Nicole Megow, Tjark Vredeveld

Approximation Results for Preemptive Stochastic Online Scheduling

RM/06/053

JEL code : C61

Maastricht research school of **E**conomics
of **TE**chnology and **OR**ganizations

Universiteit Maastricht
Faculty of Economics and Business Administration
P.O. Box 616
NL - 6200 MD Maastricht

phone : ++31 43 388 3830
fax : ++31 43 388 4873

Approximation Results for 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. E-mail: nmegow@math.tu-berlin.de

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

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

Our policies apply 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 policies extensively utilize information on processing time distributions other than the first (and second) moments. In order to derive our results 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 policies.

1 Introduction

Stochastic scheduling problems have attracted researchers for about four decades, see e.g. [22]. 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 [20, 29, 17, 24, 6] for nonpreemptive scheduling. We are not aware of any approximation results for preemptive problems. And previous approaches, based on linear programming relaxations, do not seem to carry over to the preemptive setting. In this paper, we give first approximation results for preemptive policies for 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 [17, 4]. This guarantee matches exactly the currently best known approximation result for the deterministic online version of this problem [16].

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

* 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.

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 $\Omega_j \subseteq \{1, 2, \ldots, M_j\}$ and that all release dates are integral. The sample space of all processing times is denoted by $\Omega = \Omega_1 \times \cdots \times \Omega_n$.

The objective is to schedule the processing of 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. [9], we denote the problem by $\mathrm{P} \,|\, r_j, pmtn \,|\, \mathbb{E}[\,\sum w_j C_j\,]$.

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 [18, 19]. 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 will restrict ourselves to so-called *online policies*, which learn about the existence and the characteristics of a job j only at its individual release date r_j. 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 the probability distribution of its processing time and the weight are 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 [17] for a more detailed discussion on stochastic online policies.

In this paper we concentrate on (online) approximation policies. As suggested in [17] we use a generalized definition of approximation guarantees from the stochastic scheduling setting [18].

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

$$\mathbb{E}[\,\Pi(\mathcal{I})\,] \;\leq\; \rho\,\mathbb{E}[\,\textsc{opt}(\mathcal{I})\,]\;,$$

where $\mathbb{E}[\,\Pi(\mathcal{I})\,]$ and $\mathbb{E}[\,\textsc{opt}(\mathcal{I})\,]$ 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 [13]. 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 [27] 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 [7] showed that this priority index is a special case of his Gittins index [7, 8]. Later in 1995, Weiss [33] 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, 13, 27]: 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, 32, 21, 5, 11, 10, 33]. For the parallel machine problem without release dates it is worthy to mention that Weiss [33] 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, $\mathrm{P} \,|\, r_j, pmtn \,|\, \mathbb{E}\,[\,\sum w_j C_j\,]$, 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 [14, 15]. Therefore, recently attempts have been made on obtaining approximation algorithms which have been successful in the nonpreemptive setting. Möhring, Schulz, and Uetz [20] derived first constant-factor approximations for the nonpreemptive problem with and without release dates. They were improved later by Megow et al. [17] and Schulz [24] for a more general setting. Skutella and Uetz [29] 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 [12, 30]. 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 [28] gives a 1.56-competitive algorithm for preemptive scheduling on a single machine. This is the currently best known and it improved upon an earlier result by Schulz and Skutella [25]; they generalized the classical *Smith rule* [31] 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 [16]. 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 [26] 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 [17] and randomized policies [17, 24] address nonpreemptive scheduling.

Our contribution. We derive first constant performance guarantees for preemptive stochastic scheduling. For jobs with general processing time distributions and individual release dates, we give 2-approximation policies 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 policies are not optimal. However, on restricted problem instances they coincide with policies whose optimality is known. If processing times are exponentially distributed and release dates are absent, our deterministic parallel machine policy coincide 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 $w_i \leq w_j$ [11]. If only one machine is available, we solve the simple weighted problem $1 \,|\, pmtn \,|\, \mathbb{E}[\sum w_j C_j]$ optimally by utilizing the Gittins index priority policy [13, 27, 33]. Moreover, Pinedo showed in [21] that in presence of release dates the WSEPT rule is optimal if all processing times are exponentially distributed. Furthermore, in the deterministic setting our single machine policy solves instances of the problem $1 \,|\, r_j, pmtn \,|\, \sum C_j$ optimally since it achieves the same schedule as Schrage's [23] *Shortest remaining processing time* (SRPT) rule. This performance is achieved in the weighted setting, as well, if all release dates are equal, in which case our policy coincides with Smith's rule [31].

Our results are 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 results is then the application of a Gittins index priority policy which is optimal to a relaxed version of our fast single machine relaxation.

