Models and Algorithms for Stochastic Online Scheduling [1]

Nicole Megow
Technische Universität Berlin, Institut für Mathematik, Strasse des 17. Juni 136, 10623 Berlin, Germany.
email: nmegow@math.tu-berlin.de

Marc Uetz
Maastricht University, Department of Quantitative Economics, P.O. Box 616, 6200 MD Maastricht, The Netherlands.
email: m.uetz@ke.unimaas.nl

Tjark Vredeveld[2]
Maastricht University, Department of Quantitative Economics, P.O. Box 616, 6200 MD Maastricht, The Netherlands.
email: t.vredeveld@ke.unimaas.nl

We consider a model for scheduling under uncertainty. In this model, we combine the main characteristics of online and stochastic scheduling in a simple and natural way. Job processing times are assumed to be stochastic, but in contrast to traditional stochastic scheduling models, we assume that jobs arrive online, and there is no knowledge about the jobs that will arrive in the future. The model incorporates both, stochastic scheduling and online scheduling as a special case. The particular setting we consider is non-preemptive parallel machine scheduling, with the objective to minimize the total weighted completion times of jobs. We analyze simple, combinatorial online scheduling policies for that model, and derive performance guarantees that match the currently best known performance guarantees for stochastic and online parallel machine scheduling. For processing times that follow NBUE distributions, we improve upon previously best known performance bounds from stochastic scheduling, even though we consider a more general setting.

Key words: scheduling; stochastic; online; total weighted completion time; approximation

MSC2000 Subject Classification: Primary: 90B36 ; Secondary: 68W40, 68W25, 68M20

OR/MS subject classification: Primary: Production/scheduling ; Secondary: approximation/heuristic

---

1. Introduction. Scheduling on identical parallel machines to minimize the total weighted completion time of jobs, $\mathrm{P} \,|\, | \sum w_j C_j$ in the three-field notation of Graham et al. [12], is one of the classical problems in combinatorial optimization. The problem plays a role whenever many jobs must be processed on a limited number of machines or processors, with applications in manufacturing, parallel computing [5] or compiler optimization [6]. The literature has witnessed many papers on this problem, as well as its variant where the jobs have individual release dates before which they must not be processed, $\mathrm{P} \,|\, r_j \,|\, \sum w_j C_j$. In the off-line (deterministic) setting, where the set of jobs and their characteristics are known in advance, the complexity status of both problems is solved; both are strongly NP-hard [17], and they admit a polynomial time approximation scheme [1, 27].

In order to cope with scenarios where there is uncertainty about the future, there are two major frameworks in the theory of scheduling, one is *stochastic scheduling*, the other is *online scheduling*. In stochastic scheduling the population of jobs is assumed to be known beforehand, but in contrast to deterministic models, the processing times of jobs are random variables. The actual processing times become known only upon completion of the jobs. The respective random variables, or at least their first moments, are assumed to be known beforehand. In online scheduling, the assumption is that the instance is presented to the scheduler only piecewise. Jobs are either arriving one-by-one (in the online-list model), or over time (in the online-time model) [22]. The actual processing times are usually disclosed upon arrival of a job, and decisions must be made without any knowledge of the jobs to come.

We consider a model that generalizes both, stochastic scheduling and online scheduling. Like in online scheduling, we assume that the instance is presented to the scheduler piecewise, and nothing is known about jobs that might arrive in the future. Once a job arrives, like in stochastic scheduling, we assume that its expected processing time is disclosed, but the actual processing time remains unknown until the

[1] Research partially supported by the DFG Research Center MATHEON *Mathematics for key technologies*. An extended abstract with parts of this work appeared in the Proc. of the 2nd Workshop on Approximation and Online Algorithms [19].

[2] Parts of this work were done while the author was with the Konrad-Zuse-Zentrum für Informationstechnik Berlin, Germany.

job completes. Before we discuss the model and related work in more detail, let us fix some basic notation and definitions.

Model and definitions. Given is a set $J = \{1, \ldots, n\}$ of jobs, with nonnegative weights w_j, $j \in J$. In the model with release dates, r_j denotes the earliest point in time when job j can be started. Each job must be processed non-preemptively on any of the m identical machines, and each machine can only handle one job at a time. The goal is to find a schedule that minimizes the total weighted completion time $\sum_j w_j C_j$, where C_j denotes the completion time of job j. By P_j we denote the random variable for the processing time of job j, by $\mathbb{E}[P_j]$ its expected processing time, and by p_j a particular realization of P_j. The processing time distributions P_j are assumed to be independent. We assume that the jobs are arriving over time upon their respective release dates r_j in the order $1, \ldots, n$. Therefore, we can assume w.l.o.g. that $r_j \leq r_k$ for $j < k$. Notice that the number of jobs n is not known in advance. When a job arrives at time r_j, the scheduler is informed about its weight w_j and its expected processing time $\mathbb{E}[P_j]$.

The goal is to find a *stochastic online scheduling policy* that minimizes the expected value of the weighted completion times of jobs, $\mathbb{E}[\sum w_j C_j]$. Our definition of a stochastic online scheduling policy extends the traditional definition of stochastic scheduling policies by Möhring et al. [20] to the setting where jobs arrive online. A scheduling policy specifies *actions* at decision times t. An action is a set of jobs that is started at time t, and a next decision time $t' > t$ at which the next action is taken, unless some job is released or ends at time $t'' < t'$. In that case, t'' becomes the next decision time. In order to decide, the policy may utilize the complete information contained in the partial schedule up to time t, as well as information about unscheduled jobs that have arrived before t. However, a policy is required to be *non-anticipatory*, thus at any time, it must neither utilize any information about jobs that will be released in the future, nor must it utilize the actual processing times of jobs that are scheduled (or unscheduled) but not yet completed. An *optimal* scheduling policy is defined as a non-anticipatory scheduling policy that minimizes the objective function value in expectation. An optimal scheduling policy generally fails to yield an optimal solution for all realizations of the processing times; this because it is non-anticipatory.

For any realization $p = (p_1, \ldots, p_n)$ of processing times of the jobs, a scheduling policy yields a feasible m-machine schedule. For a given policy, denoted by Π, let $S_j^{\Pi}(p)$ and $C_j^{\Pi}(p)$ the start and completion times of job j for a given realization p, and let $S_j^{\Pi}(P)$ and $C_j^{\Pi}(P)$ denote the associated random variables. Unless there is danger of ambiguity, we also write S_j and C_j, for short. We let

$$\mathbb{E}[\Pi(P)] = \mathbb{E}\left[\sum\nolimits_j w_j \, C_j^{\Pi}(P)\right] = \sum\nolimits_j w_j \, \mathbb{E}\left[C_j^{\Pi}(P)\right]$$

denote the expected performance of a scheduling policy Π. Let us denote the above defined model as SoS, for *stochastic online scheduling*.

Generalizing the definitions by Möhring et al. in [21] for traditional stochastic scheduling, we define the performance guarantee of a stochastic online scheduling policy as follows.

Definition 1.1 *A stochastic online scheduling policy* (SoS *policy*) Π *is a* ρ–approximation *if, for some* $\rho \geq 1$*, and all instances of the given problem,*

$$\mathbb{E}[\Pi(P)] \leq \rho \, \mathbb{E}[\mathrm{OPT}(P)] .$$

Here, OPT *denotes an optimal stochastic scheduling policy on the given instance, assuming a priori knowledge of the set of jobs J, their weights w_j, release dates r_j, and processing time distributions P_j. The constant ρ is called the performance guarantee of policy* Π.

Notice that the stochastic online scheduling policy Π in this definition does *not* have a priori knowledge of the set of jobs J, their weights w_j, release dates r_j, and processing time distributions P_j. Policy Π is an online policy, and only learns about the existence of any job j upon its release date r_j. Hence, the policy has to compete with an adversary that knows the online sequence of jobs in advance. However, with respect to the processing times P_j of the jobs, the adversary is just as powerful as the policy Π itself, since it does not foresee their actual realizations p_j either.

Probably the best known scheduling policy in stochastic scheduling is the WSEPT rule. Its online version will also play a prominent role in this paper, and is defined next. To this end, call a job j *available* at a given time t if it has not yet been scheduled, and if its release date has passed, that is, if $r_j \leq t$.

