Stochastic Scheduling on Unrelated Machines *

Martin Skutella[1], Maxim Sviridenko[2], and Marc Uetz[3]

1 **TU Berlin, Institut für Mathematik, Berlin, Germany**
`martin.skutella@tu-berlin.de`
2 **University of Warwick, Department of Computer Science, Coventry, United Kingdom**
`sviri@dcs.warwick.ac.uk`
3 **University of Twente, Department of Applied Mathematics, Enschede, The Netherlands**
`m.uetz@utwente.nl`

Abstract

Two important characteristics encountered in many real-world scheduling problems are heterogeneous processors and a certain degree of uncertainty about the sizes of jobs. In this paper we address both, and study for the first time a scheduling problem that combines the classical unrelated machine scheduling model with stochastic processing times of jobs. Here, the processing time of job j on machine i is governed by random variable P_{ij}, and its realization becomes known only upon job completion. With w_j being the given weight of job j, we study the objective to minimize the expected total weighted completion time $\mathbb{E}\left[\sum_j w_j C_j\right]$, where C_j is the completion time of job j. By means of a novel time-indexed linear programming relaxation, we compute in polynomial time a scheduling policy with performance guarantee $(3+\Delta)/2+\varepsilon$. Here, $\varepsilon > 0$ is arbitrarily small, and Δ is an upper bound on the squared coefficient of variation of the processing times. When jobs also have individual release dates r_{ij}, our bound is $(2+\Delta)+\varepsilon$. We also show that the dependence of the performance guarantees on Δ is tight. Via $\Delta = 0$, currently best known bounds for deterministic scheduling on unrelated machines are contained as special case.

1998 ACM Subject Classification F.2.2 Sequencing and Scheduling, G.2.1 Combinatorial Algorithms, G.1.6 Optimization

Keywords and phrases Stochastic Scheduling, Unrelated Machines, Approximation Algorithm

Digital Object Identifier 10.4230/LIPIcs.STACS.2014.639

1 Introduction

Deterministic scheduling. The problem to minimize the total weighted completion time on unrelated parallel machines, denoted $\mathrm{R} \mid (r_{ij}) \mid \sum w_j C_j$ in the three-field notation of Graham et al. [8], is one of the most important classical problems in the theory of deterministic scheduling. Each job j has a weight w_j, possibly an individual release date r_{ij} before which job j must not be scheduled on machine i, and the processing time of job j on machine i is p_{ij}. Each job has to be processed non preemptively on any one of the machines, and each machine can process at most one job at a time. The objective is to find a schedule minimizing

* The first author was supported by the DFG Research Center MATHEON "Mathematics for key technologies" in Berlin and by the DFG Focus Program 1307 within the project "Algorithm Engineering for Real-time Scheduling and Routing". The second author was supported by EPSRC grants EP/J021814/1, EP/D063191/1, FP7 Marie Curie Career Integration Grant and Royal Society Wolfson Research Merit Award. The third author was supported by the Centre for Telematics and Information Technology (CTIT) of the University of Twente.

the total weighted completion time $\sum_j w_j C_j$, where C_j denotes the completion time of job j in the schedule. The special case with identical parallel machines is already known to be strongly NP-hard [12] but there do exist polynomial time approximation schemes [1, 28]. The general setting of unrelated parallel machines turns out to be significantly harder and there is a complexity gap compared to identical parallel machines: Hoogeveen et al. [11] prove MaxSNP-hardness and hence there is no polynomial time approximation scheme. On the positive side, the currently best known approximation algorithms for unrelated parallel machines have performance guarantees 3/2 and 2, for the problem without and with release dates, respectively [4, 23, 25, 26]. Improving these bounds is considered to be among the most important open problems in scheduling [24] which is also an indication of the high significance of unrelated machine scheduling.

Stochastic scheduling. We consider for the first time the stochastic variant of unrelated machine scheduling. Here, the processing time of a job j on machine i is given by random variable P_{ij}. In stochastic scheduling, we are asked to compute a non-anticipatory scheduling policy. Roughly spoken, a scheduling policy makes scheduling decisions at certain decision times t, and these decisions are based 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 of the processing times of jobs which have not yet been completed by time t. We refer to Möhring et al. [16] for the formal definition of stochastic scheduling policies, and here confine ourselves with an intuitive description that puts stochastic scheduling in the framework of stochastic dynamic optimization: Actions of a scheduling policy at a time t consists of a set of jobs, possibly empty, to be started on a set of idle machines, together with a tentative next decision time $t^* > t$. The next action of the policy is due at t^*, or the time of the next job completion, or the time when the next job is released, whatever occurs first. Depending on the action of the policy, the next decision time as well as the state of the schedule at the next decision time is realized according to the probability distributions of the jobs' processing times. A non-anticipatory policy may learn over time, but it has only access to distributional information about remaining processing times of unfinished jobs, conditioned on the state of the schedule at time t.[1] As all previous work in the area, we assume that the random variables P_{ij} are stochastically independent across jobs. For any given non-anticipatory scheduling policy, the possible outcome of the objective function $\sum_j w_j C_j$ is a random variable, and our goal is to minimize its expected value, which by linearity of expectation equals $\sum_j w_j \mathbb{E}[C_j]$.

