
Mathematics of Operations Research

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

# Models and Algorithms for Stochastic Online Scheduling

Nicole Megow, Marc Uetz, Tjark Vredeveld,

To cite this article:
Nicole Megow, Marc Uetz, Tjark Vredeveld, (2006) Models and Algorithms for Stochastic Online Scheduling. Mathematics of Operations Research 31(3):513-525. https://doi.org/10.1287/moor.1060.0201

**MATHEMATICS OF OPERATIONS RESEARCH**
Vol. 31, No. 3, August 2006, pp. 513–525
ISSN 0364-765X | EISSN 1526-5471 | 06 | 3103 | 0513

DOI 10.1287/moor.1060.0201


# Models and Algorithms for Stochastic Online Scheduling

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

Marc Uetz, Tjark Vredeveld
Department of Quantitative Economics, Maastricht University, P.O. Box 616, 6200 MD Maastricht, The Netherlands
{m.uetz@ke.unimaas.nl, 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 nonpreemptive 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 performance guarantees previously known for stochastic and online parallel machine scheduling, respectively. For processing times that follow new better than used in expectation (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 dynamic optimization; online optimization; total weighted completion time; approximation
*MSC2000 subject classification*: Primary: 90B36; secondary: 68W40, 68W25, 68M20
*OR/MS subject classification*: Primary: production/scheduling; secondary: approximation/heuristic
*History*: Received January 6, 2005; revised September 7, 2005.

**1. Introduction.** Scheduling on identical parallel machines to minimize the total weighted completion time of jobs, $P \parallel \sum w_j C_j$ 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 (Chakrabarti and Muthukrishnan [5]), or compiler optimization (Chekuri et al. [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, $P|r_j|\sum w_j C_j$. In the offline (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 (Lenstra et al. [17]) and they admit a polynomial time approximation scheme (Afrati et al. [1], Skutella and Woeginger [28]).

To cope with scenarios where there is uncertainty about the future, there are two major frameworks in the theory of scheduling: (1) *stochastic scheduling* and (2) *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 distribution functions of 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) (Pruhs et al. [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 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 nonpreemptively 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 $r_j \le r_k$ for $j < k$. Note 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* (*SOS*) *policy* that minimizes the expected value of the weighted completion times of jobs, $\mathbb{E}[\sum w_j C_j]$. Our definition of an SOS 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. 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 *online*, thus at any time, it must not utilize any information about jobs that will be released in the future. Moreover, it needs to be *nonanticipatory*, thus at any time, it must not utilize the actual processing times of jobs that are scheduled (or unscheduled) but not yet completed. An *optimal scheduling policy* is defined as a nonanticipatory scheduling policy that minimizes the objective function value in expectation. Note that we do not assume that an optimal policy needs to be online. Note also that even an optimal scheduling policy generally fails to yield an optimal solution for all realizations of the processing times; this is because it is nonanticipatory.

For an instance $I$, consisting of the number of machines $m$, the set of jobs $J$ together with their release dates $r_j$, weights $w_j$, and processing time distributions $P_j$, let $S_j^\Pi(I)$ and $C_j^\Pi(I)$ denote the random variables for start and completion times of jobs under policy $\Pi$. We also write $S_j^\Pi$ and $C_j^\Pi$ for short. We let

$$\mathbb{E}[\Pi(I)] = \mathbb{E}\left[\sum_{j\in J} w_j C_j^\Pi(I)\right] = \sum_{j\in J} w_j \mathbb{E}[C_j^\Pi(I)]$$

denote the expected performance of a scheduling policy $\Pi$ on instance $I$. Let us denote the above-defined model as stochastic online scheduling (SOS).

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

Definition 1.1. An SOS policy $\Pi$ is a *$\rho$-approximation* if, for some $\rho \geq 1$, and all instances $I$ of the given problem,

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

Here, $\mathrm{OPT}(I)$ denotes an optimal stochastic scheduling policy on the given instance $I$, 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 value $\rho$ is called the *performance guarantee* of policy $\Pi$.

Note that the SOS policy $\Pi$ 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 $\Pi$ 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 policy $\Pi$ itself because it does not foresee their actual realizations $p_j$ either.