Definition 1.2 (Online WSEPT, weighted shortest expected processing time first) *At any point in time when a machine is idle, among all jobs that are available, schedule the job with the highest ratio of weight over expected processing time, $w_j/\mathbb{E}[P_j]$.*

Whenever release dates are absent, it reduces to the traditional WSEPT rule known from stochastic scheduling, and the jobs appear in the schedule in order of non-increasing ratios $w_j/\mathbb{E}[P_j]$. For unit weights, it reduces to SEPT, shortest expected processing time first. For single machine (stochastic) scheduling without release dates, $1 \,|\, \,|\, \sum w_j C_j$, the WSEPT rule is optimal; this follows by a simple job interchange argument [23, 29].

Related work. Stochastic machine scheduling models have been addressed mainly since the 1980s [9]. In the traditional stochastic setting, Weiss [33, 34] analyzes the performance of the WSEPT rule. He derives additive performance bounds for the stochastic parallel machine model without release dates, $\mathrm{P}|\,|\mathbb{E}[\sum w_j C_j]$. His bounds yield asymptotic optimality of the WSEPT rule for a certain class of processing time distributions. More recently, approximation algorithms for stochastic machine scheduling have been derived by Möhring et al. [21] and Skutella and Uetz [26]. In these papers, the expected performance of the WSEPT rule, and linear programming based stochastic scheduling policies, are compared against the expected performance of an optimal stochastic scheduling policy. The results are constant-factor approximations for models without or with release dates [21], and also with precedence constraints [26]. However, these papers do not address the situation where jobs arrive online.

In contrast to traditional stochastic scheduling, in online scheduling it is assumed that nothing is known about jobs that are about to arrive in the future. However, once a job becomes known, its weight w_j and its actual processing time p_j are disclosed. The quality of online algorithms is assessed by their competitive ratio [15, 28]. An algorithm is called *ρ-competitive* if, for any instance, a solution is achieved with value not worse than ρ times the value of an optimal off-line solution. In the *online-time* model jobs become known upon their release dates r_j. In the *online-list* model, jobs are presented one-by-one, all at time 0. Upon presentation, a job has to be scheduled immediately before the next job can be seen (at some time that is feasible with respect to release dates and the already scheduled jobs). We omit further details and refer to Borodin and El-Yaniv [3] or Pruhs et al. [22].

In the online-list model, Fiat and Woeginger [11] show that the single machine problem $1 \,|\, r_j \,|\, \sum C_j$ does not allow a deterministic or randomized online algorithm with a competitive ratio of $\log n$. In the online-time model, Anderson and Potts [2] provide a 2-competitive online algorithm for the same problem. This result is best possible, since Hoogeveen and Vestjens [14] prove a lower bound of 2 on the competitive ratio of any deterministic online algorithm. For settings with parallel machines, Vestjens [32] proves a lower bound of 1.309 for the competitive ratio of any deterministic online algorithm for $\mathrm{P} \,|\, r_j \,|\, \sum w_j C_j$, even for unit weights. The currently best known deterministic algorithm for $\mathrm{P} \,|\, r_j \,|\, \sum w_j C_j$ is 2.62-competitive, proposed by Correa and Wagner [8]. The currently best known randomized algorithm for the problem has an expected competitive ratio of 2; see Schulz and Skutella [25].

A model that combines features of stochastic and online scheduling has also been considered by Chou et al. [7]. They prove asymptotic optimality of the online WSEPT rule for the single machine problem $1 \,|\, r_j \,|\, \sum w_j C_j$, assuming that the weights w_j and processing time p_j can be bounded by constants. The definition of the adversary in their paper coincides with our definition. Hence, asymptotic optimality means that the ratio of the expected performance of the WSEPT rule over the expected performance of an optimal stochastic scheduling policy tends to 1 as the number of jobs tends to infinity.

A different type of analysis for stochastic scheduling has been proposed by Steger et al. [24, 30]. Both papers address the parallel machine model without release dates. They compare the performance of the (W)SEPT rule to the optimal solution *per realization*, and take the expectation of this ratio on the basis of the given processing time distributions. Their analysis is thus different from the traditional stochastic scheduling model; in particular is it not based upon an comparison to the optimal stochastic scheduling policy. Although the underlying adversary is stronger than in traditional stochastic scheduling, they derive constant bounds on what they call the *expected competitive ratio*.

Results and methodology. We propose simple, combinatorial online scheduling policies for stochastic online scheduling models on parallel machines with and without release dates, and derive constant performance bounds for these policies.

For identical parallel machine scheduling without release dates, $\mathrm{P}\,|\,|\,\mathbb{E}\left[\sum w_j\,C_j\right]$, the performance guarantee is

$$\rho = 1 + \frac{(m-1)(\Delta+1)}{2m}\,.$$

Here, Δ is an upper bound on the squared coefficients of variation of the processing time distributions P_j, that is,

$$\mathrm{CV}\left[P_j\right]^2 = \mathrm{Var}\left[P_j\right]/\mathbb{E}\left[P_j\right]^2 \leq \Delta \quad \text{for all jobs } j\,.$$

This performance guarantee matches the previously best known performance guarantee of Möhring et al. [21]; they obtain the same bound for the performance of the WSEPT rule in the traditional stochastic scheduling model. However, we derive this bound for a model other than traditional stochastic scheduling. More precisely, we consider a stochastic online model where the jobs are presented to the scheduler sequentially, and the scheduler must immediately and irrevocably assign jobs to machines, without knowledge of the jobs to come. Therefore we also speak of *fixed-assignment* policies. Once the jobs are assigned to the machines, the jobs on each machine can be sequenced in any order. We thus show that there exists a fixed-assignment policy that achieves the same performance guarantee as the above mentioned bound for the WSEPT rule. In this context, notice that the traditional WSEPT rule is not a fixed-assignment policy, since the assignment of jobs to machines depends on the actual realizations of the processing times.

For the model with release dates, $\mathrm{P}\,|\,r_j\,|\sum w_j\,C_j$, we prove a slightly more complicated performance guarantee that is valid for a class of processing time distributions that we call δ-NBUE, generalizing the well known class of NBUE distributions (new better than used in expectation).

Definition 1.3 (δ-NBUE) *A non-negative random variable X is δ-NBUE if, for $\delta \geq 1$, $\mathbb{E}\left[X - t\,|\,X > t\,\right] \leq \delta\,\mathbb{E}\left[X\right]$ for all $t \geq 0$.*

For identical parallel machine scheduling with release dates $\mathrm{P}\,|\,r_j\,|\sum w_j\,C_j$, and δ-NBUE distributions, we obtain a performance guarantee of

$$\rho = 1 + \max\left\{1 + \frac{\delta}{\alpha}\ ,\ \alpha + \delta + \frac{(m-1)(\Delta+1)}{2m}\right\}\,.$$

Here, $\alpha > 0$ is an arbitrary parameter, and again, Δ is an upper bound on the squared coefficients of variation $\mathrm{Var}[P_j]/\mathbb{E}\left[P_j\right]^2$ of the processing time distributions P_j. In particular, since we show in the appendix that $\mathrm{Var}[P_j]/\mathbb{E}\left[P_j\right]^2 \leq 2\delta - 1$ for δ-NBUE distributions, we get

$$\rho \leq 1 + \max\left\{1 + \delta/\alpha\ ,\ \alpha + \delta(2m-1)/m\right\}\,.$$

Optimizing for α yields that $\rho < 3/2 + (1 - 1/(2m))\delta + \sqrt{1 + 4\delta^2}/2$. For for ordinary NBUE distributions, where $\delta = 1$, we thus obtain a performance bound strictly less than $(\sqrt{5}+5)/2 - 1/(2m) \approx 3.62 - 1/(2m)$. Thereby, we improve upon the previously best known performance guarantee of $4 - 1/m$ for traditional stochastic scheduling, which was derived for an LP based list scheduling policy [21]. Again, this improved bound holds even though we consider an online model in which the jobs arrive over time, and the scheduler does not know anything about the jobs that are about to arrive in the future. Moreover, for deterministic processing times, where $\Delta = 0$ and $\delta = 1$, we get $\rho = 2 + \max\{1/\alpha\ ,\ \alpha + (m-1)/(2m)\}$; optimizing for α yields $\rho \approx 3.28$, matching the bound from deterministic online scheduling by Megow and Schulz [18].

For both models, our results are in fact achieved by fixed-assignment policies. That is, whenever a job is presented to the scheduler, it is immediately and irrevocably assigned to a machine. The sequencing of jobs on the individual machines is in both cases (an online version of) the traditional WSEPT rule.