The paper is organized as follows: Section 2 defines the gittins index priority rule and further preliminaries, whereas in Section 3 the new lower bound on the optimum for the preemptive stochastic scheduling problem is derived. In Section 4, a first parallel machine policy is introduced. It is followed by a more sophisticated policy for the single machine that uses more information of the current status of jobs being processed and by an immediate randomized extension to multiple machines. For the last two policies we cannot show an improved performance guarantee. However, there is well-founded hope that their approximation factor is less than proven, whereas the analysis of the deterministic policy in Section 4 is tight.

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 [13, 27, 33]. This result is crucial for the approximation results we give in this paper; it inspires the design of our policies 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, which ever comes first, as

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

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:

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

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 ([13, 27, 33]). *The Gittins index priority policy (*Gipp*) solves the preemptive stochastic scheduling problem* $1 \,|\, pmtn \,|\, \mathbb{E}\,[\,\sum w_j C_j\,]$ *optimally.*

The following properties of the Gittins indices and the lengths of sub-jobs achieving the Gittins index are well known, see [8, 33]. In parts, they have been derived earlier in the scheduling context by [13] and [27].

Proposition 1 ([8, 33]) *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 $i = 1, \ldots, n_j$. 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 preceed quantum (j, i) in the Gipp order. Let $\mathcal{Q}$ be the set of all quanta, i. e., $\mathcal{Q} = \{(k, l) \,|\, k = 1, \ldots, n,\ l = 1, \ldots, n_k\,\}$, then

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

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* $1 \,|\, pmtn \,|\, \mathbb{E}\left[\sum w_j C_j\right]$, GIPP, *achieves the expected objective value of*

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

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., $y_{ji_p} < p_j \leq y_{ji_p} + q_{ji_p}$. 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., $\mathbb{E}\left[\,\chi(\mathcal{E})\,\right] = \Pr\left[\mathcal{E}\right]$. 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

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

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

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

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 the optimum on parallel machines

For the scheduling problem $\mathrm{P}\,|\,r_j, pmtn\,|\,\mathbb{E}\left[\sum w_j C_j\right]$ 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 $\mathbb{E}\left[\,\textsc{opt}\,\right]$ of an unknown optimal policy $\textsc{opt}$. The trivial bound $\mathbb{E}\left[\,\textsc{opt}\,\right] \geq \sum_j w_j(\, r_j + \mathbb{E}\left[\, P_j \,\right])$ 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 [20, 29, 4, 17, 24] 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 $\mathcal{I}$ a scheduling instance of the problem type $\mathrm{P}\,|\,r_j, pmtn\,|\,\mathbb{E}\left[\sum w_j C_j\right]$, 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 $\textsc{opt}_1$ yields an expected value $\mathbb{E}\left[\,\textsc{opt}_1(\mathcal{I}')\,\right]$ on instance $\mathcal{I}'$. Then, for any parallel machine policy Π holds*

$$\mathbb{E}\left[\,\Pi(\mathcal{I})\,\right] \;\geq\; \mathbb{E}\left[\,\textsc{opt}_1(\mathcal{I}')\,\right]\,.$$

Proof. Given a parallel machine policy Π, we provide a policy Π' for the fast single machine that yields an expected objective value $\mathbb{E}\left[\,\Pi'(\mathcal{I}')\,\right] \leq \mathbb{E}\left[\,\Pi(\mathcal{I})\,\right]$ 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

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

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 $\Pr[P_j > x] = \Pr\left[P'_j > x/m\right]$, 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 $j \in I'$ after it has received $y' = y/m$ units of processing coincides with

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

We conclude that the partition of jobs into quanta in instance $\mathcal{I}$ immediately gives the partition for the fast single machine instance. 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

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

□

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.,*

$$\mathbb{E}\left[\,\text{OPT}(\mathcal{I})\,\right] \;\geq\; \frac{\mathbb{E}\left[\,\text{Gipp}(\mathcal{I}_0)\,\right]}{m}\,. \tag{8}$$

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 [16] which is a generalization of Smith's [31] optimal nonpreemptive single machine algorithm 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* $\text{P}\,|\,r_j, pmtn\,|\,\mathbb{E}\left[\,\sum w_j C_j\,\right]$.

Proof. This proof incorporates ideas from [16] 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:

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

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

In the case of exponentially distributed processing times and the absence of release dates, our policy F-GIPP coincides with the *Weighted shortest expected processing time* (WSEPT) rule. This rule 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 $\mathbb{E}\left[\, P_i \,\right] < \mathbb{E}\left[\, P_j \,\right]$ implies $w_i \leq w_j$ [11]. On the single machine our policy coincides with the WSEPT rule, if all processing times are drawn from exponential distributions, and is optimal [21]. In absence of release dates and for general processing times, our policy coincides with GIPP and is thus optimal (Theorem 1) even if jobs have individual weights.

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 [16].

5 A different policy

While the analysis of F-GIPP is tight, we present in this section another single machine policy with the same approximation guarantee. In contrast to the previous policy, we now deviate less from the original Gittins index priority rule and, thus, we use more information on the actual state of the set of known, unfinished jobs.

