

Approximation in Preemptive Stochastic Online Scheduling

Nicole Megow^{1,*} and Tjark Vredeveld^{2,**}

¹ Technische Universität Berlin, Institut für Mathematik, Strasse des 17. Juni 136,
10623 Berlin, Germany

`nmegow@math.tu-berlin.de`

² Maastricht University, Department of Quantitative Economics, P.O. Box 616,
6200 MD Maastricht, The Netherlands

`t.vredeveld@ke.unimaas.nl`

Abstract. We present a first constant performance guarantee for preemptive stochastic scheduling to minimize the sum of weighted completion times. For scheduling jobs with release dates on identical parallel machines we derive a policy with a guaranteed performance ratio of 2 which matches the currently best known result for the corresponding deterministic online problem.

Our policy applies to the recently introduced stochastic online scheduling model in which jobs arrive online over time. In contrast to the previously considered nonpreemptive setting, our preemptive policy extensively utilizes information on processing time distributions other than the first (and second) moments. In order to derive our result we introduce a new nontrivial lower bound on the expected value of an unknown optimal policy that we derive from an optimal policy for the basic problem on a single machine without release dates. This problem is known to be solved optimally by a Gittins index priority rule. This priority index also inspires the design of our policy.

1 Introduction

Stochastic scheduling problems have attracted researchers for about four decades, see e.g. [20]. A full range of articles concerns criteria that guarantee the optimality of simple policies for special scheduling problems. Only recently research interest has also focused on approximative policies [18, 26, 15, 21] for nonpreemptive scheduling. We are not aware of any approximation results for preemptive problems. Previous approaches, based on linear programming relaxations, do not seem to carry over to the preemptive setting. In this paper, we give a first approximation result for preemptive stochastic scheduling to minimize the weighted sum of completion times. We prove an approximation guarantee of 2 even in the recently introduced more general model of stochastic online scheduling [15, 4].

* Supported by the DFG Research Center MATHEON *Mathematics for key technologies* in Berlin.

** Research partially supported by METEOR, the Maastricht research school of Economics of Technology and Organizations.

This guarantee matches exactly the currently best known approximation result for the deterministic online version of this problem [14].

Problem definition. Let $J = \{1, 2, \dots, n\}$ be a set of jobs which must be scheduled on m identical parallel machines. Each of the machines can process at most one job at a time, and any job can be processed by no more than one machine at a time. Each job j has associated a positive weight w_j and an individual release date $r_j \geq 0$ before which it is not available for processing. We allow preemption which means that the processing of a job may be interrupted and resumed later on the same or another machine.

The stochastic component in the model we consider is the uncertainty about processing times. Any job j must be processed for P_j units of time, where P_j is a random variable. By $\mathbb{E}[P_j]$ we denote the expected value of the processing time of job j and by p_j a particular realization of P_j . We assume that all random variables of processing times are stochastically independent and follow discrete probability distributions. With the latter restriction and a standard scaling argument, we may assume w.l.o.g. that P_j attains integral values in the set $\Omega_j \subseteq \{1, 2, \dots, M_j\}$ and that all release dates are integral. The sample space of all processing times is denoted by $\Omega = \Omega_1 \times \dots \times \Omega_n$.

The objective is to schedule all jobs so as to minimize the total weighted completion time of the jobs, $\sum_{j \in J} w_j C_j$, in expectation, where C_j denotes the completion time of job j . Adopting the well-known three-field classification scheme by Graham et al. [8], we denote the problem by $P | r_j, pmtn | \mathbb{E}[\sum w_j C_j]$.

The solution of a stochastic scheduling problem is not a simple schedule, but a so-called *scheduling policy*. We follow the notion of scheduling policies as proposed by Möhring, Radermacher, and Weiss [17]. Roughly spoken, a scheduling policy makes scheduling decisions at certain *decision time points* t , and these decisions are based on information on the observed past up to time t , as well as the a priori knowledge of the input data of the problem. The policy, however, must not anticipate information about the future, such as the actual realizations p_j of the processing times of the jobs that have not yet been completed by time t .

Additionally, we restrict ourselves to so-called *online policies*, which learn about the existence and characteristics of a job only at its individual release date. This means for an online policy that it must not anticipate the arrival of a job at any time earlier than its release date. At this point in time, the job with its processing time distribution and weight is revealed. Thus, our policies are required to be online and non-anticipatory. However, an optimal policy can be offline as long as it is non-anticipatory. We refer to Megow, Uetz, and Vredeveld [15] for a more detailed discussion on stochastic online policies. In this paper, we concentrate on (online) approximation policies. As suggested in [15] we use a generalized definition of approximation guarantees from the stochastic scheduling setting [17].