Probably the best-known scheduling policy in stochastic scheduling is the rule weighted shortest expected processing time (WSEPT) first. 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). 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 nonincreasing ratios $w_j/\mathbb{E}[P_j]$. For unit weights, it reduces to shortest expected processing time (SEPT) 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 (Rothkopf [23], Smith [30]).

**Related work.** Stochastic machine scheduling models have been addressed mainly since the 1980s (Dempster et al. [9]). In the traditional stochastic setting, Weiss [34, 35] analyzes the performance of the WSEPT rule. He derives additive performance bounds for the stochastic parallel machine model without release dates, $P \,\|\, \mathbb{E}[\sum w_j C_j]$. 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 [27]. In these papers, the expected performance of the WSEPT rule

and linear programming (LP-)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 (Möhring et al. [21]) and also with precedence constraints (Skutella and Uetz [27]). 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 (Karlin et al. [15], Sleator and Tarjan [29]). An algorithm is called *$\rho$-competitive* if, for any instance, a solution is achieved with value not worse than $\rho$ times the value of an optimal offline 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 because 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 [33] proves a lower bound of 1.309 for the competitive ratio of any deterministic online algorithm for $\mathrm{P}|r_j|\sum w_jC_j$, even for unit weights. The currently best-known deterministic algorithm for $\mathrm{P}|r_j|\sum w_jC_j$ 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 [26]).

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 $1|r_j|\sum w_jC_j$, assuming that the weights $w_j$ and processing time $p_j$ can be bounded from above and below 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 Scharbrodt et al. [24] and Souza and Steger [31]. 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, it is not based on a comparison to the optimal stochastic scheduling policy. The underlying adversary is stronger than in traditional stochastic scheduling, yet they derive constant bounds on what they call the *expected competitive ratio*.

Subsequently to our work and inspired by a recent paper (Correa and Wagner [8]), Schulz [25] gave a randomized 3-approximation policy for the stochastic online version of $\mathrm{P}|r_j|\mathbb{E}[\sum w_jC_j]$, under the assumption of a certain class of processing time distributions. As stated in his paper, a derandomized version of this policy matches our performance guarantee for this special class of processing time distributions.

**Results and methodology.** We propose simple, combinatorial online scheduling policies for SOS 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, $\mathrm{P}\,\|\,\mathbb{E}[\sum w_jC_j]$, the performance guarantee is

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

Here, $\Delta$ is an upper bound on the squared coefficients of variation of the processing time distributions $P_j$, that is, $\mathrm{Var}[P_j]/\mathbb{E}[P_j]^2 \le \Delta$ 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 in a more restricted setting than traditional stochastic scheduling. 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. 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 stochastic online policy in this restricted setting that achieves the same performance guarantee as the above-mentioned bound for the WSEPT rule. (The traditional

WSEPT rule is not a feasible policy in this setting because the assignment of jobs to machines depends on the realizations of processing times.)

For the model with release dates, we prove a slightly more complicated performance guarantee that is valid for a class of processing time distributions that we call $\delta$-NBUE, generalizing the well-known class of NBUE distributions (new better than used in expectation).

DEFINITION 1.3 ($\delta$-NBUE). A nonnegative random variable $X$ is $\delta$-NBUE if, for $\delta \geq 1$, $\mathbb{E}[X - t \mid X > t] \leq \delta\mathbb{E}[X]$ for all $t \geq 0$.

For identical parallel machine scheduling with release dates $P|r_j|\sum w_jC_j$ and $\delta$-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, $\Delta$ is an upper bound on the squared coefficients of variation $\mathrm{Var}[P_j]/\mathbb{E}[P_j]^2$ of the processing time distributions $P_j$. For example, for ordinary NBUE distributions, where $\Delta = \delta = 1$, we obtain a performance guarantee $\rho < 3.62 - 1/(2m)$. 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 (Möhring et al. [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 obtain a performance guarantee $\rho < 3.281$, 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 (Möhring et al. [21], Scharbrodt et al. [24], Souza and Steger [31], Weiss [34, 35]) for models without release dates, or they use LP-relaxations to define list scheduling policies other than WSEPT (Möhring et al. [21], Skutella and Uetz [27]). As soon as we assume that jobs arrive online, the approaches of these papers fail: The traditional offline (W)SEPT rule cannot be implemented because it requires an a priori ordering of jobs in order of ratios $w_j/\mathbb{E}[P_j]$. To obtain the results for models with release dates, Möhring et al. [21] use optimal LP-solutions not only for the purpose of analysis, but also to define the corresponding list scheduling policies. Although we still use the same LP-relaxation as Möhring et al. [21] within our analysis, the main difference lies in the fact that the algorithms we propose are combinatorial, and do not require the solution of linear programs.