5.1 Single machine: generalized gittins index priority policy

We consider the online problem of preemptive scheduling on a single processor. As mentioned earlier, GIPP is not an optimal policy even in this single machine setting due to jobs arriving online over time. A straightforward extension of the policy GIPP is to choose at any time the job with highest rank among the set of available jobs. *Available* means that job j is released and has not been completed.

Algorithm 3: Generalized Gittins Index Priority Policy (Gen-Gipp)

At any time t, process an available job j with currently highest rank, $R_j(y_j(t))$, depending on the amount of processing $y_j(t)$ that the job j has completed by time t.

In principle, the jobs are still processed in non-increasing order of maximum ranks as in Gipp. Applied to an instance with equal release dates, both policies, Gipp and Gen-Gipp, yield the same schedule. The generalization in the policy Gen-Gipp concerns the fact that we incorporate release dates and cause preemptions of quanta whereas the Gipp policy preempts jobs only after completion of a quantum. Due to different arrival times in our current setting, Gen-Gipp preempts jobs within the processing of a quantum if a job with a higher rank is released. The interesting question concerns now the effect of those quantum preemptions on the job ordering.

From Proposition (1), we know that if a quantum (j,k) is preempted after $\gamma < q_{jk}$ units of processing, the rank of job j has not decreased, i.e., $R_j(y_{jk}+\gamma) \geq R_j(y_{jk})$. Hence, all quanta with a lower priority than the original priority of (j,k) available at or after the time that (j,k) is preempted will not be processed before quantum (j,k) is completed.

Consider a realization of processing times $p \in \Omega$ and a job j in the schedule obtained by Gen-Gipp. Let i_p be the index of the quantum in which job j finishes, i. e., $y_{ji_p} < p_j \leq y_{ji_p} + q_{ji_p}$. Then the completion time $C_j^{\text{Gen-Gipp}}$ of job j can be bounded by its release date plus the total length of the quanta that have a higher rank than (j,i_p) at time r_j. This includes quanta of jobs k with $r_k > r_j$ since they have rank $R_k(0)$ even though they are not available for scheduling.

This set of quanta contains not solely quanta in $H(j,i_p)$, i. e., quanta that have a higher priority than (j,i_p). The reason is that in the presence of release dates, the following situation is possible: a part of quantum $(k,l) \in L(j,i_p)$ is scheduled before quantum (j,i_p) which has higher rank even though job j is available. This happens when job k has been processed for $\gamma \in (y_{kl}, y_{k,l+1})$ units of time before time r_j and its rank improved (increased) such that $R_k(\gamma) > R_j(y_{ji_p})$. We call this event $\mathcal{E}_{k,ji_p}(\gamma)$ and we say that *job k* or *one of its quanta disturbs* (j,i_p). Formally we define,

$$\begin{aligned}\mathcal{E}_{k,ji}(\gamma) \;=\; \{\,&\text{by time } r_j,\ k \text{ has been processed for } \gamma \text{ units of time}\\ &\text{and } R_k(\gamma) > R_j(y_{ji})\,\}\,.\end{aligned}$$

The amount of time that the quantum (k,l) disturbs (j,i) is given by

$$q_{kl}^{ji}(\gamma) = \max\{q \leq y_{k,l+1} - \gamma \,:\, R_k(\gamma + q) > R_j(y_{ji})\}.$$

Note, that the event $\mathcal{E}_{k,ji}(\gamma)$ only depends on the (partial) realizations of jobs that have been processed before r_j and is thus independent of P_j. Furthermore, the amount of processing γ is integral since w.l.o.g. we restricted all release dates and processing times to integer values and therefore Gen-Gipp preempts jobs only at integral points in time.

Now, let us come back to the completion time of a job j in a realization $p \in \Omega$. As stated above, it can be bounded by r_j plus the total sum of quanta that have a higher rank at time r_j. These are

(i) all quanta in $H(j,i_p)$ except of those for which event $\mathcal{E}_{j,kl}(\gamma)$ occurs with $p_j \in (\gamma,\, \gamma + q_{ji}^{kl}(\gamma)\,]$ (i. e., quanta that are disturbed by (j,i_p) with j completing while it is disturbing), and

(ii) the quanta $(k,l) \in L(j,i_p)$ for which an event $\mathcal{E}_{k,ji_p}(\gamma)$ occurs for some $\gamma > y_{kl}$ (i. e., quanta that disturb (j,i_p)).

