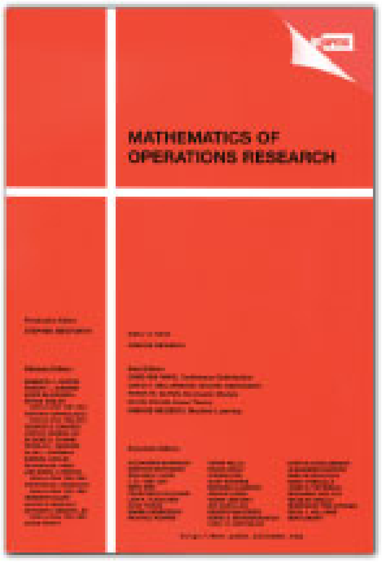

Mathematics of Operations Research

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

Unrelated Machine Scheduling with Stochastic Processing Times

Martin Skutella, Maxim Sviridenko, Marc Uetz

MATHEMATICS OF OPERATIONS RESEARCH
Vol. 41, No. 3, August 2016, pp. 851–864
ISSN 0364-765X (print) | ISSN 1526-5471 (online)

http://dx.doi.org/10.1287/moor.2015.0757

Unrelated Machine Scheduling with Stochastic Processing Times

Martin Skutella
Technische Universität Berlin, Institut für Mathematik, 10623 Berlin, Germany,
martin.skutella@tu-berlin.de

Maxim Sviridenko
Yahoo! Labs, New York, New York 10018, sviri@yahoo-inc.com

Marc Uetz
Department of Applied Mathematics, University of Twente, 7500 AE Enschede, Netherlands,
m.uetz@utwente.nl

Two important characteristics encountered in many real-world scheduling problems are heterogeneous processors and a certain degree of uncertainty about the processing times 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. By means of a novel time-indexed linear programming relaxation, we show how to compute in polynomial time a scheduling policy with provable performance guarantee for the stochastic version of the unrelated parallel machine scheduling problem with the weighted sum of completion times objective. Our performance guarantee depends on the squared coefficient of variation of the processing times and we show that this dependence is tight. Currently best-known bounds for deterministic scheduling problems are contained as special cases.

Keywords: stochastic scheduling; approximation algorithm; linear programming relaxation
MSC2000 subject classification: Primary: 90B36, 68M20, 90C27, 90C59, 68W25, 68W40, 68Q25
OR/MS subject classification: Primary: production/scheduling; secondary: approximations/heuristic
History: Received April 24, 2014; revised June 25, 2015, and July 17, 2015. Published online in *Articles in Advance* February 16, 2016.

1. Introduction. The problem to minimize the total weighted completion time on unrelated parallel machines, denoted $R|(r_{ij})|\sum w_j C_j$ 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 nonpreemptively 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 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 (Lenstra et al. [13]) but there do exist polynomial time approximation schemes (Afrati et al. [1], Skutella and Woeginger [32]). 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 deterministic unrelated parallel machines have performance guarantees 3/2 and 2, for the problem without and with release dates, respectively, Chudak [4], Schulz and Skutella [26], Sethuraman and Squillante [28], and Skutella [30]. Improving these bounds is considered to be among the most important open problems in scheduling (see Schuurman and Woeginger [27]), 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 nonanticipatory scheduling policy. Intuitively, a nonanticipatory scheduling policy must make its scheduling decisions at time t based on the observed past up to time t as well as the a priori knowledge of the input data of the problem. A policy, however, is not allowed to anticipate information about the future, i.e., the actual realizations of the processing times of jobs that have not yet been completed by time t. For a thorough discussion and definition of nonanticipatory scheduling policies, see subsequent §2. As all previous work in the area, we assume that the random variables P_{ij} are stochastically independent across jobs. For any given nonanticipatory 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 [34] for deterministic single machine scheduling, Rothkopf [22] proved in 1966 that the WSEPT rule (weighted shortest expected processing time first: schedule jobs in order of nonincreasing ratios $w_j/\mathbb{E}[P_j]$) 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

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 $\mathbb{CV}^2[P_{ij}] = \mathbb{V}\mathrm{ar}[P_{ij}]/\mathbb{E}^2[P_{ij}]$ for all P_{ij}. The third column shows the results for $\mathbb{CV}[P_{ij}] \le 1$; e.g., uniform, exponential, or Erlang distributions. As usual in stochastic scheduling, these multiplicative bounds hold with respect to the expected performance of any nonanticipatory scheduling policy.

Stochastic scheduling model	Worst-case performance guarantee: Arbitrary P_{ij}	Worst-case performance guarantee: $\mathbb{CV}[P_{ij}] \le 1$	Reference
$\mathrm{P}\|\|\mathbb{E}[\sum w_j C_j]$	$1 + \frac{(m-1)(\Delta+1)}{2m}$	$2 - 1/m$	Möhring et al. [18]
$\mathrm{P}\|r_j\|\mathbb{E}[\sum w_j C_j]$	$2 + \Delta$	3	Schulz [25]
$\mathrm{R}\|\|\mathbb{E}[\sum w_j C_j]$	$1 + \frac{\Delta+1}{2} + \varepsilon$	$2 + \varepsilon$	This paper
$\mathrm{R}\|r_{ij}\|\mathbb{E}[\sum w_j C_j]$	$2 + \Delta + \varepsilon$	$3 + \varepsilon$	This paper

identical parallel machines (Weiss [37, 38]), the first constant factor approximation algorithms for stochastic scheduling on identical parallel machines have been obtained in 1999 by Möhring et al. [18]. Next to a linear programming (LP) based analysis of the WSEPT rule, they define list scheduling policies that 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 nonanticipatory 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 [31]. Subsequently, in line with earlier work by Chou et al. [3], Megow et al. [16] 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 by Megow et al. [16], are based on one and the same linear programming relaxation, namely, that of Möhring et al. [18]. With respect to the underlying relaxation, Schulz [25] 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 Möhring et al. [18]. Nevertheless, in comparison to Megow et al. [16], Schulz obtains improved and simpler results through the clever use of an optimal solution to an equivalent time-indexed LP relaxation for deterministic scheduling.