As in traditional online optimization, the adversary in the proposed stochastic online (SOS) model may choose an arbitrary sequence of jobs. 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]. Because 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 this paper. Hence, no SOS policy can exist with a performance bound better than 1.309 (Vestjens [33]).

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 release dates $r_j$, weights $w_j$, and 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 SOS 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 $I$ of $P|r_j|\mathbb{E}[\sum w_jC_j]$, we have that*

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

*where $\Delta$ 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,

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

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 $k \le j$, then $k$ belongs to set $H(j)$, otherwise it is included in set $L(j)$. Note that, by convention, we assume that $H(j)$ also contains job $j$.

**3. Stochastic online scheduling on a single machine.** In this section, we consider the SOS problem on a single machine. When release dates are absent, it is well known that the WSEPT rule is optimal (Rothkopf [23], Smith [30]).

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 (Möhring et al. [21]). Inspired by a corresponding algorithm for the deterministic online setting on parallel machines from Megow and Schulz [18], we propose the following scheduling policy.

Algorithm 1 ($\alpha$-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 the 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, $\mathbb{E}[C^\alpha_j]$, when scheduling jobs on a single machine according to the $\alpha$-Shift-WSEPT policy.

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

$$\mathbb{E}[C^\alpha_j] \le (1 + \delta/\alpha) r'_j + \sum_{k \in H(j)} \mathbb{E}[P_k].$$

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 $\xi_k$ 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 $S^\alpha_k \ge r'_j$. 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

$$\begin{aligned}
\mathbb{E}[S^\alpha_j] &\le \mathbb{E}\left[ r'_j + X + \sum_{k \in H(j)\backslash\{j\}} P_k \xi_k \right] \\
&= r'_j + \mathbb{E}[X] + \sum_{k \in H(j)\backslash\{j\}} \mathbb{E}[P_k \xi_k] \\
&\le r'_j + \mathbb{E}[X] + \sum_{k \in H(j)\backslash\{j\}} \mathbb{E}[P_k],
\end{aligned}$$

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

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

$$\mathbb{E}[P_\ell - t \mid P_\ell > t] \le \delta \mathbb{E}[P_\ell] \le (\delta/\alpha) r'_j$$

for any job $\ell$ that could be in process at time $r'_j$. Hence, we obtain $\mathbb{E}[X] \le (\delta/\alpha) r'_j$. Finally, the fact that $\mathbb{E}[C^\alpha_j] = \mathbb{E}[S^\alpha_j] + \mathbb{E}[P_j]$ concludes the proof. □

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

THEOREM 3.1. *The* $\alpha$-SHIFT-WSEPT *algorithm is a* $(\delta+2)$*-approximation for the stochastic online single machine problem* $1|r_j|\mathbb{E}[\sum w_jC_j]$ *for* $\delta$*-NBUE processing times. The best choice for* $\alpha$ *is* $\alpha=1$.

PROOF. With Lemma 3.1 and the definition of modified release dates $r'_j=\max\{r_j,\alpha\mathbb{E}[P_j]\}$, we can bound the expected value of a schedule obtained by $\alpha$-SHIFT-WSEPT:

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

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

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

Now, $\alpha=1$ minimizes this expression, independently of $\delta$, and the theorem follows. □

Note that for NBUE processing times, the result matches the currently best-known performance bound of 3 derived by Möhring et al. in [21] for the traditional stochastic scheduling model. Their LP-based policy, however, 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 (Hoogeveen and Vestjens [14]), hence the corresponding lower bound of 2 holds for the stochastic online setting too.