2. Discussion and further preliminaries.

As a matter of fact, results in stochastic scheduling either rely on the traditional (W)SEPT rule [21, 24, 30, 33, 34] for models without release dates, or they use linear programming relaxations to define list scheduling policies other than WSEPT [21, 26]. As soon as we assume that jobs arrive online, these approaches fail: the traditional off-line (W)SEPT rule cannot be implemented since it requires an a priori ordering of jobs in order of ratios $w_j/\mathbb{E}\left[P_j\right]$. Moreover, for the models with release dates, optimal LP solutions are not only required for the purpose of analysis, but also to define the corresponding list scheduling policies [21]. Hence, it is required to know beforehand the set of jobs J, their expected processing times $\mathbb{E}[P_j]$, and in the case with release dates also a uniform upper bound Δ on the squared coefficient of variation of all processing times distributions. Although we still use the same linear programming relaxation as Möhring et al. [21] within our analysis, the main difference lies in the fact that the algorithms we propose are purely combinatorial, and do not require the solution of linear programs in advance.

As in traditional online optimization, the adversary in the SoS model may choose an arbitrary sequence of jobs as well. These jobs, however, are stochastic, with corresponding processing time distributions. The actual processing times are realized according to exogenous probability distributions. Thus, the best the adversary can do is indeed to use an optimal stochastic scheduling policy in the traditional definition of stochastic scheduling policies by Möhring et al. [20]. In this view, our model somewhat compares to the idea of a *diffuse adversary* as defined by Koutsoupias and Papadimitriou [16]. Since deterministic processing times are contained as a special case, however, all lower bounds on the approximability known from deterministic online scheduling also hold for the SoS model of the present paper. Hence, no SoS policy can exist with a performance bound better than 1.309 [32].

Observe that the expected performance of any stochastic online policy is by definition no less than the expected performance of an optimal policy for a corresponding traditional stochastic problem, where the set of jobs J, their weights w_j, and their processing time distributions P_j are given at the outset. Hence, lower bounds on the expected objective value of an optimal stochastic scheduling policy carry over to the stochastic online setting that we consider in this paper. Therefore, we have the following lower bound on the performance of any stochastic online scheduling policy; it is a generalization of a lower bound by Eastman et al. [10] to stochastic processing times.

Lemma 2.1 (Möhring et al. [21]) *For any instance of* $\mathrm{P}\,|\,r_j\,|\,\mathbb{E}[\sum w_j C_j]$, *we have that*

$$\mathbb{E}[\mathrm{OPT}(P)] \geq \sum_j w_j \sum_{k\in H(j)} \frac{\mathbb{E}[P_k]}{m} - \frac{(m-1)(\Delta-1)}{2m} \sum_j w_j \mathbb{E}[P_j] \ ,$$

where Δ bounds the squared coefficient of variation of the processing times, that is, $\mathrm{Var}[P_j]/\mathbb{E}[P_j]^2 \leq \Delta$ *for all jobs $j = 1,\ldots,n$ and some $\Delta \geq 0$.*

Here, we have used a piece of notation that comes handy also later. For a given job $j \in J$, $H(j)$ denotes the jobs that have a higher priority in the order of ratios $w_j/\mathbb{E}[P_j]$, that is

$$H(j) = \left\{k \in J \mid \frac{w_k}{\mathbb{E}[P_k]} > \frac{w_j}{\mathbb{E}[P_j]}\right\} \cup \left\{k \leq j \mid \frac{w_k}{\mathbb{E}[P_k]} = \frac{w_j}{\mathbb{E}[P_j]}\right\}.$$

Accordingly, we define

$$L(j) = J \backslash H(j)$$

as those jobs that have lower priority in the order of ratios $w_j/\mathbb{E}[P_j]$. As a tie-breaking rule for jobs k with equal ratio $w_k/\mathbb{E}[P_k] = w_j/\mathbb{E}[P_j]$ we decide depending on the position in the online sequence relative to j. That is, if $k \leq j$ then k belongs to set $H(j)$, otherwise it is included in set $L(j)$. Notice that, by convention, we assume that $H(j)$ contains job j, too.

3. Stochastic online scheduling on a single machine.

In this section we consider the stochastic online scheduling problem on a single machine. When release dates are absent, it is well known that the WSEPT rule is optimal [23, 29].

For the problem of scheduling jobs with nontrivial release dates on a single machine, the currently best known result from traditional stochastic scheduling is an LP-based list scheduling algorithm with a performance bound of 3 [21]. Inspired by a corresponding algorithm for the deterministic online setting on parallel machines from [18], we propose the following scheduling policy.

Algorithm 1: α-Shift-WSEPT

```
Modify the release date $r_j$ of each job $j$ to $r'_j = \max\{r_j, \alpha\,\mathbb{E}[P_j]\}$, for
some fixed $\alpha > 0$. At any time $t$, when the machine is idle, start the
job with highest ratio $w_j/\mathbb{E}[P_j]$ among all available jobs, respecting
the modified release dates. (In case of ties, smallest index first.)
```

We first derive an upper bound on the expected completion time of a job, $\mathbb{E}\left[C_j^\alpha\right]$, when scheduling jobs on a single machine according to the α-Shift-WSEPT policy.

Lemma 3.1 *Let all processing times be δ-NBUE. Then the expected completion time of job j under α-*Shift-WSEPT *on a single machine can be bounded by*

$$\mathbb{E}\left[C_j^\alpha\right] \leq (1+\delta/\alpha)\, r_j' + \sum_{k \in H(j)} \mathbb{E}\left[P_k\right] .$$

Proof. We consider some job j. Let X denote a random variable measuring the remaining processing time of a job being processed at time r_j', if such a job exists. Otherwise X has value 0. Moreover, for any job k, let ξ_k be an indicator random variable that equals 1 if and only if job k starts processing at the earliest at time r_j', i.e., $\xi_k = 1$ if and only if $S_k^\alpha \geq r_j'$. The start of job j will be postponed beyond r_j' by X, and until there are no more higher priority jobs available. Hence, the expected start time of job j can be bounded by

$$\begin{aligned}
\mathbb{E}\left[S_j^\alpha\right] &\leq \mathbb{E}\Big[r_j' + X + \sum_{k \in H(j)\setminus\{j\}} P_k \xi_k\Big] \\
&= r_j' + \mathbb{E}\left[X\right] + \sum_{k \in H(j)\setminus\{j\}} \mathbb{E}\left[P_k \xi_k\right] \\
&\leq r_j' + \mathbb{E}\left[X\right] + \sum_{k \in H(j)\setminus\{j\}} \mathbb{E}\left[P_k\right],
\end{aligned}$$

where the last inequality follows from the fact that $P_k \xi_k \leq P_k$ for any job k.

Next, we show that $\mathbb{E}\left[X\right] \leq (\delta/\alpha) r_j'$. If the machine just finishes a job at time r_j' or is idle at that time, X has value 0. Otherwise, some job ℓ is in process at time r_j'. Note that this job might have lower or higher priority than job j. Such job ℓ was available at time $r_\ell' < r_j'$, and by definition of the modified release dates, we therefore know that $\mathbb{E}\left[P_\ell\right] \leq (1/\alpha) r_\ell' < (1/\alpha) r_j'$ for any such job ℓ. Moreover, letting $t = r_j' - S_\ell^\alpha$, the expected remaining processing time of such job ℓ, given that it is indeed in process at time r_j', is $\mathbb{E}\left[P_\ell - t \mid P_\ell > t\,\right]$. Due to the assumption of δ-NBUE processing times, we thus know that

$$\mathbb{E}\left[P_\ell - t \mid P_\ell > t\,\right] \;\leq\; \delta\, \mathbb{E}\left[P_\ell\right] \;\leq\; (\delta/\alpha) r_j'$$

for any job ℓ that could be in process at time r_j'. Hence, we obtain $\mathbb{E}\left[X\right] \leq (\delta/\alpha) r_j'$. Finally, the fact that $\mathbb{E}\left[C_j^\alpha\right] = \mathbb{E}\left[S_j^\alpha\right] + \mathbb{E}\left[P_j\right]$ concludes the proof. □

In fact, it is quite straightforward to use Lemma 3.1 in order to show the following.