Two other research directions are related to our work, yet for different models and independent of the techniques of Möhring et al. [18] 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. [23] and Souza and Steger [35], who analyze the expected competitive ratio rather than the expected performance of a policy Π. In that model, one analyzes the ratio $\mathbb{E}[v(\Pi)/v(\text{Offline-Opt})]$, where $v(\Pi)$ is the objective function value of a policy Π. In this paper, however, we follow Megow et al. [16], Möhring et al. [18], Schulz [25], and Skutella and Uetz [31] and focus on the ratio $\mathbb{E}[v(\Pi)]/\mathbb{E}[v(\Pi^{\mathrm{Opt}})]$ instead, where Π^{Opt} is an optimal, nonanticipatory scheduling policy, and not the point-wise optimal offline solution as in Scharbrodt et al. [23] and Souza and Steger [35]. We refer to Scharbrodt et al. [23] for a discussion of the pros of their model as opposed to the one considered here. Note that here we restrict to a weaker adversary, since the adversary is also bound to be nonanticipatory. For a thorough discussion of this issue in the context of competitive analysis, see also Koutsoupias and Papadimitriou [12].

Note that all results discussed so far are restricted to the model with identical parallel machines. Table 1 gives an overview of currently best-known performance bounds in nonpreemptive stochastic scheduling with a min-sum objective, next to the new 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 [39] and Queyranne [20] (for single machine scheduling) and was later generalized to identical parallel machines by Schulz [24] 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. [18], and show how to put a novel, time-indexed linear programming relaxation to work in stochastic machine scheduling. We are optimistic that this 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 (see Schulz and Skutella [26]). In spite of that, it remained a major open problem how to come up with a meaningful time-indexed LP relaxation for stochastic scheduling problems (see Megow [14]). 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 that cannot be easily described by time-indexed variables.

In §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. Note that even for simple scheduling policies like the WSEPT rule, determining this probability is nontrivial. The machine capacity constraints say that each machine can process at most one job at a time, and formulating this is rather easy for deterministic unrelated machine scheduling. 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} in order to formulate that constraint.

Notice that, because of the stochastic nature of processing times, even a schedule produced by an optimal policy can lead to the situation where infinitely many variables x_{ijt} may take positive values. This already happens for one machine when considering the start time of the second of two jobs with exponentially distributed processing time. Nonetheless, we show how to overcome this difficulty. Indeed, we show that we can compute an LP-solution in polynomial time that approximates the optimal LP solution with arbitrary precision.

In §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 Schulz and Skutella [26]. 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 that is determined by a two-stage random process. We show how to extend our results to the setting with release dates in §5.

The scheduling policies that we use to obtain our results fall into the special class of *fixed assignment policies*. That means that, already at time zero, each job is immediately and irrevocably assigned to a machine. In particular, 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 §6 we prove a lower bound of $\Delta/2$ on the performance guarantee of *any* fixed assignment policy. Moreover, we can also show that the LP relaxation that we use can have an optimality gap in the same order of magnitude.

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

In §7 we discuss how to execute our policy with polynomial running time and in §8 we discuss an alternative linear programming relaxation.

Parallel to Stochastic Knapsack. There is an interesting parallel of our work with that on stochastic knapsack problems. Indeed, stochastic knapsack problems can be reinterpreted as single machine stochastic scheduling problems where all jobs have due date 1, and with weighted earliness objective. 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. In stochastic scheduling, the adaptivity gap would correspond to the gap between the best static list scheduling policy and an optimal adaptive scheduling policy. 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 Möhring et al. [18] 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 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, we are given a random variable P_{ij} for every job $j \in J$ and every machine $i \in M$. 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 random according to P_{ij}. In some applications job $j \in J$ cannot 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 $j \in J$. Yet all presented results also hold for the more general case where $\mathbb{E}[P_{ij}] = \infty$ for certain pairs (i, j). Later in §5, we consider a slightly more general model where each job $j \in J$ can start on machine $i \in M$ only at time $r_{ij} \in \mathbb{Z}_{\geq 0}$ or later. The parameters r_{ij} are called *machine dependent release dates*.

In the stochastic scheduling model, the actual realization of the processing time of a job j is only known upon j's completion and we are looking for a nonanticipatory scheduling policy Π that 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. We next define the notion of a scheduling policy.