Formalized, that is

Proposition 1. *Given a realization $p \in \Omega$ and a job j, let i_p be the index of the quantum in which this job finishes, i. e., $y_{ji_p} < p_j \le y_{j,i_p+1}$. Then, the completion time of job j in the* Gen-Gipp *schedule can be bounded by*

$$\begin{aligned} C_j^{\text{Gen-Gipp}}(p) \;\le\;& r_j \\ &+ \sum_{(k,l)\in H(j,i_p)\,:\,p_k>y_{kl}} \min\{q_{kl}, p_k - y_{kl}\} \qquad (9)\\ &- \sum_{\substack{(k,l)\in H(j,i_p):\\ p_k>y_{kl}}} \sum_{\substack{\gamma\,:\,\mathcal{E}_{j,kl}(\gamma),\\ p_j\in(\gamma,\gamma+q_{ji}^{kl}(\gamma)]}} \min\{q_{kl}, p_k - y_{kl}\} \qquad (10)\\ &+ \sum_{\substack{(k,l)\in L(j,i_p):\\ p_k>y_{kl}}} \sum_{\substack{\gamma\,:\,\mathcal{E}_{k,ji_p}(\gamma),\\ p_k>\gamma>y_{kl}}} \min\{q_{kl}^{ji}(\gamma), p_k - \gamma\}\,. \qquad (11)\end{aligned}$$

Given the above bound for a particular realization, we compute the expected completion time of job j.

Lemma 4. *The expected completion time of job j under* Gen-Gipp *can be bounded by*

$$\begin{aligned} &\mathbb{E}\left[\, C_j^{\text{Gen-Gipp}} \,\right] \\ &\quad\le\; r_j + \sum_{i=1}^{n_j} \sum_{(k,l)\in H'(j,i)} \Pr\left[P_j > y_{ji} \,\wedge\, P_k > y_{kl}\right] \cdot I_k(q_{kl}, y_{kl}) \\ &\quad - \sum_{i=1}^{n_j} \sum_{\substack{(k,l)\\ \in H(j,i)}} \sum_{\gamma=y_{ji}}^{\infty} \Pr\left[\mathcal{E}_{j,kl}(\gamma) \wedge \gamma < P_j \le \gamma + q_{ji}^{kl}(\gamma)\right] \cdot \Pr\left[P_k > y_{kl}\right] \cdot I_k(q_{kl}, y_{kl}) \\ &\quad + \sum_{i=1}^{n_j} \sum_{(k,l)\in L(j,i)} \sum_{\gamma=y_{kl}}^{\infty} \Pr\left[y_{ji} < P_j \le y_{j,i+1}\right] \cdot \Pr\left[\mathcal{E}_{k,ji}(\gamma) \,\wedge\, P_k > \gamma\right] \cdot I_k(q_{kl}^{ji}(\gamma), \gamma)\,. \end{aligned}$$

Proof. The bound in Proposition 1 holds for each realization $p \in \Omega$. Taking the expectation over all realizations on both sides we get an upper bound on the expected completion time of a job j scheduled by Gen-Gipp. By linearity of expectation we can consider the sum of expected values of the summands (9), (10) and (11) separately.

As in Section 2, let $\chi(\mathcal{E})$ be an indicator random variable which equals 1 if and only if the event $\mathcal{E}$ occurs and denote by ξ_{kl} the special indicator random variable for the event $P_k > y_{kl}$. In the following paragraphs, we show how to transform the expected values of (9) to (11) such that their sum plus $\mathbb{E}\left[\, r_j \,\right]$ equals the claimed expression. The term (9) equals exactly the right hand side of equation (4) in the proof of Lemma 2. In that proof we showed that

$$\mathbb{E}\left[\,(9)\,\right] \;=\; \sum_{i=1}^{n_j} \sum_{(k,l)\in H'(j,i)} \Pr\left[P_j > y_{ji} \,\wedge\, P_k > y_{kl}\right] \cdot I_k(q_{kl}, y_{kl})\,.$$

Similarly, we transform the expected value of

$$\sum_{\substack{i<n_j:\\ y_{ji}<P_j\le y_{j,i+1}}} \sum_{\substack{(k,l)\in H(j,i):\\ P_k>y_{kl}}} \sum_{\substack{\gamma:\mathcal{E}_{j,kl}(\gamma),\\ P_j\in(\gamma,\gamma+q_{ji}^{kl}(\gamma)]}} \min\{q_{kl}, P_k-y_{kl}\}$$

from (10) to

$$\begin{aligned}
&\mathbb{E}\left[\sum_{i=1}^{n_j}\chi(y_{ji}<P_j\le y_{j,i+1}) \sum_{\substack{(k,l)\in H(j,i):\\ P_k>y_{kl}}} \sum_{\substack{\gamma:\mathcal{E}_{j,kl}(\gamma),\\ P_j\in(\gamma,\gamma+q_{ji}^{kl}(\gamma)]}} \min\{q_{kl}, P_k-y_{kl}\}\right]\\
&=\mathbb{E}\left[\sum_{i=1}^{n_j}\sum_{\substack{(k,l)\\ \in H(j,i)}}\sum_{\gamma=y_{ji}}^{\infty}\chi\left(\gamma<P_j\le\gamma+q_{ji}^{kl}(\gamma)\wedge P_k>y_{kl}\wedge\mathcal{E}_{j,kl}(\gamma)\right)\min\{q_{kl},P_k-y_{kl}\}\right]\\
&=\sum_{i=1}^{n_j}\sum_{\substack{(k,l)\\ \in H(j,i)}}\sum_{\gamma=y_{ji}}^{\infty}\Pr\left[\mathcal{E}_{j,kl}(\gamma)\wedge\gamma<P_j\le\gamma+q_{ji}^{kl}(\gamma)\right]\cdot\Pr\left[P_k>y_{kl}\right]\cdot I_k(q_{kl},y_{kl})\,.
\end{aligned}$$