Theorem 3.1 *The α-*Shift-WSEPT *algorithm is a $(\delta+2)$-approximation for the stochastic online single machine problem $1|r_j|\mathbb{E}\left[\sum w_j C_j\right]$, for δ-*NBUE *processing times. The best choice for α is $\alpha = 1$.*

Proof. With Lemma 3.1 and the definition of modified release dates $r_j' = \max\{r_j, \alpha\, \mathbb{E}\left[P_j\right]\}$ we can bound the expected value of a schedule obtained by α-Shift-WSEPT.

$$\begin{aligned}
\sum_j w_j\, \mathbb{E}\left[C_j^\alpha\right] \quad &\leq \quad (1+\delta/\alpha) \sum_j w_j \;\max\{r_j, \alpha\, \mathbb{E}\left[P_j\right]\} + \sum_j w_j \sum_{k \in H(j)} \mathbb{E}\left[P_k\right] \\
&= \quad \sum_j w_j \;\max\Big\{ (1+\delta/\alpha)\, r_j,\; (\alpha+\delta)\, \mathbb{E}\left[P_j\right] \Big\} + \sum_j w_j \sum_{k \in H(j)} \mathbb{E}\left[P_k\right] \\
&\leq \quad \max\Big\{ (1+\delta/\alpha)\,,\; (\alpha+\delta) \Big\} \sum_j w_j \left(r_j + \mathbb{E}\left[P_j\right]\right) + \sum_j w_j \sum_{k \in H(j)} \mathbb{E}\left[P_k\right] .
\end{aligned}$$

We can now apply the trivial lower bound $\sum_j w_j (r_j + \mathbb{E}\left[P_j\right]) \leq \mathbb{E}\left[\mathrm{OPT}(P)\right]$, and exploit the fact that $\sum_j w_j \sum_{k \in H(j)} \mathbb{E}\left[P_k\right] \leq \mathbb{E}\left[\mathrm{OPT}(P)\right]$ by Lemma 2.1 (for $m = 1$), and obtain

$$\sum_j w_j\, \mathbb{E}\left[C_j^\alpha\right] \;\leq\; \Big(1 + \max\Big\{1 + \frac{\delta}{\alpha}\,,\; \alpha+\delta\Big\}\Big)\, \mathbb{E}\left[\mathrm{OPT}(P)\right].$$

Simple analysis shows that $\alpha = 1$ minimizes the expression above, independently of δ, and the theorem follows. □

Note that for NBUE processing times, the result matches the best known performance bound of 3 derived by Möhring et. al in [21] for the traditional stochastic scheduling model. In contrast to α-Shift-WSEPT, however, their LP-based policy requires an a priori knowledge of the set of jobs J, their weights w_j, and their expected processing times $\mathbb{E}[P_j]$. Moreover, in the deterministic online setting, the best possible algorithm is 2-competitive as shown in [14], hence the corresponding lower bound of 2 holds for the stochastic online setting as well.

4. Stochastic online scheduling on parallel machines. In this section, we define stochastic online scheduling policies for the problem on parallel machines. We first consider the problem without nontrivial release dates, i.e., $r_j = 0$ for all jobs $j \in J$. Later, we generalize the policy for the problem with nontrivial release dates.

4.1 Scheduling jobs without release dates. In the case that all jobs arrive at time 0, the problem effectively turns into a traditional stochastic scheduling problem, $\mathrm{P}\,|\,|\,\mathbb{E}[\sum w_j\, C_j]$. For that problem, it is known that the WSEPT rule yields a $(1 + (m-1)(\Delta+1)/(2m))$-approximation, Δ being an upper bound on the squared coefficients of variation of the processing time distributions [21].

Nevertheless, we consider an online variant of the problem $\mathrm{P}\,|\,|\,\mathbb{E}[\sum w_j\, C_j]$ that resembles the online-list model from online optimization. We assume that the jobs are presented to the scheduler sequentially, and each job must immediately and irrevocably be assigned to a machine: a *fixed-assignment* policy. In particular, during this assignment phase, the scheduler does not know anything about the jobs that are still about to come. Once the jobs are assigned to the machines, the jobs on each machine may be sequenced in any order. We show that an intuitive and simple fixed-assignment policy exists that eventually yields the same performance bound as the one proved in [21] for the WSEPT rule. Note, however, that the WSEPT rule is not a feasible online policy in the model we consider here.

We introduce the following notation: If a job j is assigned to machine i, this is denoted by $j \to i$. Now we can define the MinIncrease policy as follows.

Algorithm 2: MinIncrease (MI)

When a job j is presented to the scheduler, it is assigned to the machine i that minimizes the expression

$$z(j,i) \;=\; w_j \cdot \sum_{k\in H(j),\, k<j,\, k\to i} \mathbb{E}[P_k] \;+\; \mathbb{E}[P_j] \cdot \sum_{k\in L(j),\, k<j,\, k\to i} w_k \;+ w_j \mathbb{E}[P_j]\;.$$

Once all jobs are assigned to machines, the jobs on each machine are sequenced in order of non-increasing ratios $w_j/\mathbb{E}[P_j]$.

Since WSEPT is known to be optimal on a single machine, MinIncrease in fact assigns each job j to that machine where it causes the least increase in the expected objective value, given the previously assigned jobs. This is expressed in the following lemma.

Lemma 4.1 *The expected objective value* $\mathbb{E}[\mathrm{MI}(P)]$ *of the* MinIncrease *policy equals* $\sum_j \min_i z(j,i)$.

Proof. The assignment of jobs to machines is independent of the realization of processing times. Hence, the expected completion time $\mathbb{E}[C_j]$ for some job j that has been assigned to machine i_j by MinIncrease is

$$\mathbb{E}[C_j] = \sum_{k\in H(j),\, k\to i_j} \mathbb{E}[P_k]\;,$$

because all jobs that are assigned to the same machine i_j are sequenced in order of non-increasing ratios $w_j/\mathbb{E}[P_j]$. Now, weighted summation over all jobs gives by linearity of expectation

$$\begin{aligned}
\mathbb{E}[\mathrm{MI}(P)] &= \mathbb{E}\Big[\sum\nolimits_j w_j C_j\Big] \\
&= \sum_j w_j \sum_{k\in H(j),\, k\to i_j} \mathbb{E}[P_k] \\
&= \sum_j w_j \sum_{k\in H(j),\, k\to i_j, k<j} \mathbb{E}[P_k] + \sum_j w_j \sum_{k\in H(j),\, k\to i_j, k>j} \mathbb{E}[P_k] + \sum_j w_j \mathbb{E}[P_j]\;.
\end{aligned}$$

This allows us to apply the following index rearrangement.

$$\sum_j w_j \sum_{k\in H(j),\, k>j} \mathbb{E}[P_k] \quad = \quad \sum_j \mathbb{E}[P_j] \sum_{k\in L(j),\, k<j} w_k \,. \tag{1}$$

Thus, we have

$$\begin{aligned}
\mathbb{E}[\mathrm{MI}(P)] \;&=\; \sum_j w_j \sum_{k\in H(j),\, k\to i_j, k<j} \mathbb{E}[P_k] + \sum_j \mathbb{E}[P_j] \sum_{k\in L(j),\, k\to i_j, k<j} w_k + \sum_j w_j \mathbb{E}[P_j] \\
&=\; \sum_j \Big(w_j \sum_{k\in H(j),\, k\to i_j, k<j} \mathbb{E}[P_k] + \mathbb{E}[P_j] \sum_{k\in L(j),\, k\to i_j, k<j} w_k + w_j\mathbb{E}[P_j] \Big) \\
&=\; \sum_j \min_i z(j,i) \;,
\end{aligned}$$

where the second equality makes use of (1) applied to each individual machine i_j, and the last equality holds since i_j is the machine minimizing $z(j,i)$ over all machines i. □

Now, we can derive the following performance guarantee for the MinIncrease policy.