Scheduling Policies. Here, we confine ourselves with the intuitive, dynamic view on scheduling policies that puts stochastic scheduling in the framework of stochastic dynamic optimization; see, e.g., Ross [21], and refer to Möhring et al. [17] for the analytic definition of nonanticipatory stochastic scheduling policies as mappings $\Pi: \mathbb{R}^n \to \mathbb{R}^n$ that map processing times to completion times of jobs. A scheduling policy takes actions at points in time $t \geq 0$, starting at time $t = 0$. An action at a given 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 then due at time 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 way to think about this dynamic process is as follows. There is an oracle with access to the realized processing times of the jobs, and at any decision time t with jobs J_t in process or started, it computes $t' := \min_{j \in J_t} C_j$. The next decision time is then $t^{\text{next}} := \min\{t', t^*\}$. The information revealed to policy Π is the time t^{next}, along with the set of jobs finished at t^{next}. If there are also release dates r_{ij}, this is taken into account in the obvious way.

In the class of nonanticipatory scheduling policies, the existence of a policy that minimizes the expected performance for the model considered here follows from Theorem 4.2.6 in Möhring et al. [17], because of the linearity of the objective function (which is in particular lower semicontinuous). Observe that such a scheduling policy can be represented by a finite, yet generally exponential size decision tree. Finiteness is not clear a priori unless we restrict the set of scheduling policies, for example, to being elementary, where jobs are restricted to be started only upon completion times or release dates (or in other words, $t^* = \infty$). But as we argue below, we can restrict essentially w.l.o.g. to discrete processing time distributions, and then the resulting decision tree of any reasonable policy is indeed finite. Also observe that a policy may gain information about remaining processing times of jobs over time, but at any point in time it has only access to distributional information about remaining processing times of unfinished jobs, conditioned on the state of the schedule at time t. That means in particular that, for a given and fixed state at a time t, a scheduling policy cannot choose different actions dependent on the future of t. Allowing for randomization, however, is possible, meaning that a policy runs a lottery (independent of the future) among all feasible actions at time t.

A brief, concrete example may help: Imagine a job j, which has processing time either small (1) 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 $t^* = t + 1$, because then it knows with certainty what the actual processing time of job j is. Using such building blocks, one can even show that an optimal scheduling policy for the setting considered here is generally not work conserving, i.e., there exist examples where an optimal scheduling policy must leave machines deliberately idle even though there are yet unscheduled jobs; see Uetz [36].

Discretization. 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 losing 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.*

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 $\mathbb{E}[P_{ij}] \geq n/\varepsilon$ or $r_{ij} \geq n/\varepsilon$ 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 $n \le \varepsilon \mathbb{E}[C_j]$. The overall increase in the objective function is thus bounded by a factor $1+\varepsilon$. To be a bit more precise, for the rounded instance, we can simulate any given policy Π for the original instance, yielding a new scheduling policy that achieves expected completion times of any job j that exceed those of Π by at most $n \le \varepsilon \mathbb{E}[C_j]$. This is because the rounding will not yield any "new" states that Π would not know how to handle. Moreover, any scheduling policy for the rounded instance can be translated back to a scheduling policy for the original instance with the same objective value by adding deliberate idle time where necessary. This shows that the rounding is indeed no loss of generality, except for the additional multiplicative factor $(1+\varepsilon)$. $\square$

Input size and further preliminaries. 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 $t \in \mathbb{Z}_{\ge 0}$. 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

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

This, of course, raises questions about the input size of the problem. Here, we make the following assumption. In the input we are given the expected processing time $\mathbb{E}[P_{ij}]$ for each job $j \in J$ and each machine $i \in M$. 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}, the proof of which is based on standard results for the nth moment of a random variable; see, e.g., Feller [7, V.6, Lemma 1]. For the sake of completeness, we present the simple proof here.

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

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

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

Proof. First,

$$\sum_{r \ge 0} q_{ijr} = \sum_{r \ge 0} \sum_{q \ge r} \Pr[P_{ij} = q+1] = \sum_{r \ge 0} (r+1) \Pr[P_{ij} = r+1] = \mathbb{E}[P_{ij}].$$

For the second claim,

$$\sum_{r \ge 0} (r + \tfrac{1}{2}) q_{ijr} = \sum_{r \ge 0} \sum_{q \ge r} (r + \tfrac{1}{2}) \Pr[P_{ij} = q+1] = \sum_{r \ge 0} \tfrac{1}{2} (r+1)^2 \Pr[P_{ij} = r+1] = \tfrac{1}{2} \mathbb{E}[P_{ij}^2].$$

The claim now follows by definition of the coefficient of variation. $\square$

3. Time-indexed LP relaxation. In the following, we derive an LP relaxation of the stochastic scheduling problem under consideration. For a given nonanticipatory scheduling policy Π, let x_{ijt} be the probability that Π starts job $j \in J$ on machine $i \in M$ at time $t \in \mathbb{Z}_{\ge 0}$. 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, because of the nonanticipatory 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 $t \in \mathbb{Z}_{\ge 0}$, because of the nonanticipative 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

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

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_{ijs-t} by definition. Unconditioning yields

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

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,

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

Finally, since policy Π has to process all jobs, we get $\sum_{i \in M} \sum_{t \in \mathbb{Z}_{\geq 0}} x_{ijt} = 1$, for every job j. 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:

$$\min \sum_{j \in J} w_j C_j^{\mathrm{LP}}$$

$$\text{s.t.} \sum_{i \in M} \sum_{t \in \mathbb{Z}_{\geq 0}} x_{ijt} = 1 \quad \text{for all } j \in J, \tag{2}$$

$$\sum_{j \in J} \sum_{t=0}^{s} x_{ijt} q_{ij\,s-t} \leq 1 \quad \text{for all } i \in M, \quad s \in \mathbb{Z}_{\geq 0}, \tag{3}$$

$$C_j^{\mathrm{LP}} = \sum_{i \in M} \sum_{t \in \mathbb{Z}_{\geq 0}} x_{ijt}\,(t + \mathbb{E}[P_{ij}]) \quad \text{for all } j \in J, \tag{4}$$

$$x_{ijt} \geq 0 \quad \text{for all } j \in J, \quad i \in M, \quad t \in \mathbb{Z}_{\geq 0}.$$