Finally, the expected value of Term (11) can be reformulated in an equivalent way and therefore we omit the details for showing

$$\mathbb{E}\,[\,(11)\,]=\sum_{i=1}^{n_j}\sum_{\substack{(k,l)\\ \in L(j,i)}}\sum_{\gamma=y_{kl}}^{\infty}\Pr\left[y_{ji}<P_j\le y_{j,i+1}\right]\cdot\Pr\left[\mathcal{E}_{k,ji}(\gamma)\wedge P_k>\gamma\right]\cdot I_k(q_{kl}^{ji}(\gamma),\gamma)\,.$$

This concludes the proof. □

Note, that in the absence of release dates the events $\mathcal{E}_{j,kl}(\gamma)$ do not occur for any job $j\in J$ and any $\gamma>0$. Now, we can give the approximation guarantee.

Theorem 4. Gen-Gipp *is a* 2*-approximative policy for the problem* $1\,|\,r_j, pmtn\,|\,\mathbb{E}\left[\sum w_jC_j\right]$.

Proof. By Lemma 4 and Corollary 2 we can bound the expected objective value, $\mathbb{E}\,[\,\text{Gen-Gipp}\,]$, of a schedule that has been obtained by Gen-Gipp as follows

$$\begin{aligned}
\mathbb{E}\,[\,\text{Gen-Gipp}\,] &= \sum_j w_j\mathbb{E}\left[\,C_j^{\text{Gen-Gipp}}\,\right]\\
&\le \sum_j w_jr_j+\mathbb{E}\,[\,\text{Gipp}\,]+\sum_j w_j\,(O_j-N_j),
\end{aligned}$$

where

$$\begin{aligned}
O_j&=\sum_{i=1}^{n_j}\sum_{\substack{(k,l)\\ \in L(j,i)}}\sum_{\gamma=y_{kl}}^{\infty}\Pr\left[y_{ji}<P_j\le y_{j,i+1}\right]\cdot\Pr\left[\mathcal{E}_{k,ji}(\gamma)\wedge P_k>\gamma\right]\cdot I_k(q_{kl}^{ji}(\gamma),\gamma)\\
N_j&=\sum_{i=1}^{n_j}\sum_{\substack{(k,l)\\ \in H(j,i)}}\sum_{\gamma=y_{ji}}^{\infty}\Pr\left[\mathcal{E}_{j,kl}(\gamma)\wedge\gamma<P_j\le\gamma+q_{ji}^{kl}(\gamma)\right]\cdot\Pr\left[P_k>y_{kl}\right]\cdot I_k(q_{kl},y_{kl})\,.
\end{aligned}$$

We claim that $\sum_j w_j\,(O_j - N_j) \leq 0$ and give the proof in Lemma 5 below. This implies the theorem due to the trivial lower bound on the expected value of an optimal policy OPT, $\mathbb{E}\,[\,\text{OPT}\,] \geq \sum_j w_j\, r_j$, and the fact that GIPP is an optimal policy for the relaxed problem without release dates (Theorem 1), which gives $\mathbb{E}\,[\,\text{OPT}\,] \geq \mathbb{E}\,[\,\text{GIPP}\,]$. □

Lemma 5. *Let*

$$O_j = \sum_{i=1}^{n_j} \sum_{\substack{(k,l)\\ \in L(j,i)}} \sum_{\gamma=y_{kl}}^{\infty} \Pr\left[y_{ji} < P_j \leq y_{j,i+1}\right] \cdot \Pr\left[\mathcal{E}_{k,ji}(\gamma) \wedge P_k > \gamma\right] \cdot I_k(q_{kl}^{ji}(\gamma), \gamma)$$

and

$$N_j = \sum_{i=1}^{n_j} \sum_{\substack{(k,l)\\ \in H(j,i)}} \sum_{\gamma=y_{ji}}^{\infty} \Pr\left[\mathcal{E}_{j,kl}(\gamma) \wedge \gamma < P_j \leq \gamma + q_{ji}^{kl}(\gamma)\right] \cdot \Pr\left[P_k > y_{kl}\right] \cdot I_k(q_{kl}, y_{kl})\,,$$

then

$$\sum_j w_j\;(O_j - N_j) \;\leq\; 0\,.$$

Proof. In order to prove the claim, we first note that $(j,i) \in H(k,l)$ for jobs $k \neq j$ implies $(k,l) \in L(j,i)$ and vice versa. Moreover, the event $\mathcal{E}_{j,ji}(\gamma)$ is empty for all i and γ. Thus, we can transform $\sum_k w_k\; N_k$ by rearranging indices:

$$\begin{aligned}
&\sum_k w_k\; N_k \\
&\quad = \sum_k \sum_{l=1}^{n_k} \sum_{\substack{(j,i)\\ \in H(k,l)}} \sum_{\gamma=y_{kl}}^{\infty} w_k \Pr\left[\mathcal{E}_{k,ji}(\gamma) \wedge \gamma < P_k \leq y_{k,l+1}\right] \cdot \Pr\left[P_j > y_{ji}\right] \cdot I_j(q_{ji}, y_{ji}) \\
&\quad = \sum_j \sum_{i=1}^{n_j} \sum_{\substack{(j,i)\\ \in L(k,l)}} \sum_{\gamma=y_{kl}}^{\infty} w_k \Pr\left[\mathcal{E}_{k,ji}(\gamma) \wedge \gamma < P_k \leq y_{k,l+1}\right] \cdot \Pr\left[P_j > y_{ji}\right] \cdot I_j(q_{ji}, y_{ji})\,.
\end{aligned}$$

Secondly, note that by definition of the conditional probability holds

$$\Pr\left[y_{ji} < P_j \leq y_{j,i+1}\right] = \Pr\left[P_j > y_{ji}\right] \cdot \Pr\left[P_j \leq y_{j,i+1} \,|\, P_j > y_{ji}\right]\,,$$

for any quantum (j,i). Moreover, due to the independence of the processing times, we know that $\Pr\left[\mathcal{E}_{k,ji}(\gamma) \wedge \gamma < P_k \leq y\right] = \Pr\left[\mathcal{E}_{k,ji}(\gamma) \wedge P_k > \gamma\right] \cdot \Pr\left[P_k \leq y \,|\, P_k > \gamma\right]$ for any y. With these arguments we have

$$\begin{aligned}
&\sum_j w_j\,(O_j - N_j)\\
&\quad= \sum_j \sum_{i=1}^{n_j} \sum_{\substack{(k,l)\\ \in L(j,i)}} \sum_{\gamma=y_{kl}}^{\infty} \Big(w_j \Pr\left[y_{ji} < P_j \le y_{j,i+1}\right] \Pr\left[\mathcal{E}_{j,kl}(\gamma) \wedge P_k > \gamma\right] \cdot I_k(q_{kl}^{ji}(\gamma), \gamma)\\
&\qquad\qquad - w_k \Pr\Big[\mathcal{E}_{k,ji}(\gamma) \wedge \gamma < P_k \le \gamma + q_{kl}^{ji}(\gamma)\Big] \cdot \Pr\left[P_j > y_{ji}\right] \cdot I_j(q_{ji}, y_{ji}) \Big)\\
&\quad= \sum_j \sum_{i=1}^{n_j} \sum_{\substack{(k,l)\\ \in L(j,i)}} \sum_{\gamma=y_{kl}}^{\infty} \Pr\left[\mathcal{E}_{j,kl}(\gamma) \wedge P_k > \gamma\right] \cdot \Pr\left[P_j > y_{ji}\right] \cdot\\
&\qquad\qquad \Big(w_j \Pr\left[P_j \le y_{j,i+1} \,|\, P_j > y_{ji}\right] \cdot I_k(q_{kl}^{ji}(\gamma), \gamma)\\
&\qquad\qquad\qquad - w_k \Pr\Big[P_k \le \gamma + q_{kl}^{ji}(\gamma) \,|\, P_k > \gamma\Big] \cdot I_j(q_{ji}, y_{ji}) \Big)\\
&\quad\le 0\,,
\end{aligned}$$

because when event $\mathcal{E}_{k,ji}(\gamma)$ occurs, we know that $R_k(q_{kl}^{ji}(\gamma), \gamma) \ge R_j(q_{ji}, y_{ji})$ and thus,

$$\begin{aligned}
\frac{w_k \Pr\Big[P_k \le \gamma + q_{kl}^{ji}(\gamma) \,|\, P_k > \gamma\Big]}{I_k(q_{kl}^{ji}(\gamma), \gamma)} &= R_k(q_{kl}^{ji}(\gamma), \gamma) \ge R_j(q_{ji}, y_{ji})\\
&= \frac{w_j \Pr\left[P_j \le y_{j,i+1} \,|\, P_j > y_{ji}\right]}{I_j(q_{ji}, y_{ji})}\,.
\end{aligned}$$

□

We conjecture that the true approximation ratio of Gen-Gipp is less than 2. We give a *bad instance* for which Gipp cannot achieve a performance ratio less than 1.21. This example is a deterministic online instance. Note that in this case, Gen-Gipp schedules at any time the job with highest ratio of weight over remaining processing time.