Theorem 4.1 *Consider the stochastic online scheduling problem on parallel machines,* $\mathrm{P}|\,|\mathbb{E}[\sum w_j C_j]$, *as described above. Given that* $\mathrm{Var}[P_j]/\mathbb{E}[P_j]^2 \le \Delta$ *for all jobs* j *and some constant* $\Delta \ge 0$*, the* MinIncrease *policy is a* ρ*–approximation, where*

$$\rho = 1 + \frac{(m-1)(\Delta+1)}{2m} \,.$$

Proof. From Lemma 4.1 we know that $\mathbb{E}[\mathrm{MI}(P)] = \sum_j \min_i z(j,i)$ and, thus,

$$\begin{aligned}
\mathbb{E}[\mathrm{MI}(P)] \;&=\; \sum_j \min_i \Big\{ w_j \sum_{k\in H(j),\, k<j,\, k\to i} \mathbb{E}[P_k] \;+\; \mathbb{E}[P_j] \sum_{k\in L(j),\, k<j,\, k\to i} w_k \;+ w_j\mathbb{E}[P_j] \Big\} \\
&\le\; \sum_j \frac{1}{m} \Big(w_j \sum_{k\in H(j),\, k<j} \mathbb{E}[P_k] + \mathbb{E}[P_j] \sum_{k\in L(j),\, k<j} w_k \Big) + \sum_j w_j \mathbb{E}[P_j] \;,
\end{aligned}$$

where the inequality holds because the least expected increase is not more than the average expected increase over all machines.

Now, we first apply the index rearrangement (1) as in Lemma 4.1, and then plug in the inequality of Lemma 2.1. Using the trivial fact that $\sum_j w_j \mathbb{E}[P_j]$ is a lower bound for the expected performance $\mathbb{E}[\mathrm{OPT}(P)]$ of an optimal policy, we thus obtain

$$\begin{aligned}
\mathbb{E}[\mathrm{MI}(P)] \;&\le\; \frac{1}{m} \sum_j \Big(w_j \sum_{k\in H(j),\, k<j} \mathbb{E}[P_k] + w_j \sum_{k\in H(j),\, k>j} \mathbb{E}[P_k] \Big) + \sum_j w_j \mathbb{E}[P_j] \\
&=\; \frac{1}{m} \sum_j w_j \sum_{k\in H(j)} \mathbb{E}[P_k] + \frac{m-1}{m} \sum_j w_j \mathbb{E}[P_j] \\
&\le\; \mathbb{E}[\mathrm{OPT}(P)] + \frac{(m-1)(\Delta-1)}{2m} \sum_j w_j\mathbb{E}[P_j] + \frac{m-1}{m} \sum_j w_j \mathbb{E}[P_j] \\
&\le\; \Big(1 + \frac{(m-1)(\Delta+1)}{2m} \Big) \cdot \mathbb{E}[\mathrm{OPT}(P)] \,.
\end{aligned}$$

□

As mentioned above, this performance guarantee matches the currently best known performance guarantee for the traditional stochastic setting, which was derived for the performance of the WSEPT rule in [21]. The WSEPT rule, however, requires the knowledge of all jobs with their weights w_j and expected processing times $\mathbb{E}[P_j]$ at the outset. In contrast, the MinIncrease policy decides on machine assignments online, without any knowledge of the jobs to come. Finally, notice that these two policies are indeed different; this follows from simple examples.

Lower bound for fixed assignment policies. The requirement of a fixed assignment of jobs to machines beforehand may be interpreted as ignoring the additional information that evolves over time in the form of the actual realization of processing times. In the following, we therefore give a lower bound on the expected performance $\mathbb{E}[\text{FIX}(P)]$ of an optimal stochastic scheduling policy FIX that assigns jobs to machines beforehand. A fortiori, this lower bound holds for the best possible SoS policy, too.

Theorem 4.2 *For stochastic parallel machine scheduling with unit weights and i.i.d. exponential processing times,* $\text{P}|p_j \sim \exp(1)|\mathbb{E}[\sum C_j]$, *there exist instances such that*

$$\mathbb{E}[\text{FIX}(P)] \geq 3(\sqrt{2}-1) \cdot \mathbb{E}[\text{OPT}(P)] - \varepsilon\,,$$

for any $\varepsilon > 0$. *Here,* $3(\sqrt{2}-1) \approx 1.24$. *Hence, no policy that uses fixed assignments of jobs to machines can perform better in general.*

Notice that the theorem is formulated for the special case of exponentially distributed processing times. Stronger bounds can be obtained for arbitrary distributions. However, since our performance guarantees, as in [21], depend on the coefficient of variation of the processing times, we are particularly interested in lower bounds for classes of distributions where this coefficient of variation is small. The coefficient of variation of exponentially distributed random variables equals 1. For example, for the case of $m = 2$ machines, we get a lower bound of $8/7 \approx 1.14$ on the performance of any fixed-assignment policy, and for that case the performance bound of MinIncrease equals $2 - 1/m = 1.5$.

Proof of Theorem 4.2. Let us consider an instance with m machines and $n = m+k$ exponentially distributed jobs, $P_j \sim \exp(1)$, where $k \geq 1$ is an integer. The optimal stochastic scheduling policy is SEPT, shortest expected processing time first [4, 35], and the expected performance is (see, e.g., [31, Cor. 3.5.17])

$$\mathbb{E}[\text{OPT}(P)] = \mathbb{E}[\text{SEPT}(P)] = \sum_j \mathbb{E}\left[C_j^{\text{SEPT}}\right] = m + \sum_{j=m+1}^{n} \frac{j}{m} = m + k + \frac{k(k+1)}{2m}\,.$$

When, in a fixed assignment, one machine has to process at least two jobs more than another machine, the assignment can be improved by moving one job from the most loaded machine to the least loaded machine. Therefore, the best fixed assignment policy tries to distribute the jobs evenly over the machines. That is, it assigns $1 + \lfloor \frac{k}{m} \rfloor$ jobs to $m - k + m\lfloor \frac{k}{m} \rfloor$ machines and $1 + \lceil \frac{k}{m} \rceil$ jobs to $k - m\lfloor \frac{k}{m} \rfloor$ machines. Hence, there are m jobs with $\mathbb{E}[C_j] = l$ ($l = 1, \ldots, 1 + \lfloor \frac{k}{m} \rfloor$) and $k - m\lfloor \frac{k}{m} \rfloor$ jobs with $\mathbb{E}[C_j] = 2 + \lfloor \frac{k}{m} \rfloor$. The expected performance for the best fixed assignment policy FIX is

$$\mathbb{E}[\text{FIX}(P)] = \sum_j \mathbb{E}\left[C_j^{\text{FIX}}\right] = m + 2k + \left(k - \frac{m}{2} - \frac{m}{2}\left\lfloor \frac{k}{m} \right\rfloor\right) \cdot \left\lfloor \frac{k}{m} \right\rfloor\,.$$

For $m < k \leq 2m$, this value is equal to $\mathbb{E}[\text{FIX}(P)] = 3k$. Hence, for $m < k \leq 2m$, the ratio $\mathbb{E}[\text{FIX}(P)]/\mathbb{E}[\text{OPT}(P)]$ is

$$\frac{\mathbb{E}[\text{FIX}(P)]}{\mathbb{E}[\text{OPT}(P)]} = \frac{3k}{m + k + k(k+1)/(2m)}\,,$$

which is maximized for $k(m) \in \{\lfloor \sqrt{2}m \rfloor, \lceil \sqrt{2}m \rceil\}$. With this choice of k, the ratio $\mathbb{E}[\text{FIX}(P)]/\mathbb{E}[\text{OPT}(P)]$ tends to $3\sqrt{2}/(2+\sqrt{2}) = 3(\sqrt{2}-1) \approx 1.24$ as m tends to infinity. □

Notice that the lower bound of 1.24 holds whenever m, the number of machines, tends to infinity. For smaller numbers of machines, e.g. $m = 2, 3, 4, \ldots$ we use smaller numbers $k = k(m)$, namely $k(2) = 1, k(3) = 2, k(4) = 2, \ldots$, and obtain the lower bounds $8/7 \approx 1.14$, $7/6 \approx 1.16$, $32/27 \approx 1.18$, ...

Lower bound for MinIncrease**.** The lower bound on the performance ratio for *any* fixed assignment policy given in Theorem 4.2 holds for the MinIncrease policy, too. Hence, MinIncrease cannot be better than 1.24-approximative. For general (i.e., non-exponential) probability distributions we obtain a lower bound of $3/2$ on the expected performance of MinIncrease relative to the expected performance of an optimal scheduling policy, as shown by the following instance.