**4. Stochastic online scheduling on parallel machines.** In this section, we define SOS policies for the problem on parallel machines. We first consider the problem without nontrivial release dates, and later generalize to 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, $\mathrm{P}\,\|\,\mathbb{E}[\sum w_jC_j]$. For that problem, it is known that the WSEPT rule yields a $(1+(m-1)(\Delta+1)/(2m))$-approximation, $\Delta$ being an upper bound on the squared coefficients of variation of the processing time distributions (Möhring et al. [21]).

Nevertheless, we consider an online variant of the problem $\mathrm{P}\,\|\,\mathbb{E}[\sum w_jC_j]$ 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 Möhring et al. [21] for the WSEPT rule. In this context, recall that the WSEPT rule is not a feasible online policy in the considered online model.

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 nonincreasing ratios $w_j/\mathbb{E}[P_j]$.

Because 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* $\mathbb{E}[\mathrm{MI}(I)]$ *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 nonincreasing ratios $w_j/\mathbb{E}[P_j]$. Now, weighted summation over all jobs gives by linearity of expectation

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

This allows us to apply the following index rearrangement:

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

Thus, we have

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

where the second equality makes use of (1) applied to each individual machine $i_j$, and the last equality holds because $i_j$ 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 (SOS) 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* $\rho$*-approximation, where*

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

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

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

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 $\mathbb{E}[\mathrm{OPT}(I)]$ of an optimal policy, we thus obtain

$$\begin{aligned}
\mathbb{E}[\mathrm{MI}(I)] &\le \frac{1}{m} \sum_j \left( w_j \sum_{k \in H(j),\, k<j} \mathbb{E}[P_k] + \; w_j \sum_{k \in H(j),\, k>j} \mathbb{E}[P_k] \right) + \sum_j w_j \mathbb{E}[P_j] \\
&= \frac{1}{m} \sum_j w_j \sum_{k \in H(j)} \mathbb{E}[P_k] + \frac{m-1}{m} \sum_j w_j \mathbb{E}[P_j]
\end{aligned}$$

$$\leq \mathbb{E}[\mathrm{OPT}(I)] + \frac{(m-1)(\Delta-1)}{2m}\sum_j w_j\mathbb{E}[P_j] + \frac{m-1}{m}\sum_j w_j\mathbb{E}[P_j]$$

$$\leq \left(1 + \frac{(m-1)(\Delta+1)}{2m}\right)\cdot \mathbb{E}[\mathrm{OPT}(I)]. \quad \square$$

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 Möehring et al. [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, note 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 $\mathbb{E}[\mathrm{FIX}(I)]$ 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,* $\mathrm{P}|p_j \sim \exp(1)|\mathbb{E}[\sum C_j]$, *there exist instances* $I$ *such that*

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

*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.*

Note that the theorem is formulated for the special case of exponentially distributed processing times. Stronger bounds can be obtained for arbitrary distributions. However, because our performance guarantees, as in Möhring et al. [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 (Bruno et al. [4], Weiss and Pinedo [36]), and the expected performance is (see, e.g., Uetz [32, Corollary 3.5.17])

$$\mathbb{E}[\mathrm{OPT}(I)] = \mathbb{E}[\mathrm{SEPT}(I)] = \sum_j \mathbb{E}[C_j^{\mathrm{SEPT}}] = m + \sum_{j=m+1}^{n}\frac{j}{m} = m + k + \frac{k(k+1)}{2m}.$$

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 $1 + \lfloor k/m \rfloor$ jobs to $m - k + m\lfloor k/m \rfloor$ machines and $1 + \lceil k/m \rceil$ jobs to $k - m\lfloor k/m \rfloor$ machines. Hence, there are $m$ jobs with $\mathbb{E}[C_j] = l$ for each $l$ in the range $1, \ldots, 1 + \lfloor k/m \rfloor$, and $k - m\lfloor k/m \rfloor$ jobs with $\mathbb{E}[C_j] = 2 + \lfloor k/m \rfloor$. The expected performance for the best fixed assignment policy FIX is

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

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

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

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

Note 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, or 4, we use smaller numbers $k = k(m)$; namely, $k(2) = 1$, $k(3) = 2$, and $k(4) = 2$, and obtain the lower bounds $8/7 \approx 1.14$, $7/6 \approx 1.16$, and $32/27 \approx 1.18$.