Related work. Generalizing a well known result of Smith [29] for deterministic single machine scheduling, Rothkopf [19] proved in 1966 that the WSEPT rule[2] minimizes the expected total weighted completion time on a single machine. Apart from Weiss' results on the asymptotic optimality of WSEPT in stochastic scheduling on identical parallel machines [32, 33], the first constant factor approximation algorithms for stochastic scheduling on identical parallel machines have been obtained in 1999 by Möhring et al. [17]. Next to a linear programming (LP) based analysis of the WSEPT rule, they define list scheduling

[1] A concrete example may help: Imagine a job j which has processing time either small (ε) or large (M), both with probability 1/2. For a scheduling policy that starts this job at time t, it can make sense to define a tentative next decision time at $t^* = t + \varepsilon$, because then it learns with certainty what the actual processing time of job j is. Using such building blocks, one can even show that an optimal scheduling policy is generally not work conserving, i. e., machines are left deliberately idle [31].

[2] Weighted shortest expected processing time first: schedule jobs in order of non-increasing ratios $w_j/\mathbb{E}[P_j]$.

Table 1 Performance bounds for nonpreemptive stochastic machine scheduling problems. Parameter $\varepsilon > 0$ can be chosen arbitrarily small. Parameter Δ upper bounds the squared coefficient of variation $\mathbb{CV}^2[P_{ij}] = \mathbb{V}\text{ar}[P_{ij}]/\mathbb{E}^2[P_{ij}]$ for all P_{ij}. The third column shows the results for $\mathbb{CV}[P_{ij}] \leq 1$; e. g., uniform, exponential, or Erlang distributions. As usual in stochastic scheduling, these bounds hold with respect to the expected performance of any non-anticipatory scheduling policy.

stochastic scheduling model	worst case performance guarantee arbitrary P_{ij}	$\mathbb{CV}[P_{ij}] \leq 1$	reference
$\mathrm{P} \,\|\, \|\, \mathbb{E}\big[\sum w_j C_j\big]$	$1 + \frac{(m-1)(\Delta+1)}{2m}$	$2 - 1/m$	[17]
$\mathrm{P} \,\|\, r_j \,\|\, \mathbb{E}\big[\sum w_j C_j\big]$	$2 + \Delta$	3	[22]
$\mathrm{R} \,\|\, \|\, \mathbb{E}\big[\sum w_j C_j\big]$	$1 + \frac{\Delta+1}{2} + \varepsilon$	$2 + \varepsilon$	this paper
$\mathrm{R} \,\|\, r_{ij} \,\|\, \mathbb{E}\big[\sum w_j C_j\big]$	$2 + \Delta + \varepsilon$	$3 + \varepsilon$	this paper

policies which are based on linear programming relaxations in completion time variables. The performance bounds are constant whenever the coefficients of variation of the jobs' processing times are bounded by a constant. As usual in stochastic scheduling, all bounds hold with respect to any non-anticipatory scheduling policy. By using an idea from Chekuri et al. [2], that approach was extended to stochastic scheduling problems with precedence constrains by Skutella and Uetz [27]. Subsequently, in line with earlier work by Chou et al. [3], Megow et al. [14] combined the stochastic scheduling model with online scheduling, and derived combinatorial, constant competitive algorithms that are not guided by linear programming relaxations. Yet all results, including the analysis in [14], are based on one and the same linear programming relaxation, namely that of [17]. With respect to the underlying relaxation, Schulz [22] goes one step further, and uses the mean busy time relaxation that was previously used also by Correa and Wagner [5], yet its validity in stochastic scheduling still relies on the validity of the completion time relaxation of [17]. Nevertheless, in comparison to [14], the clever use of an optimal solution to an equivalent time-indexed LP relaxation for deterministic scheduling yields improved and simpler results.

Two other research directions are related to our work, yet for different models and independent of the techniques of [17] as well as ours. One is approximation algorithms for preemptive stochastic scheduling by Megow and Vredeveld [15]. They use a single machine relaxation that is optimally solved by a Gittins index policy, and thereby achieve a competitive ratio of 2 for preemptive online stochastic scheduling on parallel identical machines. The other is work by Scharbrodt et al. [20] and Souza and Steger [30], who analyze the expected competitive ratio rather than the expected performance of a policy. In that model, one analyzes the ratio $\mathbb{E}[v(\Pi)/v(\text{Offline-Opt})]$, while we follow [14, 17, 22, 27] and focus on the ratio $\mathbb{E}[v(\Pi)]/\mathbb{E}[v(\Pi^{\text{Opt}})]$ instead.

Note that all results discussed here are restricted to identical parallel machines. Table 1 gives an overview of currently best known performance bounds in nonpreemptive stochastic scheduling with minsum objective, next to the results obtained in this paper.

With respect to algorithmic ideas and techniques, the evolution of stochastic scheduling has largely benefited in the past from progress being made for the corresponding deterministic scheduling problems. For example, all LP-based approximation results for stochastic scheduling on identical parallel machines outlined above build upon a class of linear programming relaxations in completion time variables that dates back to Wolsey [34] and Queyranne [18] (for single machine scheduling), and was later generalized to identical parallel machines by

Schulz [21] and Hall et al. [10] who also presented LP-based approximation algorithms for deterministic scheduling problems.