Example 4.1 *The instance consists of* $n-1$ *deterministic unit length jobs and one job with a stochastic two-point distributed processing time. There are* $m = 2$ *machines, and we assume that* n*, the number* *of jobs is even. The $n-1$ deterministic jobs have unit weight $w_j = 1$; they appear first in the online sequence. The final job in the online sequence is the stochastic job. It has processing time $p_j = n^2/4$ with probability $2/n$, and $p_j = 1$ with probability $1-2/n$. The weight w_j of the stochastic job equals the value of its expected processing time, i.e. $1-2/n+n/2$.*

The MinIncrease policy assigns $n/2-1$ deterministic jobs to one machine, and $n/2$ deterministic jobs to the other. The stochastic job is assigned to the machine with $n/2-1$ deterministic jobs. Hence, the expected objective value of the schedule under MinIncrease is $\mathbb{E}\left[\sum w_j\, C_j\right] = 3n^2/4 + o(n^2)$. An optimal stochastic policy would start the uncertain job and one deterministic job at time 0. At time $t=1$ it is known if the stochastic job has completed, or if it blocks the machine for another $n^2/4-1$ time units. If the stochastic job has completed then the remaining unit jobs are distributed equally on both machines, otherwise all deterministic jobs are scheduled on the same machine. Thus, the expected objective value of an optimal schedule is $\mathbb{E}\left[\mathrm{OPT}(P)\right] = n^2/2 + o(n^2)$. The ratio of both values tends to $3/2$ if the number of jobs tends to infinity.

Notice, however, that this result is less meaningful in comparison to the performance bound of Theorem 4.1, which depends on an upper bound Δ on the squared coefficient of variation.

4.2 Scheduling jobs with nontrivial release dates. In this section, we consider the setting where jobs arrive over time, that is, the stochastic online version of $\mathrm{P}\,|\,r_j\,|\,\mathbb{E}\left[\sum w_j\, C_j\right]$. The main idea is to adopt the MinIncrease policy to this setting. However, the difference is that we are no longer equipped with an optimal policy (as it was WSEPT in the previous section) to schedule the jobs that are assigned to a single machine. Even worse, even if we knew such a policy for a single machine, it would not be straightforward how to use it in the setting with parallel machines to define a feasible online scheduling policy. However, we propose to use the α-Shift-WSEPT rule as introduced in Section 3 to sequence the jobs that we have assigned to the same machine. The assignment of jobs to machines, on the other hand, remains the same as before in the case without release dates. In a sense, when assigning jobs to machines, we thus ignore the possible gain of information that occurs over time in the online-time model.

Algorithm 3: Modified MinIncrease

When a job j is presented to the scheduler at its release date r_j, it is assigned to the machine i that minimizes the expression

$$z(j,i) \;=\; w_j \sum_{k\in H(j),\, k<j,\, k\to i} \mathbb{E}\left[P_k\right] \;+\; \mathbb{E}\left[P_j\right] \sum_{k\in L(j),\, k<j,\, k\to i} w_k \;+ w_j \mathbb{E}\left[P_j\right].$$

On each machine, the jobs assigned to this machine are sequenced according to the α-Shift-WSEPT rule.

The crucial observation is that the α-Shift-WSEPT policy on machine i_j learns about job j's existence immediately at time r_j. Hence, for each single machine, it is indeed feasible to use the α-Shift-WSEPT rule, and the so-defined policy is a feasible SoS policy.

Theorem 4.3 *Consider the stochastic online scheduling problem on parallel machines with release dates, $\mathrm{P}|r_j|\mathbb{E}\left[\sum w_j\, C_j\right]$. Given that all processing times are δ-NBUE, the modified* MinIncrease *policy running* α-Shift-WSEPT *on each single machine is a ρ–approximation, where*

$$\rho = 1 + \max\left\{1+\frac{\delta}{\alpha}\;,\;\alpha+\delta+\frac{(m-1)(\Delta+1)}{2m}\right\},$$

Here, Δ is such that $\mathrm{Var}[P_j]/\mathbb{E}\left[P_j\right]^2 \le \Delta$ for all jobs j. In particular, since all processing times P_j are δ-NBUE, Lemma A.1 yields that $\Delta \le 2\delta-1$, hence $\rho \le 1+\max\{1+\delta/\alpha,\ \alpha+\delta(2m-1)/m\}$.

Proof. Let i_j be the machine to which job j is assigned. Then, by Lemma 3.1 we know that

$$\mathbb{E}\left[C_j\right] \le \Big(1+\frac{\delta}{\alpha}\Big)r'_j + \sum_{k\in H(j),\, k\to i_j} \mathbb{E}\left[P_k\right], \tag{2}$$

and the expected value of MinIncrease can be bounded by

$$\mathbb{E}\left[\mathrm{MI}(P)\right] \leq \Big(1+\frac{\delta}{\alpha}\Big)\sum_j w_j\, r'_j + \sum_j w_j \sum_{k\in H(j),\, k\to i_j} \mathbb{E}\left[P_k\right] . \tag{3}$$

For the second part of the right hand side of (3), we can use the same index rearrangement as in the proof of Theorem 4.1; see (1). We thus obtain

$$\sum_j w_j \sum_{k\in H(j),\, k\to i_j} \mathbb{E}\left[P_k\right] \;=\; \sum_j \left(w_j \sum_{k\in H(j),\, k\to i_j,\, k<j} \mathbb{E}\left[P_k\right] \;+\; \mathbb{E}\left[P_j\right] \sum_{k\in L(j),\, k\to i_j,\, k<j} w_k \;+\; w_j \mathbb{E}\left[P_j\right] \right) .$$

By definition of the modified MinIncrease algorithm, we know that any job j is assigned to the machine which minimizes the term in parenthesis. Hence, by the same averaging argument as before, we know that

$$\begin{aligned}\sum_j w_j \sum_{k\in H(j),\, k\to i_j} \mathbb{E}\left[P_k\right] \;&\leq\; \sum_j \left(w_j \sum_{k\in H(j),\, k<j} \frac{\mathbb{E}\left[P_k\right]}{m} + \mathbb{E}\left[P_j\right] \sum_{k\in L(j),\, k<j} \frac{w_k}{m} + w_j\mathbb{E}\left[P_j\right] \right)\\ &=\; \sum_j w_j \sum_{k\in H(j)} \frac{\mathbb{E}\left[P_k\right]}{m} \;+\; \frac{m-1}{m} \sum_j w_j \mathbb{E}\left[P_j\right] ,\end{aligned}$$

where the last equality again follows from index rearrangement. Plugging this into (3), leads to the following bound on the expected performance of MinIncrease.

$$\mathbb{E}\left[\mathrm{MI}(P)\right] \leq \Big(1+\frac{\delta}{\alpha}\Big)\sum_j w_j r'_j + \sum_j w_j \sum_{k\in H(j)} \frac{\mathbb{E}\left[P_k\right]}{m} + \frac{m-1}{m}\sum_j w_j \mathbb{E}\left[P_j\right] .$$

Plugging the bound of Lemma 2.1 into the above inequality, we obtain

$$\begin{aligned}\mathbb{E}\left[\mathrm{MI}(P)\right] \;&\leq\; \Big(1+\frac{\delta}{\alpha}\Big)\sum_j w_j r'_j + \mathbb{E}\left[\mathrm{OPT}(P)\right] + \frac{(m-1)(\Delta+1)}{2m} \sum_j w_j \mathbb{E}\left[P_j\right]\\ &=\; \mathbb{E}\left[\mathrm{OPT}(P)\right] + \sum_j w_j \left(\Big(1+\frac{\delta}{\alpha}\Big) r'_j + \frac{(m-1)(\Delta+1)}{2m}\mathbb{E}\left[P_j\right] \right) ,\end{aligned} \tag{4}$$

where again, Δ is an upper bound on the squared coefficient of variation of the processing time distributions P_j. By bounding r'_j by $r_j + \alpha\mathbb{E}\left[P_j\right]$, we obtain the following bound on the term in parenthesis of the right hand side of (4).

$$\begin{aligned}&\Big(1+\frac{\delta}{\alpha}\Big) r'_j + \frac{(m-1)(\Delta+1)}{2m}\mathbb{E}\left[P_j\right]\\ &\qquad \leq \Big(1+\frac{\delta}{\alpha}\Big) r_j + \Big(\alpha + \delta + \frac{(m-1)(\Delta+1)}{2m}\Big)\mathbb{E}\left[P_j\right]\\ &\qquad \leq \Big(r_j + \mathbb{E}\left[P_j\right]\Big) \max\left\{1+\frac{\delta}{\alpha}\ ,\ \alpha+\delta+\frac{(m-1)(\Delta+1)}{2m}\right\} .\end{aligned}$$