Definition 1. A (online) stochastic policy Π is a ρ -approximation, for some $\rho \geq 1$, if for all problem instances \mathcal{I} ,

$$\mathbb{E}[\Pi(\mathcal{I})] \leq \rho \mathbb{E}[\text{OPT}(\mathcal{I})],$$

where $\mathbb{E}[\Pi(\mathcal{I})]$ and $\mathbb{E}[\text{OPT}(\mathcal{I})]$ denote the expected values that the policy Π and an optimal non-anticipatory offline policy, respectively, achieve on a given instance \mathcal{I} . The value ρ is called performance guarantee of policy Π .

Previous work. Stochastic scheduling has been considered for more than 30 years. Some of the first results on preemptive scheduling that can be found in literature are by Chazan, Konheim, and Weiss [2] and Konheim [11]. They formulated sufficient and necessary conditions for a policy to solve optimally the single machine problem where all jobs become available at the same time. Later Sevcik [24] developed an intuitive method for creating optimal schedules (in expectation). He introduces a priority policy that relies on an index which can be computed for each job based on the properties of a job, but not on other jobs.

Gittins [6] showed that this priority index is a special case of his Gittins index [6, 7]. Later in 1995, Weiss [30] formulated Sevcik's priority index again in terms of the Gittins index and names it a *Gittins index priority policy*. He also provided a different proof of the optimality of this priority policy, based on the work conservation invariance principle. Weiss covers a more general problem than the one considered here and in [2, 11, 24]: The holding costs (weights) of a job are not deterministic constants, but may vary during the processing of a job. At each state these holding costs are random variables.

For more general scheduling problems with release dates and/or multiple machines, no optimal policies are known. Instead, literature reflects a variety of research on restricted problems as those with special probability distributions for processing times or special job weights [1, 29, 19, 5, 9, 30].

For the parallel machine problem without release dates it is worthwhile to mention that Weiss [30] showed that the Gittins index priority policy above is asymptotically optimal and has a turnpike property, which means that there is a bound on the number of times that the policy differs from an optimal policy.

Optimal policies have only been found for a few special cases of stochastic scheduling problems. Already the deterministic counterpart of the scheduling problem we consider, is well-known to be NP-hard, even in the case that there is only a single processor or if all release dates are equal [12, 13]. Therefore, recently attempts have been made on obtaining approximation algorithms which have been successful in the nonpreemptive setting. Möhring, Schulz, and Uetz [18] derived first constant-factor approximations for the nonpreemptive problem with and without release dates. They were improved later by Megow et al. [15] and Schulz [21] for a more general setting. Skutella and Uetz [26] complemented the first approximation results by constant-approximative policies for scheduling with precedence constraints. In general, all given performance guarantees for nonpreemptive policies depend on a parameter defined by expected values of processing times and the coefficients of variation.

In contrast to stochastic scheduling, in a deterministic online model is assumed that no information about any future job arrival is available. However, once a job arrives, its weight and actual processing time become known immediately. The performance of online algorithms is typically assessed by their competitive

ratio [10, 27]. An algorithm is called ρ -competitive if it achieves for any instance a solution with a value at most ρ times the value of an optimal offline solution.

In this deterministic online model, Sitters [25] gave a 1.56-competitive algorithm for preemptive scheduling on a single machine. This is the currently best known result and it improved upon an earlier result by Schulz and Skutella [22]; they generalized the classical *Smith rule* [28] to the problem of scheduling jobs with individual release dates and achieved a competitive ratio of 2. This algorithm has been generalized further to the multiple machine problem without loss of performance by Megow and Schulz [14]. As far as we know, there is no randomized online algorithm known with a provable competitive ratio less than 2 for this problem. In contrast, Schulz and Skutella [23] provide a $4/3$ -competitive algorithm for the single machine problem.

Recently, the stochastic scheduling model as we consider it in this paper has been investigated; all obtained results which include asymptotic optimality [4] and approximation guarantees for deterministic [15] and randomized policies [15, 21] address nonpreemptive scheduling.

Our contribution. We derive a first constant performance guarantee for preemptive stochastic scheduling. For jobs with general processing time distributions and individual release dates, we give a 2-approximation policy for multiple machines. This performance guarantee matches the currently best known result in deterministic online scheduling although we consider a more general model. In comparison to the previously known results in this model in a nonpreemptive setting, our result stands out by being constant and independent of the probability distribution of processing times.