Our contribution. We obtain the first approximation algorithms for stochastic scheduling on unrelated machines. Despite the fact that the unrelated machine scheduling model is significantly richer than identical machine scheduling, our bounds essentially match all previous performance bounds that have been obtained for the corresponding stochastic scheduling problems on identical parallel machines; see Table 1. We also give a tight lower bound, showing that the dependence of the performance bound on the squared coefficient of variation Δ is unavoidable for the class of policies that we use. For the first time we completely depart from the LP relaxation of Möhring et al. [17], and show how to put a novel, time-indexed linear programming relaxation to work in stochastic machine scheduling. We are optimistic that this novel approach will inspire further research and prove useful for other stochastic optimization problems in scheduling and related areas.

Time-indexed linear programming relaxations have played a pivotal role in the development of constant factor approximation algorithms for deterministic scheduling on unrelated parallel machines [23]. In spite of that, it remained unclear and a major open problem how to come up with a meaningful time-indexed LP relaxation for stochastic scheduling problems [13]. Here the main difficulty is that, in contrast to deterministic schedules that can be fully described by time-indexed 0-1-variables, scheduling *policies* feature a considerably richer structure including complex dependencies between the execution of different jobs which cannot be easily described by time-indexed variables.

In Section 3 we show how to overcome this difficulty and present the first time-indexed LP relaxation for stochastic scheduling on unrelated parallel machines. Here, the value of the time-indexed variable x_{ijt} represents the probability of job j being started on machine i at time t.[3] While writing down the machine capacity constraints[4] is rather easy for deterministic scheduling in this formulation, the situation is somewhat more complicated in the stochastic setting and we require a fair amount of information about the exact probability distributions of random variables P_{ij}.

Notice that, due to the stochastic nature of processing times, even a schedule produced by an optimal policy can be arbitrarily long such that infinitely many variables x_{ijt} may take positive values. Nonetheless, in the full version of the paper we show how to overcome this difficulty. Indeed, we can compute an LP-solution in polynomial time that approximates the optimal LP solution with arbitrary precision.

In Section 4 we discuss how to turn a feasible solution to the time-indexed LP relaxation into a simple scheduling policy. Our approach is inspired by the randomized rounding algorithm for deterministic scheduling on unrelated parallel machines in [23]. Each job j is randomly assigned to a machine i with probability $\sum_t x_{ijt}$; then, on each machine i, the WSEPT policy is used to schedule the jobs assigned to i. The analysis, however, is based on a somewhat more elaborate, random sequencing of jobs which is determined by a two-stage random process.

Since each job is immediately and irrevocably assigned to a machine, our scheduling policies fall into the special class of *fixed assignment policies*. Notice that these policies ignore the additional information that evolves over time in the form of the actual realizations of processing times. Not surprisingly, this ignorance comes at a price. In Section 6 we

[3] Even for simple scheduling policies like the WSEPT rule, determining this probability is non-trivial.
[4] The machine capacity constraints say that each machine can process at most one job at a time.

prove a lower bound of $\Delta/2$ on the performance guarantee of *any* fixed assignment policy. Moreover, we also show that the LP relaxation can have an optimality gap in the same order of magnitude. These negative results nicely complement our positive results; see Table 1.

In order to keep the presentation as simple as possible, we ignore release dates and restrict to the problem $R||\mathbb{E}[\sum w_j C_j]$ throughout most of the paper. Only in Section 5 we show how release dates can be taken care of in our approach.

Parallel to Stochastic Knapsack. There is an interesting parallel of the present work on stochastic scheduling with that on stochastic knapsack problems[5]. The first study of approximation algorithms for stochastic knapsack problems is due to Dean et al. [6], presenting constant factor approximation algorithms along with an analysis of the adaptivity gap[6]. Their results are based on a linear programming relaxation that is essentially the deterministic knapsack LP where item sizes and weights are replaced by expected values. In that sense, methodology-wise their linear program parallels that of [17] in stochastic scheduling on parallel machines. Recently, Gupta et al. [9] were able to obtain constant factor approximation algorithms for a much broader class of stochastic knapsack problems (and other problems, too). Key to these results is a more sophisticated, time-indexed linear programming relaxation, based on the same type of variables as we use here. It is interesting to note that in their paper as well as in ours, moving from "natural yet simple" LP relaxations to richer time-indexed LP relaxations is key to more general results.

2 Notation and preliminaries

We are given a set of jobs J of cardinality n with job weights $w_j \in \mathbb{Z}_{>0}$, $j \in J$, and a set of unrelated parallel machines M of cardinality m. Moreover, for every job $j \in J$ and every machine $i \in M$, we are given a random variable P_{ij}. Each job j needs to be executed on any one of the machines $i \in M$, and each machine can process at most one job at a time. If job j is processed on machine i, its processing time is P_{ij}. However, the actual realization of the processing time is only known upon j's completion and we are thus looking for a non-anticipatory scheduling policy which minimizes the expected total weighted completion time $\mathbb{E}[\sum_j w_j C_j]$, where C_j denotes the completion time of job j.