By using this inequality in equation (4), and applying the trivial lower bound $\sum_j w_j(r_j + \mathbb{E}\left[P_j\right]) \leq \mathbb{E}\left[\mathrm{OPT}(P)\right]$ on the expected optimum performance, we get the claimed performance bound of

$$\rho = 1 + \max\left\{1+\frac{\delta}{\alpha}\ ,\ \alpha+\delta+\frac{(m-1)(\Delta+1)}{2m}\right\} .$$

Since all processing times are δ-NBUE, we know by Lemma A.1 in the appendix that $\Delta \leq 2\delta - 1$ and thus the second claim of the theorem follows, namely $\rho \leq 1 + \max\{1+\delta/\alpha,\ \alpha + \delta(2m-1)/m\}$. □

For NBUE processing times, where $\Delta = \delta = 1$, Theorem 4.3 yields a performance bound of

$$\rho = 2 + \max\left\{\frac{1}{\alpha},\ \alpha + \frac{m-1}{m}\right\} .$$

This term is minimal for $\alpha = (\sqrt{5m^2-2m+1} - m + 1)/(2m)$, which yields a ratio of $\rho = 2 + (\sqrt{5m^2-2m+1} + m - 1)/(2m)$. This is less than $(5+\sqrt{5})/2 - 1/(2m) \approx 3.62 - 1/(2m)$ improving upon the previously best known bound of $4-1/m$ from [21] for the traditional stochastic problem. More generally, for δ-NBUE processing times, optimizing the term $\max\{1+\delta/\alpha,\ \alpha+\delta(2m-1)/m\}$ for α yields $\rho < 3/2+\delta(2m-1)/2m+\sqrt{4\delta^2+1}/2$.

Moreover, for deterministic processing times, where $\Delta = 0$ and $\delta = 1$, Theorem 4.3 yields a performance bound of

$$\rho = 2 + \max\left\{\frac{1}{\alpha},\ \alpha + \frac{m-1}{2m}\right\} .$$

Optimizing for α yields $\alpha = (\sqrt{17m^2-2m+1}-m+1)/(4m)$, which yields a ratio of $\rho = 2 + (\sqrt{17m^2-2m+1}+m-1)/(4m)$. This is less than 3.281 for any value of m, leaving only a small gap to the currently best known bound of 2.62 for deterministic online scheduling [8]. In fact, it matches the competitive ratio of the deterministic parallel machine version of α-Shift-WSEPT from [18].

4.3 Randomized job assignment. As a matter of fact, the MinIncrease policy can be interpreted as the derandomized version of a policy that assign jobs uniformly at random to the machines. Even though randomly assigning jobs to machines ignores much information, it is nevertheless known to be quite powerful as has been observed already by, e.g., Schulz and Skutella [25]. They apply a random assignment strategy, based on the solution of an LP-relaxation, for scheduling jobs with deterministic processing times on unrelated machines. For the special case of identical machines, their approach corresponds to assigning jobs uniformly at random to the machines. The random assignment strategy for the stochastic online scheduling problem at hand is as follows.

Algorithm 4: RandAssign

```
When a job is presented to the scheduler, it is assigned to machine $i$
with probability $1/m$ for all $i = 1,\dots,m$. The jobs assigned to
machine $i$ are scheduled according to the $\alpha$-Shift-WSEPT policy.
```

Theorem 4.4 *Consider the stochastic online scheduling problem on parallel machines with release dates,* $\mathrm{P}|r_j|\mathbb{E}\left[\sum w_j\, C_j\right]$. *Given that all processing times are δ-NBUE, the* RandAssign *policy running* α-Shift-WSEPT *on each single machine is a ρ–approximation, where*

$$\rho = 1 + \max\left\{1+\frac{\delta}{\alpha}\ ,\ \alpha+\delta+\frac{(m-1)(\Delta+1)}{2m}\right\} ,$$

Here, Δ is such that $\mathrm{Var}[P_j]/\mathbb{E}\left[P_j\right]^2 \le \Delta$ for all jobs j. In particular, since all processing times P_j are δ-NBUE, Lemma A.1 yields that $\Delta \le 2\delta-1$, hence $\rho \le 1+\max\{1+\delta/\alpha,\ \alpha+\delta(2m-1)/m\}$.

Proof. Consider a job j and let i denote the machine to which it has been assigned. Let $\Pr[j\to i]$ be the probability for job j being assigned to machine i. Then, by Lemma 3.1 we know that

$$\begin{aligned}\mathbb{E}\left[C_j\,|\,j\to i\right] &\le \left(1+\frac{\delta}{\alpha}\right)r'_j + \sum_{k\in H(j)} \Pr\left[k\to i\,|\,j\to i\right]\cdot\mathbb{E}\left[P_k\,|\,j\to i\right]\\ &= \left(1+\frac{\delta}{\alpha}\right)r'_j + \sum_{k\in H(j)} \Pr\left[k\to i\,|\,j\to i\right]\cdot\mathbb{E}\left[P_k\right] .\end{aligned}$$

The probability that a job is assigned to a certain machine is equal for all machines, i.e., $\Pr[j\to i] = 1/m$ for all $i = 1,\dots,m$, and for any job j. Unconditioning the expected value of job j's completion time yields

$$\begin{aligned}\mathbb{E}\left[C_j\right] &= \sum_{i=1}^{m}\Pr\left[j\to i\right]\cdot\mathbb{E}\left[C_j\,|\,j\to i\right]\\ &\le \left(1+\frac{\delta}{\alpha}\right)r'_j + \sum_{i=1}^{m}\Pr\left[j\to i\right]\cdot\sum_{k\in H(j)}\Pr\left[k\to i\,|\,j\to i\right]\cdot\mathbb{E}\left[P_k\right]\\ &= \left(1+\frac{\delta}{\alpha}\right)r'_j + \sum_{k\in H(j)}\frac{\mathbb{E}\left[P_k\right]}{m} + \frac{m-1}{m}\,\mathbb{E}\left[P_j\right] ,\end{aligned}$$

where the last equality is due to the independence of the job assignments to the machines. Then, the expected objective value of RandAssign, $\mathbb{E}[\text{RA}(P)]$, can be bounded by

$$\begin{aligned} \mathbb{E}[\text{RA}(P)] &= \sum_j w_j \mathbb{E}[C_j] \\ &\leq \Big(1+\frac{\delta}{\alpha}\Big)\sum_j w_j r'_j + \sum_j w_j \sum_{k\in H(j)} \frac{\mathbb{E}[P_k]}{m} + \frac{m-1}{m}\sum_j w_j \mathbb{E}[P_j] \,. \end{aligned}$$

This bound equals the upper bound that we achieved on the expected performance of the MinIncrease policy in the proof of Theorem 4.3. Hence, we conclude the proof in the same way and with the same result as for MinIncrease. □

Acknowledgments. The authors thank the two anonymous referees for valuable comments, and Martin Skutella for pointing out that the MinIncrease algorithm can be interpreted as the derandomization of an algorithm that distributes jobs to machines at random. Thanks to Rolf H. Möhring for helpful discussions, and Andreas S. Schulz and Sven O. Krumke for pointing out some misinterpretations in an earlier version of this paper.

References

[1] F. N. Afrati, E. Bampis, C. Chekuri, D. R. Karger, C. Kenyon, S. Khanna, I. Milis, M. Queyranne, M. Skutella, C. Stein, and M. Sviridenko, *Approximation schemes for minimizing average weighted completion time with release dates*, Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS), 1999, pp. 32–43.

[2] E. J. Anderson and C. N. Potts, *On-line scheduling of a single machine to minimize total weighted completion time*, Mathematics of Operations Research **29** (2004), 686–697.

[3] A. Borodin and R. El-Yaniv, *Online computation and competitive analysis*, Cambridge University Press, 1998.

[4] J. L. Bruno, P. J. Downey, and G. N. Frederickson, *Sequencing tasks with exponential service times to minimize the expected flowtime or makespan*, Journal of the ACM **28** (1981), 100–113.

[5] S. Chakrabarti and S. Muthukrishnan, *Resource scheduling for parallel database and scientific applications*, Proc. 8th Ann. ACM Symp. on Parallel Algorithms and Architectures (Padua, Italy), 1996, pp. 329–335.