In general our policy is not optimal. However, on restricted problem instances it coincides with policies whose optimality is known. If processing times are exponentially distributed and release dates are absent, our policy coincides with the *Weighted shortest expected processing time* (WSEPT) rule. This classical policy is known to be optimal if all weights are equal [1] or, more general, if they are *agreeable*, which means that for any two jobs i, j holds that $\mathbb{E}[P_i] < \mathbb{E}[P_j]$ implies $w_i \leq w_j$ [9]. If only one machine is available, we solve the weighted problem $1 | pmtn | \mathbb{E}[\sum w_j C_j]$ optimally by utilizing the Gittins index priority policy [11, 24, 30]. Moreover, Pinedo showed in [19] that in presence of release dates the WSEPT rule is optimal if all processing times are exponentially distributed.

Our result is based on a new nontrivial lower bound for the preemptive stochastic scheduling problem. This bound is derived by borrowing ideas for a *fast single machine relaxation* from Chekuri et al. [3]. The crucial ingredient to our result is then the application of a Gittins index priority policy which is optimal to a relaxed version of our fast single machine relaxation.

2 A Gittins Index Priority Policy

As mentioned in the introduction, a Gittins index priority policy solves the single machine problem with trivial release dates to optimality, see [11, 24, 30]. This result is crucial for the approximation result we give in this paper; it inspires

the design of our policy and it serves as a tool for bounding the expected value of an unknown optimal policy for the more general problem that we consider. Therefore, we introduce in this section the Gittins index priority rule and some useful notation.

Given that a job j has been processed for y time units, we define the *expected investment* of processing this job for q time units or up to completion, whichever comes first, as

$$I_j(q, y) = \mathbb{E} [\min\{P_j - y, q\} \mid P_j > y].$$

The ratio of the weighted probability that this job is completed within the next q time units over the expected investment, is the basis of the Gittins index priority rule. Therefore, we define this as the *rank* of a sub-job of length q of job j , after it has completed y units of processing:

$$R_j(q, y) = \frac{w_j \Pr [P_j - y \leq q \mid P_j > y]}{I_j(q, y)}.$$

For a given (unfinished) job j and attained processing time y , we are interested in the maximal rank it can achieve. We call this the Gittins index, or rank, of job j , after it has been processed for y time units.

$$R_j(y) = \max_{q \in \mathbb{R}^+} R_j(q, y).$$

The length of the sub-job achieving the maximal rank is denoted as

$$q_j(y) = \max\{q \in \mathbb{R}^+ : R_j(q, y) = R_j(y)\}.$$

With these definitions, we define the Gittins index priority policy.

Algorithm 1. Gittins index priority policy (GIPP)

At any time t , process an unfinished job j with currently highest rank $R_j(y_j(t))$, where $y_j(t)$ denotes the amount of processing that has been done on job j by time t . Break ties by choosing the job with the smallest job index.

Theorem 1 ([11, 24, 30]). *The Gittins index priority policy (GIPP) solves the preemptive stochastic scheduling problem $1 \mid pmtn \mid \mathbb{E} [\sum w_j C_j]$ optimally.*

The following properties of the Gittins indices and the lengths of sub-jobs achieving the Gittins index are well known, see [7, 30]. In parts, they have been derived earlier in the scheduling context by [11] and [24].

Proposition 1 ([7, 30]) *Consider a job j that has been processed for y time units. Then, for any $0 < \gamma < q_j(y)$ holds*

$$R_j(y) \leq R_j(y + \gamma), \tag{1}$$

$$q_j(y + \gamma) \leq q_j(y) - \gamma, \tag{2}$$

$$R_j(y + q_j(y)) \leq R_j(y). \tag{3}$$

Let us denote the sub-job of length $q_j(y)$ that causes the maximal rank $R_j(y)$, a *quantum* of job j . We now split a job j into a set of n_j quanta, denoted by tuples (j, i) , for $i = 1, \dots, n_j$. The processing time y_{ji} that a job j has attained up to a quantum (j, i) and the length of each quantum, q_{ji} , are recursively defined as $y_{j1} = 0$, $q_{ji} = q_j(y_{ji})$, and $y_{j,i+1} = y_{j,i} + q_{ji}$. By Proposition 1(1), we know that, while processing a quantum, the rank of the job does not decrease, whereas Proposition 1(3) and the definition of $q_j(y)$ tell us that the rank is strictly lower at the beginning of the next quantum. Hence, once a quantum has been started GIPP will process it for its complete length or up to the completion of the job, whatever comes first. Thus, a job is preempted only at the end of a quantum. Obviously, the policy GIPP processes job quanta nonpreemptively in non-increasing order of their ranks.