Later, in Section 5, we consider a slightly more general model where each job $j \in J$ also comes with a machine dependent release date $r_{ij} \in \mathbb{Z}_{\geq 0}$ before which job j must not be scheduled on machine i. One can think of applications where some job $j \in J$ might not be processed on a certain machine $i \in M$, i. e., $\mathbb{E}[P_{ij}] = \infty$. For the sake of simplicity of presentation, we assume in this paper that $\mathbb{E}[P_{ij}]$ is finite for all $i \in M$ and $j \in J$. But all presented results also hold for the more general case where $\mathbb{E}[P_{ij}] = \infty$ for certain pairs i, j.

Throughout this paper we assume that the random variables P_{ij}, $i \in M$, $j \in J$, take positive integral values only. The following lemma states that this assumption costs at most a factor $1 + \varepsilon$ in the objective function value.

▶ **Lemma 1.** *For any fixed $\varepsilon > 0$, while only loosing a factor $1 + \varepsilon$ in the objective function value, an arbitrary instance can be modified such that the random variables P_{ij}, $i \in M$, $j \in J$, take positive integral values only.*

[5] Note that a stochastic knapsack problem can be reinterpreted as a single machine stochastic scheduling problem where all jobs have due date 1, and with weighted earliness objective.

[6] In stochastic scheduling, this would correspond to the gap between the best static list scheduling policy and an optimal (adaptive) scheduling policy.

Proof. If $\mathbb{E}[P_{ij}] = 0$ and $r_{ij} = 0$ for some pair i, j, then we can ignore job j since it can be scheduled at no further cost on machine i at time 0. We can thus assume from now on that $\mathbb{E}[P_{ij}] > 0$ or $r_{ij} > 0$ for all pairs i, j. By scaling processing times and release dates appropriately, we can make sure that $\mathbb{E}[P_{ij}] \geq \frac{n}{\varepsilon}$ or $r_{ij} \geq \frac{n}{\varepsilon}$ for each pair i, j. As a result of this scaling step we know that, for any scheduling policy, $\mathbb{E}[C_j] \geq n/\varepsilon$ for each job $j \in J$. Rounding up all processing times to the nearest positive integer therefore increases the (expected) completion time of any job j by at most $n \leq \varepsilon\,\mathbb{E}[C_j]$. The overall increase in the objective function is thus bounded by a factor $1 + \varepsilon$. ◀

Given that all processing times are integral, we can obviously assume with no further loss of generality that jobs can only be started at integral points in time $t \in \mathbb{Z}_{\geq 0}$. In order to write down an LP relaxation in time-indexed variables, we require a fair amount of information about the exact probability distributions of random variables P_{ij}. More precisely, besides the expectations $\mathbb{E}[P_{ij}]$, we also need the values

$$q_{ijr} := \Pr[P_{ij} \geq r + 1] \qquad \text{for } i \in M,\ j \in J, \text{ and } r \in \mathbb{Z}_{\geq 0}.$$

This, of course, raises questions about the input size of the problem. Here we make the following assumption. In the input we are given, for each job $j \in J$ and each machine $i \in M$, the expected processing time $\mathbb{E}[P_{ij}]$. Moreover, we have access to an oracle which, for any triple i, j, r returns q_{ijr}. We emphasize that, in order for our approach to work, it suffices to get these values within some finite precision at the expense of an additional factor $1 + \varepsilon$ in the performance guarantee of our algorithms. More precisely, it suffices to get the values q_{ijr} rounded to multiples of ε/n, which, in particular, can be encoded polynomially in the input size. Notice that such an oracle can be simulated by a polynomial-time Monte Carlo algorithm that can sample from the distribution of the random variables P_{ij}. Having said that, in order to keep the presentation simple we neglect these aspects throughout the paper and assume that we have access to the exact values q_{ijr}.

In the analysis of our algorithm we need the following standard property of the moments of random variable P_{ij}.

▶ **Lemma 2.** *Let $j \in J$ and $i \in M$. Then,*

$$\sum_{r \in \mathbb{Z}_{\geq 0}} q_{ijr} = \mathbb{E}[P_{ij}] \qquad \textit{and} \qquad \sum_{r \in \mathbb{Z}_{\geq 0}} (r + \tfrac{1}{2})\, q_{ijr} = \frac{1 + \mathbb{CV}[P_{ij}]^2}{2}\, \mathbb{E}[P_{ij}]^2 \ ,$$

where $\mathbb{CV}[P_{ij}]^2 := \big(\mathbb{E}[P_{ij}^2] - E[P_{ij}]^2\big)/\mathbb{E}[P_{ij}]^2$ is the squared coefficient of variation of P_{ij}.

The proof of the lemma is based on standard results for the nth moment of a random variable, see, e. g. [7, V.6, Lemma 1].