**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., nonexponential) 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 $\mathbb{E}[\sum w_j C_j]=3n^2/4+o(n^2)$. 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 $\mathbb{E}[\mathrm{OPT}(I)]=n^2/2+o(n^2)$. The ratio of both values tends to 3/2 if the number of jobs tends to infinity.

Note, however, that this result is less meaningful in comparison to the performance bound of Theorem 4.1, which depends on an upper bound $\Delta$ 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 $P|r_j|\mathbb{E}[\sum w_j C_j]$. 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. In addition, 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 $\alpha$-Shift-WSEPT rule as introduced in §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

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

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

The crucial observation is that the $\alpha$-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 $\alpha$-Shift-WSEPT rule, and the so-defined policy is a feasible SOS policy.

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

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

*Here, $\Delta$ is such that $\mathrm{Var}[P_j]/\mathbb{E}[P_j]^2\le\Delta$ for all jobs $j$. In particular, because all processing times $P_j$ are $\delta$-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

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

and the expected value of MinIncrease can be bounded by

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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 Möhring et al. [21] for the traditional stochastic problem. More generally, for $\delta$-NBUE processing times, optimizing the term $\max\{1+\delta/\alpha, \alpha+\delta(2m-1)/m\}$ for $\alpha$ 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 $\alpha$ 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 (Correa and Wagner [8]). In fact, it matches the competitive ratio of the deterministic parallel machine version of $\alpha$-Shift-WSEPT from Megow and Schulz [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 assigns 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 [26]. 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 SOS 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 (SOS) problem on parallel machines with release dates,* $P|r_j|\mathbb{E}[\sum w_j C_j]$. *Given that all processing times are $\delta$-NBUE, the* RandAssign *policy running $\alpha$-Shift-WSEPT on each single machine is a $\rho$-approximation, where*

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

*Here, $\Delta$ is such that $\mathrm{Var}[P_j]/\mathbb{E}[P_j]^2 \le \Delta$ for all jobs $j$. In particular, because all processing times $P_j$ are $\delta$-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

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

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

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

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

$$\begin{aligned}\mathbb{E}[\mathrm{RA}(I)] &= \sum_j w_j \mathbb{E}[C_j] \\ &\le \left(1 + \frac{\delta}{\alpha}\right) \sum_j w_j r'_j + \sum_j w_j \sum_{k \in H(j)} \frac{\mathbb{E}[P_k]}{m} + \frac{m-1}{m} \sum_j w_j \mathbb{E}[P_j].\end{aligned}$$

This bound equals the upper bound that we achieved on the expected performance of the (modified) 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. □

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

Lemma A.1. *Let $X$ be a $\delta$-NBUE random variable, and let $\mathrm{CV}[X] = \sqrt{\mathrm{Var}[X]}/\mathbb{E}[X]$ denote the coefficient of variation of $X$. Then, $\mathrm{CV}[X]^2 \le 2\delta - 1$.*

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 nonnegative $\delta$-NBUE random variable, with cumulative distribution function $F$ and density $f$.

By definition of conditional expectation, we know that

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

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 $\delta$-NBUE, i.e., $\mathbb{E}[X - t \mid X > t] \le \delta \mathbb{E}[X]$, it follows from (5) and (6) that

$$\int_{x=t}^\infty 1 - F(x)\,dx = \mathbb{E}[X - t \mid X > t](1 - F(t)) \le \delta \mathbb{E}[X](1 - F(t)). \tag{7}$$

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

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

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

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

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

$$\mathbb{E}[X^2] \le 2\delta \mathbb{E}[X]^2.$$

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

$$\mathrm{CV}[X]^2 = \frac{\mathbb{E}[X^2] - \mathbb{E}[X]^2}{\mathbb{E}[X]^2} \le 2\delta - 1. \quad \square$$

**Acknowledgments.** The authors thank the two anonymous referees for valuable comments and Martin Skutella for pointing out that the (modified) MININCREASE algorithm can be interpreted as the derandomization of a randomized algorithm. Thanks to Rolf H. Möhring for helpful discussions and to Andreas S. Schulz and Sven O. Krumke for pointing out some flaws in an earlier version of this paper. Research was partially supported by the DFG Research Center, MATHEON, *Mathematics for Key Technologies* in Berlin. An extended abstract with parts of this work appeared in the *WAOA* 2004 *Conference Proceedings on Approximation and Online Algorithms* (Megow et al. [19]). Parts of this work were done while Tjark Vredeveld was with the Konrad-Zuse-Zentrum für Informationstechnik, Berlin, Germany.

