frac{1}{f} \sum_{i \in M} \sum_{s} \beta_{is}^f \right).$$

Step 3.3 Upper bound on ALG_D^f . Finally, to complete the proof, we prove that the dual variables (α^f, β^f) bound the cost of the algorithm for the speed augmented deterministic instance, ALG_D^f. This in turn allows us to upper bound ALG_D^f in terms of the optimal value of the deterministic LP relaxation z^{P_o}.

Lemma 8. *The total weighted completion time of the modified greedy algorithm on the deterministic instance with f-speedup satisfies*

$$\mathsf{ALG}_D^f \le \sum_{j \in J} \alpha_j^f, \qquad \mathsf{ALG}_D^f = \sum_{i \in M} \sum_{s \in \mathbb{Z}_{\ge 0}} \beta_{is}^f .$$

Combined with Corollary 2, the above gives:

$$\mathsf{ALG}_D^f \le \frac{6f}{f-1} z^{P_o}.$$

Proof. For each job j, let i_j denote the machine to which it is assigned. By the same argument as that for (12), we can obtain the following upper bound on the start time of job j :

$$S_j \leq R^f_{i_j j} + X + \sum_{k \to i_j, k \in H(j,i_j) \setminus \{j\}} \mathbb{E}[P^f_{i_j k}]\,,$$

where X is the remaining processing time of a job l in process at time $R^f_{i_j j}$, if any such job exists; otherwise X has value 0. Next observe that $X \leq R^f_{i_j j}$: This clearly holds if machine i is idle at $R^f_{i_j j}$. Assume that job l is being processed at $R^f_{i_j j}$. Then it must be true that $\max\{r_j, \mathbb{E}[P^f_{i_j l}]\} = R^f_{i_j l} \leq R^f_{i_j j}$. Hence $X \leq \mathbb{E}[P^f_{i_j l}] \leq R^f_{i_j j}$. Therefore, we have

$$\mathsf{ALG}^f_D = \sum_j w_j C^f_j \leq 2 \sum_j w_j R^f_{i_j j} + \sum_j w_j \sum_{k \to i_j, k \in H(j,i_j)} P^f_{i_j k}. \tag{16}$$

By applying the following index rearrangement,

$$\sum_j w_j \sum_{\substack{k \to i_j \\ k \in H(j,i_j) \\ k > j}} P^f_{i_j k} = \sum_j P^f_{i_j j} \sum_{\substack{k \to i_j \\ k \in L(j,i_j) \\ k < j}} w_k,$$

we can rewrite the second part of the right hand side of (16) as

$$\sum_j w_j \sum_{\substack{k \to i_j \\ k \in H(j,i_j)}} P^f_{i_j k} = \sum_j w_j \sum_{\substack{k \to i_j \\ k \in H(j,i_j) \\ k \leq j}} P^f_{i_j k} + \sum_j P^f_{i_j j} \sum_{\substack{k \to i_j \\ k \in L(j,i_j) \\ k < j}} w_k.$$

We thus obtain

$$\begin{aligned} \mathsf{ALG}^f_D &\leq \sum_j \left(2 w_j (r^f_j + P^f_{i_j j}) + w_j \sum_{\substack{k \to i_j \\ k \in H(j,i_j) \\ k \leq j}} P^f_{i_j k} + P^f_{i_j j} \sum_{\substack{k \to i_j \\ k \in L(j,i_j) \\ k < j}} w_k \right) \\ &= \sum_j \left(2 w_j r^f_j + \frac{1}{f} \min_i \mathrm{EI}(j \to i) \right) \\ &= \sum_j \alpha^f_j\,, \end{aligned}$$

where the first equality follows by the fact that i_j minimizes $\mathrm{EI}(j \to i)$ (note that the machine assignment is the same under the speed augmented instance as in the original stochastic instance) under the modified greedy algorithm, and the second equality comes from the definition of α^f_j.

The proof of $\mathsf{ALG}^f_D = \sum_i \sum_s \beta^f_{is}$ is the same as that of Lemma 3. □

B Auxiliary Lemmas

Lemma 9. *Let $X_1, X_2, \ldots,$ be a sequence of random variables, with X_i adapted to the filtration $\mathcal{F}_{i-1} = \sigma(X_1, \ldots, X_{i-1})$ for all $i \geq 1$. Further, let $A_1, A_2, \ldots$ be another sequence, with A_i adapted to $\mathcal{F}_{i-1}$ satisfying*

1. *$0 \leq A_i \leq T$ almost surely,*
2. *$X_i \geq \alpha A_i$ almost surely, and $\mathbb{E}[X_i] \leq (1+\alpha)A_i$ for some $\alpha > 0$.*

Define the T-stopping time of the $X_1, X_2, \ldots$ sequence as:

$$\tau := \min\{n : X_1 + \cdots + X_n \geq T\} \tag{17}$$

Under the assumption that stopping time τ satisfies $\mathbb{E}[\tau] < \infty$, we have

$$\mathbb{E}\left[\sum_{i=1}^{\tau} X_i\right] \leq \frac{(1+\alpha)^2}{\alpha} T. \tag{18}$$

In particular, choosing $\alpha = 1$, so that $\mathbb{E}[X_i | \mathcal{F}_{i-1}] \leq 2A_i$ and $X_i \geq A_i$, we have

$$\mathbb{E}\left[\sum_{i=1}^{\tau} X_i\right] \leq 4T.$$

Proof. The lemma is a straightforward consequence of the Optional Stopping Theorem. We first note that since $\mathbb{E}[X_i|\mathcal{F}_{i-1}] \leq (1+\alpha)A_i$, the sequence:

$$M_n = \sum_{i=0}^{n} (X_i - (1+\alpha)A_i)$$

defines a supermartingale with $M_0 = 0$. Under the assumption that $\mathbb{E}[\tau] < \infty$, Optional Stopping Theorem gives:

$$\mathbb{E}[M_\tau] \leq M_0 = 0$$

Therefore,

$$\begin{aligned}
\mathbb{E}\left[\sum_{i=1}^{\tau} X_i\right] &\leq \mathbb{E}\left[(1+\alpha)\sum_{i=1}^{\tau} A_i\right] \\
&= (1+\alpha)\mathbb{E}[A_\tau] + (1+\alpha)\mathbb{E}\left[\sum_{i=1}^{\tau-1} A_i\right] \\
&\leq (1+\alpha)T + (1+\alpha)\mathbb{E}\left[\sum_{i=1}^{\tau-1} A_i\right] \\
&\leq (1+\alpha)T + (1+\alpha)\frac{T}{\alpha} \\
&= \frac{(1+\alpha)^2}{\alpha} T
\end{aligned}$$

Where the first inequality follows from $A_i \leq T$, and second inequality follows from the observations: $A_i \leq \frac{1}{\alpha} X_i$ almost surely, and $\sum_{i=1}^{\tau-1} X_i \leq T$.