3 Time-indexed LP relaxation

In the following we derive an LP relaxation of the stochastic scheduling problem under consideration. For a given non-anticipatory scheduling policy Π, let x_{ijt} be the probability that Π starts job $j \in J$ on machine $i \in M$ at time $t \in \mathbb{Z}_{\geq 0}$. Notice that this random decision may depend on the actual processing times of other jobs started by Π before time t. On the other hand, due to the non-anticipatory nature of policy Π, the random variable P_{ij} is independent of Π's random decision to start job j on machine i at time t.

As the x_{ijt}'s are going to be the variables of our LP relaxation, we derive crucial properties that are going to be the constraints of the LP relaxation. If job $j \in J$ is started on machine $i \in M$ at time $t \in \mathbb{Z}_{\geq 0}$, due to the non-anticipative nature of policy Π, j's expected completion time is $t + \mathbb{E}[P_{ij}]$. Thus, by linearity of expectation, the expected completion time of j is

$$\mathbb{E}[C_j] = \sum_{i \in M} \sum_{t \in \mathbb{Z}_{\geq 0}} x_{ijt} \left(t + \mathbb{E}[P_{ij}]\right) \ .$$

A more careful look at j's behavior reveals the following property. Conditioning on j being started on machine i at time t, the probability that j is still occupying machine i within the later time interval $[s, s+1]$, $s \in \mathbb{Z}_{\geq t}$, is equal to $q_{ij\,s-t}$ by definition. Unconditioning yields

$$\Pr\bigl[i \text{ processes } j \text{ in } [s, s+1]\bigr] = \sum_{t=0}^{s} x_{ijt}\, q_{ij\,s-t} \ . \tag{1}$$

As machine i can process at most one job at a time, also the *expected* number of jobs being processed by i in $[s, s+1]$ is bounded by 1. That is, by linearity of expectation,

$$\sum_{j \in J} \sum_{t=0}^{s} x_{ijt}\, q_{ij\,s-t} \leq 1 \ .$$

Finally, since policy Π has to process all jobs, we get for every job j $\sum_{i \in M} \sum_{t \in \mathbb{Z}_{\geq 0}} x_{ijt} = 1$. Thus, the probabilities x_{ijt} corresponding to policy Π form a feasible solution to the following LP relaxation, and the value of this LP solution x is equal to the expected value of the schedule produced by policy Π:

$$\begin{aligned}
\min \quad & \sum_{j \in J} w_j\, C_j^{\mathrm{LP}} & & \\
\text{s.t.} \quad & \sum_{i \in M} \sum_{t \in \mathbb{Z}_{\geq 0}} x_{ijt} = 1 & & \text{for all } j \in J, & (2)\\
& \sum_{j \in J} \sum_{t=0}^{s} x_{ijt}\, q_{ij\,s-t} \leq 1 & & \text{for all } i \in M,\ s \in \mathbb{Z}_{\geq 0}, & (3)\\
& C_j^{\mathrm{LP}} = \sum_{i \in M} \sum_{t \in \mathbb{Z}_{\geq 0}} x_{ijt}\, (t + \mathbb{E}[P_{ij}]) & & \text{for all } j \in J, & (4)\\
& x_{ijt} \geq 0 & & \text{for all } j \in J,\ i \in M,\ t \in \mathbb{Z}_{\geq 0}. &
\end{aligned}$$

Notice that the LP variables C_j^{LP} are uniquely determined by the x-variables and could as well be omitted by replacing them in the objective function with the right hand side of (4).

Also notice that this linear program suffers from infinitely many variables and constraints. We claim that this can be dealt with at the expense of an additional factor $1 + \varepsilon$ in the performance guarantee of our algorithms. For a detailed discussion and formal proof, see the full version of the paper.

▶ **Theorem 3.** *The above infinite time-indexed LP relaxation can be solved in pseudo-polynomial time in the input size. Moreover, a $(1 + \varepsilon)$-approximate LP solution can be found in time polynomial in the input size and $1/\varepsilon$.*

4 Turning an LP solution into a scheduling policy

For a feasible LP solution x, let $X_{ij} := \sum_{t \in \mathbb{Z}_{\geq 0}} x_{ijt}$ for $i \in M$, $j \in J$. LP constraints (2) imply that $\sum_{i \in M} X_{ij} = 1$ for every job $j \in J$.

Given the values X_{ij} corresponding to a feasible LP solution x, our scheduling policy Assign(X) assigns each job $j \in J$ independently at random to one machine $i \in M$ with probability X_{ij}. Then, on each machine $i \in M$, it sequences jobs assigned to i according to the WSEPT rule.

▶ **Theorem 4.** *The expected value of the schedule constructed by policy* Assign(X) *is at most* $\frac{3}{2} + \frac{\Delta}{2}$ *times the value of the underlying LP solution* x.

Notice that Theorem 4 and Theorem 3 imply the existence of a polynomial-time algorithm that, for any given instance of our stochastic scheduling problem and for any $\varepsilon > 0$, finds a scheduling policy with performance guarantee $\frac{3}{2} + \frac{\Delta}{2} + \varepsilon$. Remember that Δ upper bounds the squared coefficient of variation $\mathbb{CV}[P_{ij}]^2$ for all P_{ij}. It is not difficult to see that, instead of the random assignment of jobs to machines, we can use a deterministic assignment obtained via the method of conditional probabilities and still get the same performance guarantee.