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 LP is not a formulation but only relaxation of the scheduling problem. This can be seen by realizing that the LP allows to greedily "pack" fractions of jobs in a way that cannot be translated back into any feasible scheduling policy. This is exactly what is exploited in the example that we give in the proof of Theorem 6, where we show that the LP has a large optimality gap even on a single machine. Intuitively, the LP is allowed to neglect implicit temporal dependencies that upper bound the probabilities for jobs being started. For instance, in the example where we have two jobs with exponentially distributed processing times that must be scheduled on a single machine, we can define LP solutions where both jobs are started with probability one at some finite point in time. However in any schedule, for any time $t \geq 0$ there is nonzero probability for one job to be started at time t or later.

That said, it becomes clear that the linear program generally suffers from infinitely many variables and constraints. Indeed, it is true that infinitely many variables may be needed in order to map an optimal scheduling policy Π to a corresponding LP solution. Consider again the same two jobs with exponentially distributed processing times that must be scheduled on a single machine, and consider the policy that schedules the jobs one after another. Then the start time of the second job cannot be bounded, i.e., for any $t \geq 0$, the second job is started with positive probability at time t or later. Despite this peculiarity, we can show that an *optimal* solution to the LP relaxation does get along with finite support. More precisely, we give a pseudopolynomial upper bound on the largest time index t such that $x_{ijt} > 0$ in an optimal LP solution for any $j \in J$ and $i \in M$. Thereby, the problem can be overcome at the expense of an additional factor $1 + \varepsilon$ in the performance guarantee of our algorithms.

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

PROOF. Any reasonable scheduling policy produces a schedule where the expected completion time of every job is at most

$$D := \max_{i \in M} \sum_{j \in J} \mathbb{E}[P_{ij}].$$

In particular, the expected value of the schedule is at most $U := D\sum_{j\in J} w_j$, which is thus also an upper bound on the optimal LP solution. Moreover, let

$$R := 2n \max_{j\in J,\, i\in M} \mathbb{E}[P_{ij}].$$

Lemma 3. *Let $T := 2U + R$. There is an optimal solution x to the LP relaxation such that $x_{ijt} = 0$ for $i \in M$, $j \in J$, and $t > T$.*

Proof. Consider an optimal LP solution x and an arbitrary machine $i \in M$. Since U is an upper bound on the value corresponding to x and since $w_j \geq 1$ for each job $j \in J$, it follows from Markov's inequality that

$$\sum_{j\in J}\sum_{t\geq 2U} x_{ijt} \leq \frac{1}{2}. \tag{5}$$

Using Markov's inequality once again, we derive that for each job $j \in J$ and $r \geq R$

$$q_{ijr} = \Pr[P_{ij} \geq r+1] \leq \Pr[P_{ij} \geq 2n\,\mathbb{E}[P_{ij}]] \leq \frac{1}{2n}. \tag{6}$$

We now define a new LP solution x' by letting

$$x'_{ijt} := \begin{cases} x_{ijt} & \text{for } t < T,\\ \sum_{t'\geq T} x_{ijt'} & \text{for } t = T,\\ 0 & \text{for } t > T. \end{cases}$$

By definition, x' has the desired property stated in the lemma and its objective function value is bounded from above by the value of x. It remains to prove that x' is a feasible LP solution. It is clear that x' fulfills LP constraints (2) since x fulfills these constraints. It is, however, less obvious that x' also fulfills all LP constraints (3). For $s < T$

$$\sum_{j\in J}\sum_{t=0}^{s} x'_{ijt} q_{ijs-t} = \sum_{j\in J}\sum_{t=0}^{s} x_{ijt} q_{ijs-t} \leq 1$$

since x is a feasible LP solution and thus satisfies (3). Finally, for $s \geq T$,

$$\begin{aligned}\sum_{j\in J}\sum_{t=0}^{s} x'_{ijt} q_{ijs-t} &= \sum_{j\in J}\sum_{t=0}^{2U-1} x_{ijt} q_{ijs-t} + \sum_{j\in J}\sum_{t=2U}^{s} x'_{ijt} q_{ijs-t}\\ &\leq \frac{1}{2n}\sum_{j\in J}\sum_{t=0}^{2U-1} x_{ijt} + \sum_{j\in J}\sum_{t\geq 2U} x_{ijt} \leq \frac{1}{2} + \frac{1}{2} = 1,\end{aligned}$$

where the first inequality follows from (6) and the definition of x', and the last inequality follows from (2) and (5). This concludes the proof of Lemma 3. □

It can be easily derived from the proof that the property claimed in Lemma 3 indeed holds for any optimal LP solution x. It is important to notice that, for such a solution x, constraints (3) for $s > T$ are implied by the constraint for $s = T$ since q_{ijr} is nonincreasing in r. We have thus reduced the problem of finding an optimal LP solution to solving a *truncated* time-indexed LP of pseudopolynomial size. It is well known that such time-indexed LPs can be solved approximately in polynomial time at the expense of losing a factor $1+\varepsilon$ in the objective function. The underlying idea is to replace the discretization of time into unit size intervals by a slightly rougher discretization using intervals of geometrically increasing lengths. We refer to Skutella [29, Chapter 2.13] for a more thorough discussion of this point. This concludes the proof of Theorem 1. □

We finally mention that Theorem 1 can be easily generalized to the more general problem with release dates studied in §5.