Based on the definitions above, we define the set $H(j, i)$ of all quanta that precede quantum (j, i) in the GIPP order. Let \mathcal{Q} be the set of all quanta, i. e., $\mathcal{Q} = \{(k, l) \mid k = 1, \dots, n, l = 1, \dots, n_k\}$, then

$$H(j, i) = \{(k, l) \in \mathcal{Q} \mid R_k(y_{kl}) > R_j(y_{ji})\} \cup \{(k, l) \in \mathcal{Q} \mid R_k(y_{kl}) = R_j(y_{ji}) \wedge k \leq j\}.$$

As the Gittins index of a job is decreasing with every finished quantum 1(3), we know that $H(j, h) \subseteq H(j, i)$, for $h \leq i$. In order to uniquely relate higher priority quanta to one quantum of a job, we introduce the notation $H'(j, i) = H(j, i) \setminus H(j, i - 1)$, where we define $H(j, 0) = \emptyset$. Note that the quantum (j, i) is also contained in the set of its higher priority quanta $H'(j, i)$. In the same manner, we define the set of lower priority quanta as $L(j, i) = \mathcal{Q} \setminus H(j, i)$.

With these definitions and the observations above we can give a closed formula for the expected objective value of GIPP.

Lemma 2. *The optimal policy for the scheduling problem $1 \mid pmtn \mid \mathbb{E}[\sum w_j C_j]$, GIPP, achieves the expected objective value of*

$$\mathbb{E}[\text{GIPP}] = \sum_j w_j \sum_{i=1}^{n_j} \sum_{(k,l) \in H'(j,i)} \Pr [P_j > y_{ji} \wedge P_k > y_{kl}] \cdot I_k(q_{kl}, y_{kl}).$$

Proof. Consider a realization of processing times $p \in \Omega$ and a job j . Let i_p be the index of the quantum in which job j finishes, i. e., $y_{j i_p} < p_j \leq y_{j i_p} + q_{j i_p}$. The policy GIPP processes quanta of jobs that have not completed nonpreemptively in non-increasing order of their ranks. Hence,

$$C_j(p) = \sum_{(k,l) \in H(j, i_p) : p_k > y_{kl}} \min\{q_{kl}, p_k - y_{kl}\}. \tag{4}$$

For an event \mathcal{E} , let $\chi(\mathcal{E})$ be an indicator random variable which equals 1 if and only if the event \mathcal{E} occurs. The expected value of $\chi(\mathcal{E})$ equals then the probability with that the event \mathcal{E} occurs, i. e., $\mathbb{E}[\chi(\mathcal{E})] = \Pr[\mathcal{E}]$. Additionally, we denote by ξ_{kl} the special indicator random variable for the event $P_k > y_{kl}$.

We take expectations on both sides of equation (4) over all realizations. This yields

$$\begin{aligned}
 \mathbb{E}[C_j] &= \mathbb{E} \left[\sum_{h: y_{jh} < P_j \leq y_{j,h+1}} \sum_{(k,l) \in H(j,h): P_k > y_{kl}} \min\{q_{kl}, P_k - y_{kl}\} \right] \\
 &= \mathbb{E} \left[\sum_{h=1}^{n_j} \chi(y_{jh} < P_j \leq y_{j,h+1}) \sum_{(k,l) \in H(j,h)} \xi_{kl} \cdot \min\{q_{kl}, P_k - y_{kl}\} \right] \\
 &= \mathbb{E} \left[\sum_{h=1}^{n_j} \chi(y_{jh} < P_j \leq y_{j,h+1}) \sum_{i=1}^h \sum_{(k,l) \in H'(j,i)} \xi_{kl} \cdot \min\{q_{kl}, P_k - y_{kl}\} \right] \\
 &= \mathbb{E} \left[\sum_{i=1}^{n_j} \sum_{h=i}^{n_j} \chi(y_{jh} < P_j \leq y_{j,h+1}) \sum_{(k,l) \in H'(j,i)} \xi_{kl} \cdot \min\{q_{kl}, P_k - y_{kl}\} \right] \\
 &= \mathbb{E} \left[\sum_{i=1}^{n_j} \chi(y_{ji} < P_j) \sum_{(k,l) \in H'(j,i)} \xi_{kl} \cdot \min\{q_{kl}, P_k - y_{kl}\} \right] \\
 &= \mathbb{E} \left[\sum_{i=1}^{n_j} \xi_{ji} \sum_{(k,l) \in H'(j,i)} \xi_{kl} \cdot \min\{q_{kl}, P_k - y_{kl}\} \right]. \tag{5}
 \end{aligned}$$

The equalities follow from an index rearrangement and the facts that $H(j, h) = \cup_{i=1}^h H'(j, i)$ for any h and that n_j is an upper bound on the number of quanta of job j .