The proof of Theorem 4 is based on a refined, somewhat more complicated policy, that takes the entire LP solution x into account and yields a worse schedule in expectation. It is therefore sufficient to prove the bound stated in Theorem 4 for this alternative policy which we refer to as Assign&Sequence(x).

Assign&Sequence(x)

1. For every job $j \in J$, choose a pair (i, t) independently at random with probability x_{ijt} and some $r \in \mathbb{Z}_{\geq 0}$ independently at random with probability $q_{ijr}/\mathbb{E}[P_{ij}]$; assign job j to machine i and set its *tentative start time* s to $s := t + r$ (we write "$j \to i, s$" for short).
2. On each machine $i \in M$, sequence all jobs assigned to i in order of increasing tentative start times; ties are broken randomly.

Notice that, as in the simpler policy Assign(X), job j is assigned to machine i with probability $\sum_{t \in \mathbb{Z}_{\geq 0}} x_{ijt} = X_{ij}$. Since Assign($X$) sequences the jobs on every machine in an optimal way, it is superior to policy Assign&Sequence(x). By construction of policy Assign&Sequence(x), the probability of assigning job $j \in J$ to machine $i \in M$ and setting its tentative start time to $s \in \mathbb{Z}_{\geq 0}$ is

$$\Pr[j \to i, s] = \sum_{t=0}^{s} x_{ijt} \, \frac{q_{ij\,s-t}}{\mathbb{E}[P_{ij}]} \quad . \tag{5}$$

We prove the following job-by-job performance guarantee for Assign&Sequence(x).

▶ **Theorem 5.** *For every job* $j \in J$*, the expected value of* j*'s completion time in the schedule constructed by policy* Assign&Sequence(x) *is at most* $(\frac{3}{2} + \frac{\Delta_j}{2})\, C_j^{\mathrm{LP}}$ *where* $\Delta_j := \max_{i \in M} \mathbb{CV}[P_{ij}]^2$.

By linearity of expectation, Theorem 5 immediately implies Theorem 4. In the proof of Theorem 5 we make use of the following lemma.

▶ **Lemma 6.** *Let* $j \in J$*,* $i \in M$*, and* $s \in \mathbb{Z}_{\geq 0}$*. If* $j \to i, s$*, then the expected total processing time of jobs that policy* Assign&Sequence(x) *schedules on machine* i *before job* j *is at most* $s + \frac{1}{2}$.

Proof. We first bound the expected total processing time of jobs $k \neq j$ with $k \to i, s'$ for some fixed $s' \in \mathbb{Z}_{\geq 0}$:

$$\sum_{k \neq j} \mathbb{E}[P_{ik}] \, \Pr[k \to i, s'] \overset{(5)}{=} \sum_{k \neq j} \sum_{t'=0}^{s'} x_{ikt'} \, q_{ik\,s'-t'} \leq 1 \qquad \text{by (3).}$$

Thus, the expected[7] total processing times of jobs processed before job j on machine i is at most

$$\sum_{k\neq j} \mathbb{E}[P_{ik}] \left(\sum_{s'=0}^{s-1} \Pr[k \to i, s'] + \tfrac{1}{2} \Pr[k \to i, s] \right) \le s + \tfrac{1}{2} \ .$$

This concludes the proof. ◀

Proof of Theorem 5. By Lemma 6 we get

$$\mathbb{E}[C_j \mid j \to i, s] \le s + \tfrac{1}{2} + \mathbb{E}[P_{ij}] \tag{6}$$

for every job $j \in J$, machine $i \in M$, and tentative start time $s \in \mathbb{Z}_{\geq 0}$. Unconditioning the expectation yields

$$\mathbb{E}[C_j] = \sum_{i\in M} \sum_{s\in\mathbb{Z}_{\geq 0}} \mathbb{E}\big[C_j \mid j \to i, s\big] \Pr[j \to i, s] \ .$$

Applying inequality (6) and equation (5) we get

$$\mathbb{E}[C_j] \le \sum_{i=1}^{m} \sum_{s\in\mathbb{Z}_{\geq 0}} \left(s + \tfrac{1}{2} + \mathbb{E}[P_{ij}]\right) \sum_{t=0}^{s} x_{ijt} \frac{q_{ij\,s-t}}{\mathbb{E}[P_{ij}]} \ .$$

Exchanging the order of summation of s and t, and setting $r := s - t$ yields

$$\begin{aligned}
\mathbb{E}[C_j] &\le \sum_{i=1}^{m} \sum_{t\in\mathbb{Z}_{\geq 0}} x_{ijt} \left(t + \mathbb{E}[P_{ij}] + \sum_{r\in\mathbb{Z}_{\geq 0}} \left(r + \tfrac{1}{2}\right) \frac{q_{ijr}}{\mathbb{E}[P_{ij}]} \right) \\
&= \sum_{i=1}^{m} \sum_{t\in\mathbb{Z}_{\geq 0}} x_{ijt} \left(t + \left(\frac{3}{2} + \frac{\mathbb{CV}[P_{ij}]^2}{2} \right) \mathbb{E}[P_{ij}] \right) \\
&\le \left(\frac{3}{2} + \frac{\Delta_j}{2} \right) C_j^{\mathrm{LP}}
\end{aligned}$$

by Lemma 2 and (4). This concludes the proof. ◀

We note that the same results can in fact be obtained by considering a weaker LP relaxation in variables y_{ijs}, corresponding to the probability that job j is being processed on machine i in time interval $[s, s+1]$.