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, 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. To formulate our main theorem, remember that Δ upper bounds the squared coefficient of variation $\mathbb{CV}[P_{ij}]^2$ for all P_{ij}.

Theorem 2. *The expected value of the schedule constructed by policy* Assign(X) *is at most* $3/2 + \Delta/2$ *times the value of the underlying LP solution* x*. Thus, by Theorem 1, for any given instance of the stochastic scheduling problem and for any* $\varepsilon > 0$, *a* $(3/2 + \Delta/2 + \varepsilon)$*-approximate scheduling policy can be found in polynomial time.*

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. A similar approach for deterministic scheduling was used by Schulz and Skutella [26].

The proof of Theorem 2 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 2 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

$$\Pr[j \to (i, s)] = \sum_{t=0}^{s} x_{ijt} \frac{q_{ijs-t}}{\mathbb{E}[P_{ij}]}. \tag{7}$$

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

Theorem 3. *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* $(3/2 + \Delta_j/2)C_j^{\mathrm{LP}}$, *where* $\Delta_j := \max_{i\in M} \mathbb{CV}[P_{ij}]^2$.

By linearity of expectation, Theorem 3 immediately implies Theorem 2. In the proof of Theorem 3 we make use of the following lemma.

Lemma 4. *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 $s' \in \mathbb{Z}_{\geq 0}$:

$$\sum_{k\neq j} \mathbb{E}[P_{ik}] \Pr[k \to (i, s')] \stackrel{(7)}{=} \sum_{k\neq j} \sum_{t'=0}^{s'} x_{ikt'} q_{iks'-t'} \leq 1 \quad \text{by (3)}.$$

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

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

where the expectation is taken with respect to both the random decisions of policy Assign&Sequence(x) as well as the random processing times of jobs $k \neq j$. This concludes the proof of Lemma 4. □

Proof of Theorem 3. By Lemma 4 we get

$$\mathbb{E}[C_j \mid j \to (i,s)] \le s + \tfrac{1}{2} + \mathbb{E}[P_{ij}] \tag{8}$$

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

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

Applying inequality (8) and equation (7) we get

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

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

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

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 $[s, s+1]$; see §8.

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 $r_{ij} \in \mathbb{Z}_{\ge 0}$ 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:

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

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.

Theorem 4. *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 4 is almost identical to the proof of Theorem 3; for the sake of completeness we nevertheless present it here.

Proof of Theorem 4. Note that the release dates of all jobs that have tentative start times less than s is at most s. Thus, by Lemma 4, we get

$$\mathbb{E}[C_j \mid j \to (i,s)] \le s + s + \tfrac{1}{2} + \mathbb{E}[P_{ij}] \tag{9}$$

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

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

Applying inequality (9) and Equation (7) we get

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

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

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

by Lemma 2 and (4). This concludes the proof of Theorem 4. □

We conclude this section with the following result for the model with release dates.

THEOREM 5. *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. We argue that our results cannot be easily improved, because both LP relaxation as well as our scheduling policies have an optimality gap of $\Theta(\Delta)$. The following theorem is 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 6. *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* $\Delta/2$.

PROOF OF THEOREM 6. Consider the following single machine instance. We are given a set of n identical jobs $J = \{0,\ldots,n-1\}$ with unit weight $w_j = 1$ and stochastic processing times

$$P_j = \begin{cases}1 & \text{with probability } 1-1/(n+1),\\ n^4+n^3-n & \text{with probability } 1/(n+1),\end{cases} \quad \text{for all } j\in J.$$

In particular, $\mathbb{E}[P_j] = n^3$, $\mathbb{E}[P_j^2] = n^7+n^6-2n^4+n$, and $\mathbb{CV}[P_j]^2 = n - 2/n^2 + 1/n^5$. Also notice that

$$q_{jr} = \begin{cases}1 & \text{for } r=0,\\ 1/(n+1) & \text{for } r = 1,\ldots,n^4+n^3-n-1,\\ 0 & \text{otherwise,}\end{cases} \quad \text{for all } j\in J.$$

Moreover, the expected value of the schedule found by the optimal WSEPT policy is equal to

$$\sum_{j=0}^{n-1}(j+1)n^3 = \tfrac{1}{2}n^4(n+1).$$

Consider the LP solution given by

$$x_{j,t} = \begin{cases}1/n & \text{for } t = jn,\ldots,(j+1)n-1,\\ 0 & \text{otherwise,}\end{cases} \quad \text{for all } j\in J.$$

It is not difficult to check that x is a feasible LP solution: notice that LP constraints (2) are fulfilled by definition of x. Moreover, the left-hand side of LP constraints (3) is maximal for $s = n^2-1$, where it takes the value

$$\sum_{j=0}^{n-1}\sum_{t=0}^{n^2-1} x_{j,t}q_{j,n^2-1-t} = \sum_{j=0}^{n-2}\frac{1}{n+1} + \sum_{t=0}^{n^2-2}\frac{x_{n-1,t}}{n+1} + x_{n-1,n^2-1} = \frac{n-1}{n+1} + \frac{n-1}{n(n+1)} + \frac{1}{n} = 1.$$