## References

[1] Afrati, F. N., E. Bampis, C. Chekuri, D. R. Karger, C. Kenyon, S. Khanna, I. Milis, M. Queyranne, M. Skutella, C. Stein, M. Sviridenko. 1999. Approximation schemes for minimizing average weighted completion time with release dates. *Proc. 40th Annual Sympos. on Foundations of Comput. Sci.*, New York. IEEE Computer Society, Los Alamitos, CA, 32–43.

[2] Anderson, E. J., C. N. Potts. 2004. On-line scheduling of a single machine to minimize total weighted completion time. *Math. Oper. Res.* **29** 686–697.

[3] Borodin, A., R. El-Yaniv. 1998. *Online Computation and Competitive Analysis*. Cambridge University Press, Cambridge, UK.

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

[5] Chakrabarti, S., S. Muthukrishnan. 1996. Resource scheduling for parallel database and scientific applications. *Proc. 8th Annual ACM Sympos. on Parallel Algorithms and Architectures*, Padua, Italy. ACM Press, New York, 329–335.

[6] Chekuri, C., R. Johnson, R. Motwani, B. Natarajan, B. Rau, M. Schlansker. 1996. An analysis of profile-driven instruction level parallel scheduling with application to super blocks. *Proc. 29th IEEE/ACM Internat. Sympos. on Microarchitecture*, Paris, France. IEEE Computer Society, Los Alamitos, CA, 58–69.

[7] Chou, M. C., H. Liu, M. Queyranne, D. Simchi-Levi. 2006. On the asymptotic optimality of a simple on-line algorithm for the stochastic single machine weighted completion time problem and its extensions. *Oper. Res.* **54**(3) 464–474.