Example 1. The instance consists of $k+2$ jobs: a high priority job h with unit weight and processing requirement, a low priority job l of length p_l and unit weight and k small jobs of length ϵ. The job l and the first small job are released at time 0 followed by the remaining small jobs at times $r_j = (j-1)\epsilon$ for $j = 2, \ldots, k$ whereas the high priority job is released at time $r_h = p_l - 1$. The weights of the small jobs are $w_j = \epsilon/(p_l - (j-1)\epsilon)$ for $j = 1, \ldots, k$. We choose ϵ such that all small jobs could be processed until r_h, i. e., $\epsilon = r_h/k = (p_l - 1)/k$.

W.l.o.g. we can assume that Gen-Gipp starts processing job l at time 0. Note that the weights of the small jobs are chosen such that the ratio of weight over remaining processing time of job l at the release date of a small job equals the ratio of the newly released small job, and thus Gen-Gipp does not preempt l until at time $r_h = p_l - 1$ when job h is released and starts processing. After its completion, job j is finished, followed by the small jobs $l, l-1, \ldots, 1$. The value of the achieved schedule is

$$2p_l + 1 + \sum_{i=1}^{k} (p_l + 1 + i\epsilon) \frac{\epsilon}{p_l - (k-i)\epsilon}\,.$$

An optimal policy, instead, processes first all small jobs followed by the high priority job and finishes with the job l. The value of such a schedule is

$$\sum_{i=1}^{k} i\epsilon \frac{\epsilon}{p_l - (i-1)\epsilon} + 3p_l \,.$$

If the number of small jobs, k, tends to infinity then the performance ratio of Gen-Gipp is no less than

$$\frac{p_l(3 - \log\frac{1}{p_l-1} + \log\frac{p_l}{p_l-1})}{1 + 2p_l + p_l \log p_l}$$

which gives a lower bound of 1.21057 for $p_l \approx 5.75$.

However, Gen-Gipp solves deterministic instance with equal weights optimally, since in that case it coincides with Schrage's [23] *Shortest remaining processing time* (SRPT) rule which is known to find the optimal schedule.

5.2 A randomized extension to parallel machines

In this section we derive a randomized policy for multiple machines that utilizes the single machine policy Gen-Gipp in a straightforward way. It yields again an approximation guarantee of 2.

Algorithm 4: Randomized Gittins Index Priority Policy (Rand-Gipp)

Assign a job at its release date to any of the m machines by choosing one with probability $1/m$. On each of the machines run the Gen-Gipp policy, i. e., at any point in time schedule on each machine m_i the quantum with currently highest rank among the available, not yet completed jobs that have been assigned to m_i.

Theorem 5. *The online policy* Rand-Gipp *is a* 2*-approximation for the preemptive scheduling problem on parallel machines* $\mathrm{P} \,|\, r_j, pmtn \,|\, \mathbb{E}\,[\,\sum w_j C_j\,]$.

Proof. The policy uses the single machine policy Gen-Gipp and parts of the performance analysis from the previous section can also be recycled. Therefore, we avoid repeating rather complex terms and ask the reader to follow the references.

Consider a realization $p \in \Omega$ of processing times and a job j. Denote by $j \to m_x$ that job j is assigned to machine m_x in the considered realization. Since on each machine m_x the single machine policy Gen-Gipp runs, the completion time of job j, given that it is processing on machine m_x, is given in Corollary 1 with a minor modification for our current setting, i. e., we sum up only over jobs that are assigned to the same machine m_x as job j. We denote the corresponding value by $(9)' + (10)' + (11)'$. Thus, the expected completion time of j over all realizations is

$$\begin{aligned}
&\mathbb{E}\left[\, C_j^{\textsc{Rand-Gipp}} \,\middle|\, j \to m_x \right] \\
&\quad\le\ r_j + \ \mathbb{E}\left[\ (9)' \ +\ (10)' \ +\ (11)' \,|\, j \to m_x \,\right] \\
&\quad\le\ r_j + \ \sum_k \Pr\left[k \to m_x \,|\, j \to m_x\right] \cdot \mathbb{E}\left[\ (9) \ +\ (10) \ +\ (11) \,|\, j \to m_x \,\right] \\
&\quad=\ r_j + \ \sum_k \Pr\left[k \to m_x \,|\, j \to m_x\right] \cdot \mathbb{E}\left[\ (9) \ +\ (10) \ +\ (11)\ \right] .
\end{aligned}$$

Unconditioning the expected completion time from the fixed machine assignment and using the fact that all jobs are assigned to m_x with the same probability $1/m$, independently of each other, yield

$$\begin{aligned}
&\mathbb{E}\left[\, C_j^{\textsc{Rand-Gipp}} \,\right] \\
&= \sum_{x=1}^{m} \Pr\left[j \to m_x\right] \cdot \mathbb{E}\left[\, C_j^{\textsc{Rand-Gipp}} \,\middle|\, j \to m_x \right] \\
&\le \sum_{x=1}^{m} \Pr\left[j \to m_x\right] \big(r_j + \sum_k \Pr\left[k \to m_x \,|\, j \to m_x\right] \mathbb{E}\left[\,(9) + (10) + (11)\,\right] \big) \\
&\le\ r_j + \frac{1}{m}\ \mathbb{E}\left[\ (9) \ +\ (10) \ +\ (11)\ \right] .
\end{aligned}$$