For jobs $k \neq j$, the processing times P_j and P_k are independent random variables and thus, the same holds for their indicator random variables ξ_{ji} and ξ_{kl} for any i, l . Using linearity of expectation, we rewrite (5) as

$$\begin{aligned}
 &= \sum_{i=1}^{n_j} \sum_{(k,l) \in H'(j,i)} \mathbb{E}[\xi_{ji} \cdot \xi_{kl} \cdot \min\{q_{kl}, P_k - y_{kl}\}] \\
 &= \sum_{i=1}^{n_j} \sum_{(k,l) \in H'(j,i)} \sum_x x \cdot \Pr[\xi_{ji} = \xi_{kl} = 1 \wedge \min\{q_{kl}, P_k - y_{kl}\} = x] \\
 &= \sum_{i=1}^{n_j} \sum_{(k,l) \in H'(j,i)} \sum_x x \cdot \Pr[\xi_{ji} = \xi_{kl} = 1] \cdot \Pr[\min\{q_{kl}, P_k - y_{kl}\} = x \mid \xi_{kl} = 1] \\
 &= \sum_{i=1}^{n_j} \sum_{(k,l) \in H'(j,i)} \Pr[P_j > y_{ji} \wedge P_k > y_{kl}] \cdot \mathbb{E}[\min\{q_{kl}, P_k - y_{kl}\} \mid P_k > y_{kl}] \\
 &= \sum_{i=1}^{n_j} \sum_{(k,l) \in H'(j,i)} \Pr[P_j > y_{ji} \wedge P_k > y_{kl}] \cdot I_k(q_{kl}, y_{kl}),
 \end{aligned}$$

where the third equality follows from conditional probability and the fact that either $j \neq k$, thus ξ_{ji} and ξ_{kl} are independent, or $(j, i) = (k, l)$ and thus the variables ξ_{ji} and ξ_{kl} are the same. Weighted summation over all jobs concludes the proof. \square

3 A New Lower Bound on Parallel Machines

For the scheduling problem $P | r_j, pmtn | \mathbb{E} [\sum w_j C_j]$ and most of its relaxations, optimal offline policies and the corresponding expected objective values are unknown. Therefore, we use lower bounds on the optimal value in order to compare the expected outcome of a policy with the expected outcome $\mathbb{E}[\text{OPT}]$ of an unknown optimal policy OPT . The trivial bound $\mathbb{E}[\text{OPT}] \geq \sum_j w_j (r_j + \mathbb{E}[P_j])$ is unlikely to suffice proving constant approximation guarantees. However, we are not aware of any other bounds known for the general preemptive problem. LP-based approaches as they are used in the non-preemptive setting [18, 26, 4, 15, 21] do not transfer directly.

We derive in this section a new non-trivial lower bound for preemptive stochastic scheduling on parallel machines. We utilize the knowledge of GIPP’s optimality for the single machine problem without release dates, see Theorem 1. To do so, we show first that the *fast single machine* relaxation introduced in deterministic online environment [3] applies in the stochastic setting as well.

Lemma 3. *Denote by \mathcal{I} an instance of the problem $P | r_j, pmtn | \mathbb{E} [\sum w_j C_j]$, and let \mathcal{I}' be the same instance to be scheduled on a single machine of speed m times the speed of the machines used for scheduling instance \mathcal{I} . The optimal single machine policy OPT_1 yields an expected value $\mathbb{E}[\text{OPT}_1(\mathcal{I}')]$ on instance \mathcal{I}' . Then, for any parallel machine policy Π holds*

$$\mathbb{E}[\Pi(\mathcal{I})] \geq \mathbb{E}[\text{OPT}_1(\mathcal{I}')] .$$

Proof. Given a parallel machine policy Π , we provide a policy Π' for the fast single machine that yields an expected objective value $\mathbb{E}[\Pi'(\mathcal{I}')] \leq \mathbb{E}[\Pi(\mathcal{I})]$ for any instance \mathcal{I} . Then the lemma follows since an optimal policy OPT_1 on the single machine yields an expected objective value $\mathbb{E}[\text{OPT}_1(\mathcal{I}')] \leq \mathbb{E}[\Pi'(\mathcal{I}')]$.