[8] Correa, J., M. Wagner. 2005. LP-based online scheduling: From single to parallel machines. M. Jünger, V. Kaibel, eds. *Proc. 8th Integer Programming and Combinatorial Optimization Conf.*, Berlin, Germany. *Lecture Notes in Computer Science,* Vol. 3509. Springer-Verlag, 196–209.
[9] Dempster, M. A. H., J. K. Lenstra, A. H. G. Rinnooy Kan, eds. 1982. *Deterministic and Stochastic Scheduling*. D. Reidel Publishing Company, Dordrecht, The Netherlands.
[10] Eastman, W., S. Even, I. Isaacs. 1964. Bounds for the optimal scheduling of $n$ jobs on $m$ processors. *Management Sci.* **11** 268–279.
[11] Fiat, A., G. J. Woeginger. 1999. On-line scheduling on a single machine: Minimizing the total completion time. *Acta Informatica* **36** 287–293.
[12] Graham, R. L., E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan. 1979. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Ann. Discrete Math.* **5** 287–326.
[13] Hall, W. J., J. A. Wellner. 1981. Mean residual life. M. Csörgö, D. A. Dawson, J. N. K. Rao, A. K. Md. E. Saleh, eds. *Proc. Internat. Sympos. Statist. Related Topics*. Ottawa, Ontario, Canada. North-Holland, Amsterdam, The Netherlands, 169–184.
[14] Hoogeveen, H., A. P. A. Vestjens. 1996. Optimal on-line algorithms for single-machine scheduling. W. H. Cunningham, S. T. McCormick, M. Queyranne, eds. *Proc. 5th Integer Programming and Combinatorial Optimization Conf.*, Vancouver, British Columbia, Canada. *Lecture Notes in Computer Science*, Vol. 1084. Springer-Verlag, 404–414.
[15] Karlin, A., M. Manasse, L. Rudolph, D. Sleator. 1988. Competitive snoopy paging. *Algorithmica* **3** 70–119.
[16] Koutsoupias, E., C. H. Papadimitriou. 2000. Beyond competitive analysis. *SIAM J. Comput.* **30** 300–317.
[17] Lenstra, E. L., A. H. G. Rinooy Kan, P. Brucker. 1977. Complexity of machine scheduling problems. *Ann. Discrete Math.* **1** 243–362.
[18] Megow, N., A. S. Schulz. 2004. On-line scheduling to minimize average completion time revisited. *Oper. Res. Lett.* **32** 485–490.
[19] Megow, N., M. Uetz, T. Vredeveld. 2005. Stochastic online scheduling on parallel machines. G. Persiano, R. Solis-Oba, eds. *Proc. Second Workshop on Approximation and Online Algorithms*, Bergen, Norway. *Lecture Notes in Computer Science*, Vol. 3351. Springer-Verlag, 167–180.
[20] Möhring, R. H., F. J. Radermacher, G. Weiss. 1984. Stochastic scheduling problems I: General strategies. *ZOR—Zeitschrift für Oper. Res.* **28** 193–260.
[21] Möhring, R. H., A. S. Schulz, M. Uetz. 1999. Approximation in stochastic scheduling: The power of LP-based priority policies. *J. ACM* **46** 924–942.
[22] Pruhs, K., J. Sgall, E. Torng. 2004. Online scheduling. J. Leung, ed. *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, Chapter 15. CRC Press, Boca Raton, FL.
[23] Rothkopf, M. H. 1966. Scheduling with random service times. *Management Sci.* **12** 703–713.
[24] Scharbrodt, M., T. Schickinger, A. Steger. 2002. A new average case analysis for completion time scheduling. *Proc. 34th Annual ACM Sympos. on the Theory of Comput.*, Montreal, Quebec, Canada. ACM Press, New York, 170–178.
[25] Schulz, A. S. 2005. New old algorithms for stochastic scheduling. S. Albers, R. H. Möhring, G. Ch. Pflug, R. Schultz, eds. *Algorithms for Optimization with Incomplete Information. Dagstuhl Seminar Proceedings, 05031*. Internationales Begegnungs- und Forschungszentrum (IBFI), Schloss Dagstuhl, Germany.
[26] Schulz, A. S., M. Skutella. 2002. Scheduling unrelated machines by randomized rounding. *SIAM J. Discrete Math.* **15** 450–469.
[27] Skutella, M., M. Uetz. 2005. Stochastic machine scheduling with precedence constraints. *SIAM J. Comput.* **34** 788–802.
[28] Skutella, M., G. J. Woeginger. 2000. A PTAS for minimizing the total weighted completion time on identical parallel machines. *Math. Oper. Res.* **25** 63–75.
[29] Sleator, D., R. Tarjan. 1985. Amortized efficiency of list update and paging rules. *Comm. ACM* **28** 202–208.
[30] Smith, W. 1956. Various optimizers for single-stage production. *Naval Res. Logist. Quart.* **3** 59–66.
[31] Souza, A., A. Steger. 2004. The expected competitive ratio for weighted completion time scheduling. V. Diekert, M. Habib, eds. *Proc. 21st Sympos. on Theoret. Aspects of Comput. Sci.*, Montpellier, France. *Lecture Notes in Computer Science,* Vol. 2996. Springer-Verlag, 620–631.
[32] Uetz, M. 2002. *Algorithms for Deterministic and Stochastic Scheduling*. Cuvillier Verlag, Göttingen, Germany.
[33] Vestjens, A. P. A. 1997. On-line machine scheduling. Doctoral dissertation, Eindhoven University of Technology, Eindhoven, The Netherlands.
[34] Weiss, G. 1990. Approximation results in parallel machines stochastic scheduling. *Ann. Oper. Res.* **26** 195–242.
[35] Weiss, G. 1992. Turnpike optimality of Smith's rule in parallel machines stochastic scheduling. *Math. Oper. Res.* **17** 255–270.
[36] Weiss, G., M. Pinedo. 1980. Scheduling tasks with exponential service times on non-identical processors to minimize various cost functions. *J. Appl. Probab.* **17** 187–202.