5 Adding release dates

In this section we show how to adapt our analysis for a more general problem where each job $j \in J$ comes with a machine dependent deterministic release date $r_{ij} \in \mathbb{Z}_{\geq 0}$ before which job j must not be scheduled on machine i. To handle release dates we add one additional family of constraints to our time-indexed LP relaxation:

$$x_{ijt} = 0 \qquad\qquad \text{for all } i \in M,\ j \in J,\ t < r_{ij}.$$

These constraints are obviously fulfilled by the probabilities x_{ijt} corresponding to an arbitrary scheduling policy Π as no job may be started before it is released. We consider the same LP based policy Assign&Sequence(x) for this more general problem.

[7] Notice that the expectation is taken with respect to both the random decisions of our policy Assign&Sequence(x) as well as the random processing times of jobs $k \neq j$.

▶ **Theorem 7.** *In the presence of release dates, for every job $j \in J$, the expected value of j's completion time in the schedule constructed by policy* ASSIGN&SEQUENCE(x) *is at most* $(2 + \Delta_j)\, C_j^{\mathrm{LP}}$ *where* $\Delta_j := \max_{i \in M} \mathbb{CV}[P_{ij}]^2$.

The proof of Theorem 7 is almost identical to the proof of Theorem 5, and contained in the full version of the paper. We conclude this section with the following result.

▶ **Corollary 8.** *In the presence of release dates, the expected value of the schedule constructed by policy* ASSIGN&SEQUENCE(x) *is at most* $2 + \Delta$ *times the value of the underlying LP solution* x*. Thus, for any given instance of the stochastic scheduling problem and for any* $\varepsilon > 0$*, a* $(2 + \Delta + \varepsilon)$*-approximate scheduling policy can be found in polynomial time.*

6 Tightness of Performance Bounds

In this section, we argue that our results cannot be easily improved, because both LP relaxation as well as our scheduling policies have an optimality gap of $\Theta(\Delta)$.

▶ **Theorem 9.** *Even for the special case of a single machine, the multiplicative gap between the expected value of an optimal policy and the value of an optimal LP solution can be as large as* $\Delta/2$.

This is remarkable and somewhat surprising since the corresponding time-indexed linear program for the deterministic single machine scheduling problem has the same optimal value as an optimal schedule.

▶ **Theorem 10.** *Even for the special case of identical parallel machines, the performance ratio of any fixed-assignment policy can be as large as* $\frac{(1-\delta)\Delta}{2}$ *for any* $\delta > 0$*, for large enough number of machines* m.

For the proof of these two theorems we refer to the full version of the paper.

7 Execution of Scheduling Policies

We have argued that the policy we propose can be computed in polynomial time, but so far did not discuss the computation time to actually execute the scheduling policy, or more generally, any stochastic scheduling policy. The major issue is how, and with which computational effort the scheduler learns about the next job completion when executing a set of jobs. Probabilistically, this event is described by the minimum of a set of random variables, of which we just sample while executing the policy. In general, and already if there is just one single job to be processed, there might of course be nonzero probability for a job to be exponentially longer than expected. But due to Markov's inequality, the probability for exceeding the expected processing time by an exponential factor is exponentially small, too. Therefore, with high probability the sampled processing times of jobs can be encoded polynomially in the input size of the problem. Apart from this minor issue inherent in all stochastic scheduling problems, we note that the policy ASSIGN(X) is in particular elementary [16], meaning that jobs are only started upon release times or completion times of other jobs. Hence, there is only a linear number of decision times.

8 Concluding remarks

One of the main technical contributions of this paper is to introduce the important concept of time-indexed linear programming relaxations to the area of stochastic scheduling, which yields, for the first time, performance bounds for stochastic unrelated machine scheduling that even match currently best known results for deterministic unrelated machine scheduling. Obtaining performance bounds independent of the coefficient of variation of the underlying processing times remains an interesting challenge, even for the special case of parallel machines.

Acknowledgements. The authors would like to thank Nicole Megow for helpful discussions on the topic of this paper, and Andreas S. Schulz and Benjamin Labonté for valuable comments on an earlier version of this paper. Most of the results presented in this paper were obtained during the Dagstuhl Seminar 13111 "Scheduling". The authors would like to thank the organizers and Schloss Dagstuhl for providing an enjoyable and stimulating atmosphere.

References

1 F. Afrati, E. Bampis, C. Chekuri, D. Karger, C. Kenyon, S. Khanna, I. Milis, M. Queyranne, M. Skutella, C. Stein, and M. Sviridenko. Approximation schemes for minimizing average weighted completion time with release dates. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science*, pages 32–43, New York City, NY, 1999.

2 C. Chekuri, R. Motwani, B. Natarajan, and C. Stein. Approximation techniques for average completion time scheduling. *SIAM Journal on Computing*, 31:146–166, 2001.