We construct policy Π' by letting its first decision point coincide with the first decision point of policy Π (the earliest release date). At any of its decision points, Π' can compute the jobs to be scheduled by policy Π and due to the fact that the processing times of all jobs are discrete random variables, it computes the earliest possible completion time of these jobs, in the parallel machine schedule. The next decision point of Π' , is the minimum of these possible completion times and the next decision point of Π . Between two consecutive decision points of Π' , the policy schedules the same set of jobs that Π schedules, for the same amount of time. This is possible as the single machine on which Π' operates works m times as fast.

In this way, we ensure that all job completions in the parallel machine schedule obtained by Π , coincide with a decision point of policy Π' . Moreover, as Π' schedules the same set of jobs as Π between two decision points, any job that completes its processing at a certain time t in the schedule of Π , will also be completed by time t in the schedule of Π' . □

With this relaxation, we derive a lower bound on the expected optimal value.

Theorem 2. *The expected value of an optimal policy OPT for the parallel machine problem \mathcal{I} is bounded by*

$$\mathbb{E}[\text{OPT}(\mathcal{I})] \geq \frac{1}{m} \sum_j w_j \sum_{i=1}^{n_j} \sum_{(k,\ell) \in H'(j,i)} \Pr [P_j > y_{ji} \wedge P_k > y_{k\ell}] \cdot I_k(q_{k\ell}, y_{k\ell}).$$

Proof. We consider the fast single machine instance \mathcal{I}' as introduced in the previous lemma and relax it further to instance \mathcal{I}'_0 by setting all release dates equal. By Theorem 1, the resulting problem can be solved optimally by GIPP. With Lemma 3 we have then

$$\mathbb{E}[\text{OPT}(\mathcal{I})] \geq \mathbb{E}[\text{OPT}_1(\mathcal{I}')] \geq \mathbb{E}[\text{GIPP}(\mathcal{I}'_0)]. \tag{6}$$

By Lemma 2 we know

$$\mathbb{E}[\text{GIPP}(\mathcal{I}'_0)] = \sum_j w_j \sum_{i=1}^{n_j} \sum_{(k,\ell) \in H'(j,i)} \Pr [P'_j > y'_{ji} \wedge P'_k > y'_{k\ell}] \cdot I'_k(q'_{k\ell}, y'_{k\ell}), \tag{7}$$

where the dashes indicate the modified variables in the fast single machine instance \mathcal{I}'_0 . By definition holds $P'_j = P_j/m$ for any job j as well as $\Pr [P_j > x] = \Pr [P'_j > x/m]$, and the probability $\Pr [P_j - y = x \mid P_j > y]$ for the remaining processing time after y units of processing remains the same on the fast machine. Moreover, the investment $I'_j(q', y')$ for any sub-job of length $q' = q/m$ of job $j \in \mathcal{I}'$ after it has received $y' = y/m$ units of processing coincides with

$$\begin{aligned} I'_j(q', y') &= \mathbb{E}[\min\{P'_j - y', q'\} \mid P'_j > y'] \\ &= \frac{1}{m} \mathbb{E}[\min\{P_j - y, q\} \mid P_j > y] = \frac{1}{m} I_j(q, y). \end{aligned}$$

We conclude that the partition of jobs into quanta in instance \mathcal{I} immediately gives the partition for the fast single machine instance \mathcal{I}' . Each quantum (j, i) of job j maximizes the rank $R_j(q, y_{ji})$ and thus $q' = q/m$ maximizes the rank $R'_j(q/m, y/m) = R_j(q, y)/m$ on the single machine; thus, the quanta are simply shortened to an m -fraction of the original length, $q'_{ji} = q_{ji}/m$ and thus, $y'_{ji} = \sum_{l=1}^{i-1} q'_{jl} = y_{ji}/m$.

Combining these observations with (6) and (7) yields

$$\mathbb{E}[\text{OPT}(\mathcal{I})] \geq \frac{1}{m} \sum_j w_j \sum_{i=1}^{n_j} \sum_{(k,\ell) \in H'(j,i)} \Pr [P_j > y_{ji} \wedge P_k > y_{k\ell}] \cdot I_k(q_{k\ell}, y_{k\ell}).$$

□

Theorem 2 above and Lemma 2 imply immediately

Corollary 1. *The lower bound on the optimal preemptive policy for parallel machine scheduling on an instance \mathcal{I} equals an m -fraction of the expected value achieved by GIPP on the relaxed instance \mathcal{I}_0 without release dates but the same processing times to be scheduled on one machine, i. e.,*

$$\mathbb{E}[\text{OPT}(\mathcal{I})] \geq \frac{\mathbb{E}[\text{GIPP}(\mathcal{I}_0)]}{m}. \tag{8}$$

4 A 2-Approximation on Parallel Machines

Simple examples show that GIPP is not an optimal policy for scheduling problems with release dates and/or multiple machines. The following policy uses a modified version of GIPP where the rank of jobs is updated only after the completion of a quantum.