The value of the LP solution x is

$$\sum_{j=0}^{n-1}\sum_{t=jn}^{(j+1)n-1}\tfrac{1}{n}(t+n^3) = \frac{1}{n}\sum_{t=0}^{n^2-1}(t+n^3) = \frac{n^2(n^2-1)}{2n} + n^4 = n^4 + \tfrac{1}{2}n^3 - \tfrac{1}{2}n.$$

Thus, the multiplicative gap between the expected value of the optimal WSEPT policy and the value of an optimal LP solution is at least

$$\frac{n^4(n+1)}{2n^4+n^3-n} = \frac{n}{2} + \Theta(1) = \frac{\Delta}{2} + \Theta(1),$$

where $\Delta = \mathbb{CV}[P_j]^2 = n - 2/n^2 + 1/n^5$. □

Next we show that the dependence of our performance guarantee on the squared coefficients of variation $\Delta = \max_{ij} \mathbb{CV}[P_{ij}]^2$ has the right order of magnitude for all algorithms that use fixed job assignments to machines, i.e., for all policies where the assignment of all jobs to machines is fixed right in the beginning and not changed after learning about the processing time realizations of already completed jobs. This even holds for the case of identical parallel machines, where, for each job $j \in J$, there is one random variable P_j such that $P_{ij} = P_j$ for all machines $i \in M$. The following theorem shows that our approximation result cannot be significantly improved without considering adaptive policies.

Theorem 7. *Even for the special case of identical parallel machines, the performance ratio of any fixed-assignment policy can be as large as* $(1-\delta)\Delta/2$ *for any* $\delta > 0$, *for large enough number of machines* m.

Proof of Theorem 7. Let $\varepsilon > 0$ be a fixed small constant. Consider an instance consisting of m identical parallel machines and m^2 jobs with unit weights $w_j = 1$ and stochastic processing times P_j with

$$P_j = \begin{cases} 1 & \text{with probability } \dfrac{1-\varepsilon}{m}, \\[2ex] 0 & \text{with probability } 1 - \dfrac{1-\varepsilon}{m}. \end{cases}$$

The expected processing time of each job is $\mathbb{E}[P_j] = \Pr[P_j = 1] = (1-\varepsilon)/m$ and the total expected processing time of all jobs is

$$\mathbb{E}\left[\sum_{j \in J} P_j\right] = (1-\varepsilon)m.$$

The squared coefficient of variation of random variable P_j is

$$\Delta = \mathbb{CV}[P_j]^2 = \frac{m}{1-\varepsilon} - 1.$$

The probability that our instance has at least $(1/(1-\varepsilon))\mathbb{E}[\sum_{j\in J} P_j] = m$ jobs with processing times equal to one is upper bounded by $e^{-(1/4)m\varepsilon^2/(1-\varepsilon)}$ by the Chernoff bounds for the sum of independent Boolean random variables (see, e.g., Motwani and Raghavan [19, Theorems 4.1 and 4.3]). Note also that, under any realization of processing times, the value of the schedule computed by the optimal policy is upper bounded by m^3.

To derive an upper bound on the value of an optimal policy, consider the following adaptive (but nonanticipatory) policy. Initially all machines are available. Start one job on each available machine. If some of the jobs are immediately finished (i.e., they have processing time 0), then start one job per available machine again. Once we have a job that has not been finished immediately, we declare the machine where this job is processed *unavailable* and continue to assign jobs to available machines. The expected value of the schedule produced by the optimal policy can thus be bounded from above by

$$\begin{aligned} &\Pr\left[\sum_{j\in J} P_j < m\right] \mathbb{E}\left[\sum_{j \in J} P_j \,\middle|\, \sum_{j\in J} P_j < m\right] + \Pr\left[\sum_{j\in J} P_j \ge m\right] m^3 \\ &\le \mathbb{E}\left[\sum_{j\in J} P_j\right] + \Pr\left[\sum_{j\in J} P_j \ge m\right] m^3 \\ &\le (1-\varepsilon)m + e^{-(1/4)m\varepsilon^2/(1-\varepsilon)} m^3 < m, \end{aligned}$$

where the last inequality holds for large enough m.

Consider now any fixed-assignment policy that assigns k_i jobs to be processed on machine $i \in M$ with $\sum_{i\in M} k_i = m^2$. Since all jobs have identical distributions and weights, the optimal single machine policy is to process jobs according to an arbitrary permutation. The expected value of such a schedule on machine $i \in M$ is

$$\sum_{q=1}^{k_i} q \frac{1-\varepsilon}{m} = (1-\varepsilon)\frac{k_i(k_i+1)}{2m}.$$

Thus, because of convexity, the value of the fixed assignment policy is

$$(1-\varepsilon)\sum_{i\in M} \frac{k_i(k_i+1)}{2m} \ge (1-\varepsilon)\frac{m(m+1)}{2}.$$

Therefore, the worst-case ratio between an optimal fixed assignment policy and an optimal policy is at least $(1-\varepsilon)((m+1)/2)$. Since we can choose $\varepsilon > 0$ to be arbitrarily small and m arbitrarily large, we derive that this ratio is at least $(1-\delta)\Delta/2$ for any $\delta > 0$ and large enough parameter m. $\square$