[6] C. Chekuri, R. Johnson, R. Motwani, B. Natarajan, B. Rau, and M. Schlansker, *An analysis of profile-driven instruction level parallel scheduling with application to super blocks*, Proc. 29th IEEE/ACM Int. Symp. on Microarchitecture (Paris, France), 1996, pp. 58–69.

[7] M.C. Chou, H. Liu, M. Queyranne, and D. Simchi-Levi, *On the asymptotic optimality of a simple on-line algorithm for the stochastic single machine weighted completion time problem and its extensions*, 2005, To appear in Operations Research.

[8] J. Correa and M. Wagner, *LP-based online scheduling: From single to parallel machines*, Integer Programming and Combinatorial Optimization (M. Jünger and V. Kaibel, eds.), Lecture Notes in Computer Science, vol. 3509, Springer, 2005, pp. 196–209.

[9] M. A. H. Dempster, J. K. Lenstra, and A. H. G. Rinnooy Kan (editors), *Deterministic and stochastic scheduling*, D. Reidel Publishing Company, Dordrecht, 1982.

[10] W. Eastman, S. Even, and I. Isaacs, *Bounds for the optimal scheduling of n jobs on m processors*, Management Science **11** (1964), 268–279.

[11] A. Fiat and G. J. Woeginger, *On-line scheduling on a single machine: Minimizing the total completion time*, Acta Informatica **36** (1999), 287–293.

[12] 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** (1979), 287–326.

[13] W. J. Hall and J. A. Wellner, *Mean residual life*, Proc. Int. Symp. on Statistics and Related Topics (Ottawa, ON, Canada) (M. Csörgö, D. A. Dawson, J. N. K. Rao, and A. K. Md. E. Saleh, eds.), North-Holland, Amsterdam, The Netherlands, 1981, pp. 169–184.

[14] H. Hoogeveen and A. P. A. Vestjens, *Optimal on-line algorithms for single-machine scheduling*, Integer Programming and Combinatorial Optimization (W. H. Cunningham, S. T. McCormick, and M. Queyranne, eds.), Lecture Notes in Computer Science, vol. 1084, Springer, 1996, pp. 404–414.

[15] A. Karlin, M. Manasse, L. Rudolph, and D. Sleator, *Competitive snoopy paging*, Algorithmica **3** (1988), 70–119.

[16] E. Koutsoupias and C. H. Papadimitriou, *Beyond competitive analysis*, SIAM Journal on Computing **30** (2000), 300–317.

[17] E. L. Lenstra, A. H. G. Rinooy Kan, and P. Brucker, *Complexity of machine scheduling problems*, Annals of Discrete Mathematics **1** (1977), 243–362.

[18] N. Megow and A. S. Schulz, *On-line scheduling to minimize average completion time revisited*, Operations Research Letters **32** (2004), 485–490.

[19] N. Megow, M. Uetz, and T. Vredeveld, *Stochastic online scheduling on parallel machines*, Approximation and Online Algorithms (G. Persiano and R. Solis-Oba, eds.), Lecture Notes in Computer Science, vol. 3351, Springer, 2005, pp. 167–180.

[20] R. H. Möhring, F. J. Radermacher, and G. Weiss, *Stochastic scheduling problems I: General strategies*, ZOR - Zeitschrift für Operations Research **28** (1984), 193–260.

[21] 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** (1999), 924–942.

[22] K. Pruhs, J. Sgall, and E. Torng, *Online scheduling*, Handbook of Scheduling: Algorithms, Models, and Performance Analysis (J. Leung, ed.), CRC Press, 2004.

[23] M. H. Rothkopf, *Scheduling with random service times*, Management Science **12** (1966), 703–713.

[24] M. Scharbrodt, T. Schickinger, and A. Steger, *A new average case analysis for completion time scheduling*, Proc. 34th Ann. ACM Symp. on the Theory of Computing (Montréal, QB, Canada), 2002, pp. 170–178.

[25] A. S. Schulz and M. Skutella, *Scheduling unrelated machines by randomized rounding*, SIAM Journal on Discrete Mathematics **15** (2002), 450–469.

[26] M. Skutella and M. Uetz, *Stochastic machine scheduling with precedence constraints*, SIAM Journal on Computing **34** (2005), 788–802.

[27] M. Skutella and G.J. Woeginger, *A PTAS for minimizing the total weighted completion time on identical parallel machines*, Mathematics of Operations Research **25** (2000), 63–75.

[28] D. Sleator and R. Tarjan, *Amortized efficiency of list update and paging rules*, Communications of the ACM **28** (1985), 202–208.

[29] W. Smith, *Various optimizers for single-stage production*, Naval Research Logistics Quarterly **3** (1956), 59–66.

[30] A. Souza and A. Steger, *The expected competitive ratio for weighted completion time scheduling*, Proc. 21st Symp. on Theoretical Aspects of Computer Science (V. Diekert and M. Habib, eds.), Lecture Notes in Computer Science, vol. 2996, Springer, 2004, pp. 620–631.

[31] M. Uetz, *Algorithms for deterministic and stochastic scheduling*, Cuvillier Verlag, Göttingen, Germany, 2002.

[32] A. P. A. Vestjens, *On-line machine scheduling*, Ph.D. thesis, Eindhoven University of Technology, Netherlands, 1997.

[33] G. Weiss, *Approximation results in parallel machines stochastic scheduling*, Annals of Operations Research **26** (1990), 195–242.

[34] ______, *Turnpike optimality of Smith's rule in parallel machines stochastic scheduling*, Mathematics of Operations Research **17** (1992), 255–270.

[35] 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** (1980), 187–202.

Appendix A. Coefficient of variation for δ-NBUE random variables In this section, we show the relation between the value of δ and the (squared) coefficient of variation $\mathrm{CV}\left[X\right]^2 = \mathrm{Var}\left[X\right]/\mathbb{E}\left[X\right]^2$ for a δ-NBUE random variable X.

Lemma A.1 *Let X be a δ-NBUE random variable and let* $\mathrm{CV}[X]$ *denote the coefficient of variation of X. Then* $\mathrm{CV}[X]^2 \leq 2\delta - 1$.

Proof. We prove the lemma for continuous random variables X; the proof for discrete random variables goes along the same lines. Let X be a non-negative δ-NBUE random variable, with cumulative distribution function F and density f.

By definition of conditional expectation, we know that

$$\mathbb{E}\left[X - t \,\middle|\, X > t\right] = \frac{\int_t^\infty (x-t) f(x) dx}{1 - F(t)}. \tag{5}$$

As $x - t = \int_0^{x-t} dy$, we can write the nominator of the right hand side as

$$\begin{aligned}\int_{x=t}^{\infty} (x-t) f(x) dx &= \int_{x=t}^{\infty}\int_{y=0}^{x-t} f(x) dy dx = \int_{y=0}^{\infty}\int_{x=y+t}^{\infty} f(x) dx dy \\ &= \int_{x=t}^{\infty} 1 - F(x) dx, \end{aligned}\tag{6}$$

where the second equality is obtained by changing the order of integration.

As X is δ-NBUE, i.e., $\mathbb{E}\left[X - t \,|\, X > t\right] \leq \delta \mathbb{E}\left[X\right]$, it follows from (5) and (6) that

$$\int_{x=t}^{\infty} 1 - F(x) dx = \mathbb{E}\left[X - t \,\middle|\, X > t\right](1 - F(t)) \leq \delta \mathbb{E}\left[X\right](1 - F(t)). \tag{7}$$

By integrating the right hand side of the above inequality over t, we obtain

$$\delta \mathbb{E}\left[X\right] \int_{t=0}^{\infty} 1 - F(t) dt = \delta \mathbb{E}\left[X\right]^2. \tag{8}$$

Hall and Wellner [13, Equality (4.1)] showed that integrating the left hand side of (7) over t yields

$$\int_{t=0}^{\infty}\int_{x=t}^{\infty} 1 - F(t) dt = \frac{1}{2}\mathbb{E}\left[X^2\right]. \tag{9}$$

Hence, using (8) and (9) in (7), we have

$$\mathbb{E}\left[X^2\right] \leq 2\delta \mathbb{E}\left[X\right]^2.$$

Rearranging terms yields the desired bound on the squared coefficient of variation:

$$\mathrm{CV}[X]^2 = \frac{\mathbb{E}\left[X^2\right] - \mathbb{E}\left[X\right]^2}{\mathbb{E}\left[X\right]^2} \leq 2\delta - 1.$$

□