Algorithm 2. Follow Gittins Index Priority Policy (F-GIPP)

At any time t , process an available job j with highest rank $R_j(y_{j,k+1})$, where (j, k) is the last quantum of j that has completed, or $k = 0$ if no quantum of job j has been completed.

Note, that the decision time points in this policy are release dates and any time, when a quantum or a job completes. In contrast to the original Gittins index priority policy, F-GIPP considers only the rank $R_j(y_{ji} = \sum_{k=1}^{i-1} q_{jk})$ that a job had before processing quanta (j, i) even if (j, i) has been processing for some time less than q_{ji} . Informally speaking, the policy F-GIPP updates the ranks only after quantum completions and then follows GIPP.

This policy applied to a deterministic scheduling instance coincides with the P-WSPT rule by Megow and Schulz [14], which is a generalization of Smith’s optimal nonpreemptive single machine algorithm [28] to the deterministic counterpart of our scheduling problem without release dates. It has a competitive ratio of 2, and we prove the same performance guarantee for the more general stochastic online setting.

Theorem 3. *The online policy F-GIPP is a deterministic 2-approximation for the preemptive scheduling problem $P \mid r_j, pmtn \mid \mathbb{E}[\sum w_j C_j]$.*

Proof. This proof incorporates ideas from [14] applied to the more complex stochastic setting. Fix a realization $p \in \Omega$ of processing times and consider a job j and its completion time $C_j^{\text{F-GIPP}}(p)$. Job j is processing in the time interval $[r_j, C_j^{\text{F-GIPP}}(p)]$. We split this interval into two disjunctive sets of sub-intervals, $T(j, p)$ and $\overline{T}(j, p)$, respectively. Let $T(j, p)$ denote the set of sub-intervals in which job j is processing and $\overline{T}(j, p)$ contains the remaining sub-intervals. Denoting the total length of all intervals in a set T by $|T|$, we have

$$C_j^{\text{F-GIPP}}(p) = r_j + |T(j, p)| + |\overline{T}(j, p)|.$$

In intervals of the set $\overline{T}(j, p)$, no machine is idle and F-GIPP schedules only quanta with a higher priority than (j, i_p) , the final quantum of job j . Thus $|\overline{T}(j, p)|$ is maximized if all these quanta are scheduled between r_j and $C_j^{\text{F-GIPP}}(p)$ with an upper bound on the overall length of the total sum of quantum lengths on m machines. The total length of intervals in $T(j, p)$ is p_j and it follows

$$C_j^{\text{F-GIPP}}(p) \leq r_j + p_j + \frac{1}{m} \cdot \sum_{\substack{(k,l) \in H(j,i_p) : \\ p_k > y_{kl}}} \min\{q_{kl}, p_k - y_{kl}\}.$$

Weighted summation over all jobs and taking expectations on both sides give with the same arguments as in Lemma 2:

$$\begin{aligned} \sum_j w_j \mathbb{E} [C_j^{\text{F-GIPP}}] &\leq \sum_j w_j (r_j + \mathbb{E}[P_j]) \\ &\quad + \frac{1}{m} \cdot \sum_j w_j \sum_{i=1}^{n_j} \sum_{(k,l) \in H'(j,i)} \Pr [P_j > y_{ji} \wedge p_k > y_{kl}] \cdot I_k(q_{kl}, y_{kl}). \end{aligned}$$

Finally, we apply the trivial lower bound $\mathbb{E}[\text{OPT}] \geq \sum_j w_j (r_j + \mathbb{E}[P_j])$ and Theorem 2, and the approximation result follows. \square

In absence of release dates, our policy coincides with GIPP and is thus optimal on a single machine (Theorem 1). Nevertheless, for general input instances the approximation factor of 2 is best possible for F-GIPP which follows directly from a deterministic worst-case instance in [14].

Concluding remarks. In a full version of our paper [16], we introduce a second single machine policy, which deviates less from the original Gittins index priority rule than F-GIPP does. Thus, we use more information on the actual state of the set of known, unfinished jobs. This single machine policy can be immediately extended to the parallel machine setting by randomized machine assignment. For both policies, on the single and on multiple machines, we prove the performance guarantee of 2. Clearly, this result does not improve the approximation guarantee of F-GIPP in the previous section. But, while the analysis of F-GIPP is tight, we conjecture that the true approximation ratio of the new policy is less than 2.