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 (Möhring et al. [17]), 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. A weaker time-indexed LP relaxation. In this section we discuss a weaker time-indexed LP relaxation of the stochastic scheduling problem. Instead of using variables x_{ijt} corresponding to the probability that job j is started on machine i at time t as in §3, we use variables y_{ijs} corresponding to the probability that job j is being processed on machine i within time interval $[s, s+1]$. Then, by (1), we get

$$y_{ijs} = \sum_{t=0}^{s} x_{ijt} q_{ijs-t}. \tag{10}$$

Thus, the machine capacity constraints (3) can now simply be written as

$$\sum_{j\in J} y_{ijs} \le 1 \quad \text{for all } i \in M, \quad s \in \mathbb{Z}_{\ge 0}. \tag{11}$$

Moreover, making use of the first part of Lemma 2, constraints (2) translate to

$$\sum_{i\in M}\sum_{s\in\mathbb{Z}_{\ge 0}} \frac{y_{ijs}}{\mathbb{E}[P_{ij}]} = 1 \quad \text{for all } j \in J. \tag{12}$$

Finally, with the help of the second part of Lemma 2, the expected completion time (4) of job j can be rewritten as

$$C_j^{\mathrm{LP}} = \sum_{i\in M}\sum_{s\in\mathbb{Z}_{\ge 0}} \left(\frac{y_{ijs}}{\mathbb{E}[P_{ij}]}\left(s+\tfrac{1}{2}\right) + \frac{1-\mathbb{CV}[P_{ij}]^2}{2} y_{ijs} \right) \quad \text{for all } j \in J. \tag{13}$$

Thus, we get the following new LP relaxation:

$$\begin{aligned} \min \quad & \sum_{j\in J} w_j C_j^{\mathrm{LP}} \\ \text{s.t.} \quad & (11), (12), (13) \\ & y_{ijs} \ge 0 \quad \text{for all } j\in J, \; i \in M, \; s \in \mathbb{Z}_{\ge 0}. \end{aligned}$$

As any feasible solution x to the LP relaxation from §3 can be mapped to a feasible solution y of the same value via (10), the new relaxation is not stronger than the old one. Moreover, there exist deterministic instances ($\Delta = 0$) on identical parallel machines for which the gap between the optimal solution values of the two relaxations is arbitrarily close to 2. For example, consider one job j with weight $w_j = 1$, processing time $p_j = m$, and m identical parallel machines. The optimal LP solution of the linear program in x-variables has value m, which is also the value of an optimal schedule. The optimal value of the linear program in y-variables, however, is $(m+1)/2$. The new LP can, however, be strengthened by adding the following constraints:

$$C_j^{\mathrm{LP}} \ge \sum_{i\in M}\sum_{s\in\mathbb{Z}_{\ge 0}} y_{ijs} \quad \text{for all } j \in J. \tag{14}$$

The algorithms presented in §§4 and 5 can be easily reinterpreted with respect to a solution y to the new LP relaxation and yield the same performance bounds.

We finally also remark that because of our insights about the LP relaxation in y-variables, the work of Skutella [30] on approximation algorithms based on convex quadratic programming relaxations can be generalized to the setting of stochastic unrelated machine scheduling.

Acknowledgments. 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" in 2013. The authors would like to thank the organizers and Schloss Dagstuhl for providing an enjoyable and stimulating atmosphere. The first author was supported in the framework of Matheon by the Einstein Foundation Berlin. 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 of the University of Twente. An extended abstract with parts of this paper appeared in the Proc. of the 31st Symposium on Theoretical Aspects of Computer Science, STACS 2014 (Skutella et al. [33]).