3 C.F. 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. *Operations Research*, 54:464– 474, 2006.

4 F. A. Chudak. A min-sum 3/2-approximation algorithm for scheduling unrelated parallel machines. *Journal of Scheduling*, 2:73–77, 1999.

5 J. Correa and M. Wagner. LP-based online scheduling: From single to parallel machines. *Mathematical Programming*, 119:109–136, 2008.

6 B. Dean, M. X. Goemans, and J. Vondrák. Approximating the stochastic knapsack problem: The benefit of adaptivity. *Mathematics of Operations Research*, 33:945–964, 2008.

7 W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 2. Wiley Series in Probability and Mathematical Statistics, 2nd edition, 1971.

8 R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5:287–326, 1979.

9 A. Gupta, R. Krishnaswamy, M. Molinaro, and R. Ravi. Approximation algorithms for correlated knapsacks and non-martingale bandits. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science*, pages 827–836. IEEE Computer Society, 2011.

10 L. A. Hall, A. S. Schulz, D. B. Shmoys, and J. Wein. Scheduling to minimize average completion time: Off-line and on-line approximation algorithms. *Mathematics of Operations Research*, 22:513–544, 1997.

11 H. Hoogeveen, P. Schuurman, and G. J. Woeginger. Non-approximability results for scheduling problems with minsum criteria. *INFORMS Journal on Computing*, 13:157–168, 2001.

12 J. K. Lenstra, A. H. G. Rinnooy Kan, and P. Brucker. Complexity of machine scheduling problems. *Annals of Discrete Mathematics*, 1:343–362, 1977.

13 N. Megow. Approximation in stochastic scheduling. Invited talk at the Dagstuhl Seminar 13111 "Scheduling", Schloss Dagstuhl, Wadern, 2013.

14 N. Megow, M. Uetz, and T. Vredeveld. Models and algorithms for stochastic online scheduling. *Mathematics of Operations Research*, 31(3):513–525, 2006.

15 N. Megow and T. Vredeveld. Approximation in preemptive stochastic online scheduling. In Y. Azar and T. Erlebach, editors, *Algorithms - ESA 2006*, volume 4168 of *Lecture Notes in Computer Science*, pages 516–527. Springer-Verlag, 2006.

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

17 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.

18 M. Queyranne. Structure of a simple scheduling polyhedron. *Mathematical Programming*, 58:263–285, 1993.

19 M. H. Rothkopf. Scheduling with random service times. *Management Science*, 12:703–713, 1966.

20 M. Scharbrodt, T. Schickinger, and A. Steger. A new average case analysis for completion time scheduling. *Journal of the ACM*, 53:121–146, 2006.

21 A. S. Schulz. Scheduling to minimize total weighted completion time: Performance guarantees of LP-based heuristics and lower bounds. In W. H. Cunningham, S. T. McCormick, and M. Queyranne, editors, *Integer Programming and Combinatorial Optimization*, volume 1084 of *Lecture Notes in Computer Science*, pages 301–315. Springer, 1996.

22 A. S. Schulz. Stochastic online scheduling revisited. In B. Yang, D.-Z. Du, and C. Wang, editors, *Combinatorial Optimization and Applications*, volume 5165 of *Lecture Notes in Computer Science*, pages 448–457. Springer, 2008.

23 A. S. Schulz and M. Skutella. Scheduling unrelated machines by randomized rounding. *SIAM Journal on Discrete Mathematics*, 15:450–469, 2002.

24 P. Schuurman and G. J. Woeginger. Polynomial time approximation algorithms for machine scheduling: Ten open problems. *Journal of Scheduling*, 2:203–213, 1999.

25 J. Sethuraman and M. S. Squillante. Optimal scheduling of multiclass parallel machines. In *Proceedings of the 10th Annual ACM–SIAM Symposium on Discrete Algorithms*, pages 963–964, 1999.

26 M. Skutella. Convex quadratic and semidefinite programming relaxations in scheduling. *Journal of the ACM*, 48:206–242, 2001.

27 M. Skutella and M. Uetz. Stochastic machine scheduling with precedence constraints. *SIAM Journal on Computing*, 34:788–802, 2005.

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

29 W. E. Smith. Various optimizers for single-stage production. *Naval Research and Logistics Quarterly*, 3:59–66, 1956.

30 A. Souza and A. Steger. The expected competitive ratio for weighted completion time scheduling. *Theory of Computing Systems*, 39:121–136, 2006.

31 M. Uetz. When greediness fails: Examples from stochastic scheduling. *Operations Research Letters*, 31:413–419, 2003.

32 Gideon Weiss. Approximation results in parallel machines stochastic scheduling. *Annals of Operations Research*, 26:195–242, 1990.

33 Gideon Weiss. Turnpike optimality of Smith's rule in parallel machines stochastic scheduling. *Mathematics of Operations Research*, 17:255–270, 1992.

34 L. A. Wolsey. Mixed integer programming formulations for production planning and scheduling problems. Invited talk at the 12th International Symposium on Mathematical Programming, MIT, Cambridge, 1985.