References

1. 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*, 28:100–113, 1981.
2. D. Chazan, A. G. Konheim, and B. Weiss. A note on time sharing. *Journal of Combinatorial Theory*, 5:344–369, 1968.
3. C. Chekuri, R. Motwani, B. Natarajan, and C. Stein. Approximation techniques for average completion time scheduling. *SIAM Journal on Computing*, 31:146–166, 2001.
4. 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, 2006. *Operations Research*, to appear.
5. E. G. Coffman, M. Hofri, and G. Weiss. Scheduling stochastic jobs with a two point distribution on two parallel machines. *Probability in the Engineering and Informational Sciences*, 3:89–116, 1989.
6. J. C. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society, Series B*, 41:148–177, 1979.
7. J. C. Gittins. *Multi-armed Bandit Allocation Indices*. Wiley, New York, 1989.
8. R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5:287–326, 1979.

9. T. Kämpke. Optimal scheduling of jobs with exponential service times on identical parallel processors. *Operations Research*, 37(1):126–133, 1989.
10. A. Karlin, M. Manasse, L. Rudolph, and D. Sleator. Competitive snoopy paging. *Algorithmica*, 3:70–119, 1988.
11. A. G. Konheim. A note on time sharing with preferred customers. *Probability Theory and Related Fields*, 9:112–130, 1968.
12. J. Labetoulle, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinooy Kan. Preemptive scheduling of uniform machines subject to release dates. In W. R. Pulleyblank, editor, *Progress in Combinatorial Optimization*, pages 245–261. Academic Press, New York, 1984.
13. J. K. Lenstra, A. H. G. Rinooy Kan, and P. Brucker. Complexity of machine scheduling problems. *Annals of Discrete Mathematics*, 1:243–362, 1977.
14. N. Megow and A. S. Schulz. On-line scheduling to minimize average completion time revisited. *Operations Research Letters*, 32:485–490, 2004.
15. N. Megow, M. Uetz, and T. Vredeveld. Models and algorithms for stochastic online scheduling. *Mathematics of Operations Research*, to appear, 2006.
16. N. Megow and T. Vredeveld. Approximation results for preemptive stochastic online scheduling. Technical Report 8/2006, Technische Universität Berlin, April 2006.
17. R. H. Möhring, F. J. Radermacher, and G. Weiss. Stochastic scheduling problems I: General strategies. *ZOR - Zeitschrift für Operations Research*, 28:193–260, 1984.
18. R. H. Möhring, A. S. Schulz, and M. Uetz. Approximation in stochastic scheduling: the power of LP-based priority policies. *Journal of the ACM*, 46:924–942, 1999.
19. M. Pinedo. Stochastic scheduling with release dates and due dates. *Operations Research*, 31:559–572, 1983.
20. M. Pinedo. Off-line deterministic scheduling, stochastic scheduling, and online deterministic scheduling: A comparative overview. In J. Leung, editor, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, chapter 38. CRC Press, 2004.
21. A. S. Schulz. New old algorithms for stochastic scheduling. In S. Albers, R. H. Möhring, G. Ch. Pflug, and R. Schultz, editors, *Algorithms for Optimization with Incomplete Information*, number 05031 in Dagstuhl Seminar Proceedings, 2005.
22. A. S. Schulz and M. Skutella. The power of α -points in preemptive single machine scheduling. *Journal of Scheduling*, 5:121–133, 2002.
23. A. S. Schulz and M. Skutella. Scheduling unrelated machines by randomized rounding. *SIAM Journal on Discrete Mathematics*, 15:450–469, 2002.
24. K.C. Sevcik. Scheduling for minimum total loss using service time distributions. *Journal of the ACM*, 21:65–75, 1974.
25. R. A. Sitters. *Complexity and Approximation in Routing and Scheduling*. PhD thesis, Technische Universiteit Eindhoven, 2004.
26. M. Skutella and M. Uetz. Stochastic machine scheduling with precedence constraints. *SIAM Journal on Computing*, 34:788–802, 2005.
27. D. Sleator and R. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28:202–208, 1985.
28. W. Smith. Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 3:59–66, 1956.
29. R. R. Weber. Scheduling jobs with stochastic processing requirements on parallel machines to minimize makespan or flow time. *Journal of Applied Probability*, 19:167–182, 1982.
30. G. Weiss. On almost optimal priority rules for preemptive scheduling of stochastic jobs on parallel machines. *Advances in Applied Probability*, 27:827–845, 1995.