The total expected value of the schedule is then

$$\begin{aligned}
\mathbb{E}\left[\,\textsc{Rand-Gipp}\,\right] \ &=\ \sum_j w_j \mathbb{E}\left[\, C_j^{\textsc{Rand-Gipp}} \,\right] \\
&\le\ \sum_j w_j\, r_j \ + \frac{1}{m}\ \sum_j w_j\, \mathbb{E}\left[\ (9) \ +\ (10) \ +\ (11)\ \right] \\
&\le\ \sum_j w_j\, r_j \ + \frac{1}{m}\ \sum_j w_j\, \mathbb{E}\left[\, C_j^{\textsc{Gipp}} \,\right] \\
&\le\ 2 \cdot \mathbb{E}\left[\,\textsc{opt}\,\right] .
\end{aligned}$$

The second inequality follows from Lemma 4 and Theorem 4. Finally, the third inequality is derived from the trivial lower bound on the optimum, $\mathbb{E}\left[\,\textsc{opt}\,\right] \ge \sum_j w_j\, r_j$ and from the bound in Corollary 1. □

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. B. C. Dean. *Approximation Algorithms for Stochastic Scheduling Problems.* PhD thesis, Massachusetts Institute of Technology, 2005.
7. J. C. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society, Series B*, 41:148–177, 1979.
8. J. C. Gittins. *Multi-armed Bandit Allocation Indices.* Wiley, New York, 1989.
9. 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.
10. C. C. Huang and G. Weiss. Preemptive scheduling of stochastic jobs with a two stage processing time distribution on $m + 1$ parallel machines. *Probability in the Engineering and Informational Sciences*, 6:171–191, 1992.
11. T. Kämpke. Optimal scheduling of jobs with exponential service times on identical parallel processors. *Operations Research*, 37(1):126–133, 1989.
12. A. Karlin, M. Manasse, L. Rudolph, and D. Sleator. Competitive snoopy paging. *Algorithmica*, 3:70–119, 1988.
13. A. G. Konheim. A note on time sharing with preferred customers. *Probability Theory and Related Fields*, 9:112–130, 1968.
14. 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.
15. J. K. Lenstra, A. H. G. Rinooy Kan, and P. Brucker. Complexity of machine scheduling problems. *Annals of Discrete Mathematics*, 1:243–362, 1977.
16. N. Megow and A. S. Schulz. On-line scheduling to minimize average completion time revisited. *Operations Research Letters*, 32:485–490, 2004.
17. N. Megow, M. Uetz, and T. Vredeveld. Models and algorithms for stochastic online scheduling. *Mathematics of Operations Research, to appear*, 2006.
18. R. H. Möhring, F. J. Radermacher, and G. Weiss. Stochastic scheduling problems I: General strategies. *ZOR - Zeitschrift für Operations Research*, 28:193–260, 1984.
19. R. H. Möhring, F. J. Radermacher, and G. Weiss. Stochastic scheduling problems II: Set strategies. *ZOR - Zeitschrift für Operations Research*, 29(3):A65–A104, 1985.
20. 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.
21. M. Pinedo. Stochastic scheduling with release dates and due dates. *Operations Research*, 31:559–572, 1983.
22. 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.
23. L. Schrage. A proof of the optimality of the shortest remaining processing time discipline. *Operations Research*, 16:687–690, 1968.
24. 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. Internationales Begegnungs- und Forschungszentrum (IBFI), Schloss Dagstuhl, Germany, 2005.
25. A. S. Schulz and M. Skutella. The power of α-points in preemptive single machine scheduling. *Journal of Scheduling*, 5:121–133, 2002.

26. A. S. Schulz and M. Skutella. Scheduling unrelated machines by randomized rounding. *SIAM Journal on Discrete Mathematics*, 15:450–469, 2002.
27. K.C. Sevcik. Scheduling for minimum total loss using service time distributions. *Journal of the ACM*, 21:65–75, 1974.
28. R. A. Sitters. *Complexity and Approximation in Routing and Scheduling.* PhD thesis, Technische Universiteit Eindhoven, 2004.
29. M. Skutella and M. Uetz. Stochastic machine scheduling with precedence constraints. *SIAM Journal on Computing*, 34:788–802, 2005.
30. D. Sleator and R. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28:202–208, 1985.
31. W. Smith. Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 3:59–66, 1956.
32. 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.
33. G. Weiss. On almost optimal priority rules for preemptive scheduling of stochastic jobs on parallel machines. *Advances in Applied Probability*, 27:827–845, 1995.