References

[1] Afrati F, Bampis E, Chekuri C, Karger D, Kenyon C, Khanna S, Milis I , et al. (1999) Approximation schemes for minimizing average weighted completion time with release dates. *Proc. 40th Ann. IEEE Sympos. Foundations Comput. Sci.* IEEE Computer Society, Washington, DC, 32–43.
[2] Chekuri C, Motwani R, Natarajan B, Stein C (2001) Approximation techniques for average completion time scheduling. *SIAM J. Comput.* 31:146–166.
[3] Chou CF, Liu H, Queyranne M, Simchi-Levi D (2006) On the asymptotic optimality of a simple on-line algorithm for the stochastic single machine weighted completion time problem and its extensions. *Oper. Res.* 54(3):464–474.
[4] Chudak FA (1999) A min-sum 3/2-approximation algorithm for scheduling unrelated parallel machines. *J. Scheduling* 2:73–77.
[5] Correa J, Wagner M (2008) LP-based online scheduling: From single to parallel machines. *Math. Programming* 119:109–136.
[6] Dean B, Goemans MX, Vondrák J (2008) Approximating the stochastic knapsack problem: The benefit of adaptivity. *Math. Oper. Res.* 33(4):945–964.
[7] Feller W (1971) *An Introduction to Probability Theory and Its Applications*, Vol. 2, 2nd ed., Wiley Series in Probability and Mathematical Statistics (John Wiley & Sons, New York).
[8] Graham RL, Lawler EL, Lenstra JK, Rinnooy Kan AHG (1979) Optimization and approximation in deterministic sequencing and scheduling: A survey. *Ann. Discrete Math.* 5:287–326.
[9] Gupta A, Krishnaswamy R, Molinaro M, Ravi R (2011) Approximation algorithms for correlated knapsacks and non-martingale bandits. *Proc. 52nd Ann. IEEE Sympos. Foundations Comput. Sci.* (IEEE Computer Society, Washington, DC), 827–836.
[10] Hall LA, Schulz AS, Shmoys DB, Wein J (1997) Scheduling to minimize average completion time: Off-line and on-line approximation algorithms. *Math. Oper. Res.* 22(3):513–544.
[11] Hoogeveen H, Schuurman P, Woeginger GJ (2001) Non-approximability results for scheduling problems with minsum criteria. *INFORMS J. Comput.* 13(2):157–168.
[12] Koutsoupias E, Papadimitriou CH (2000) Beyond competitive analysis. *SIAM J. Comput.* 30:300–317.
[13] Lenstra JK, Rinnooy Kan AHG, Brucker P (1977) Complexity of machine scheduling problems. *Ann. Discrete Math.* 1:343–362.
[14] Megow N (2013) Approximation in stochastic scheduling. Invited talk at the Dagstuhl Seminar 13111 "Scheduling," Schloss Dagstuhl, Wadern.
[15] Megow N, Vredeveld T (2014) A tight 2-approximation for preemptive stochastic scheduling. *Math. Oper. Res.* 39(4):1297–1310.
[16] Megow N, Uetz M, Vredeveld T (2006) Models and algorithms for stochastic online scheduling. *Math. Oper. Res.* 31(3):513–525.
[17] Möhring RH, Radermacher FJ, Weiss G (1984) Stochastic scheduling problems I: General strategies. *ZOR—Zeitschrift für Oper. Res.* 28:193–260.
[18] Möhring RH, Schulz AS, Uetz M (1999) Approximation in stochastic scheduling: The power of LP-based priority policies. *J. ACM* 46:924–942.
[19] Motwani R, Raghavan P (1995) *Randomized Algorithms* (Cambridge University Press, Cambridge, UK).
[20] Queyranne M (1993) Structure of a simple scheduling polyhedron. *Math. Programming* 58:263–285.
[21] Ross SH (1983) *Introduction to Stochastic Dynamic Programming*, Probability and Mathematical Statistics (Academic Press, New York).
[22] Rothkopf MH (1966) Scheduling with random service times. *Management Sci.* 12(9):707–713.
[23] Scharbrodt M, Schickinger T, Steger A (2006) A new average case analysis for completion time scheduling. *J. ACM* 53:121–146.
[24] Schulz AS (1996) Scheduling to minimize total weighted completion time: Performance guarantees of LP-based heuristics and lower bounds. Cunningham WH, McCormick ST, Queyranne M, eds. *Integer Programming and Combinatorial Optimization*, Lecture Notes in Computer Science, Vol. 1084 (Springer, Berlin), 301–315.
[25] Schulz AS (2008) Stochastic online scheduling revisited. Yang B, Du D-Z, Wang C, eds. *Combinatorial Optimization and Applications*, Lecture Notes in Computer Science, Vol. 5165 (Springer, Berlin), 448–457.
[26] Schulz AS, Skutella M (2002) Scheduling unrelated machines by randomized rounding. *SIAM J. Discrete Math.* 15:450–469.
[27] Schuurman P, Woeginger GJ (1999) Polynomial time approximation algorithms for machine scheduling: Ten open problems. *J. Scheduling* 2:203–213.
[28] Sethuraman J, Squillante MS (1999) Optimal scheduling of multiclass parallel machines. *Proc. 10th Ann. ACM–SIAM Sympos. Discrete Algorithms, SODA '99* (SIAM, Philadelphia), 963–964.
[29] Skutella M (1998) Approximation and randomization in scheduling. Ph.D. thesis, Technische Universität Berlin, Germany.
[30] Skutella M (2001) Convex quadratic and semidefinite programming relaxations in scheduling. *J. ACM* 48:206–242.
[31] Skutella M, Uetz M (2005) Stochastic machine scheduling with precedence constraints. *SIAM J. Comput.* 34:788–802.
[32] Skutella M, Woeginger GJ (2000) A PTAS for minimizing the total weighted completion time on identical parallel machines. *Math. Oper. Res.* 25(1):63–75.
[33] Skutella M, Sviridenko M, Uetz M (2014) Stochastic scheduling on unrelated machines. Portier N, Mayr EW, eds. *Proc. 31st Symp. Theoret. Aspects Comput. Sci., Leibniz Internat. Proc. Informatics*, Vol. 25 (Dagstuhl Publishing, Wadern, Germany), 639–650.

[34] Smith WE (1956) Various optimizers for single-stage production. *Naval Res. Logist. Quart.* 3:59–66.
[35] Souza A, Steger A (2006) The expected competitive ratio for weighted completion time scheduling. *Theory Comput. Systems* 39: 121–136.
[36] Uetz M (2003) When greediness fails: Examples from stochastic scheduling. *Oper. Res. Lett.* 31:413–419.
[37] Weiss G (1990) Approximation results in parallel machines stochastic scheduling. *Ann. Oper. Res.* 26:195–242.
[38] Weiss G (1992) Turnpike optimality of Smith's rule in parallel machines stochastic scheduling. *Math. Oper. Res.* 17(2):255–270.
[39] Wolsey LA (1985) Mixed integer programming formulations for production planning and scheduling problems. *Invited talk at the 12th Internat. Sympos. Math. Programming, MIT, Cambridge, MA*.

CORRECTION

In this article, "Unrelated Machine Scheduling with Stochastic Processing Times" by Martin Skutella, Maxim Sviridenko, and Marc Uetz (first published in *Articles in Advance*, February 16, 2016, *Mathematics of Operations Research*, DOI:10.1287/moor.2015.0757), an additional Section 9 was left in the text by mistake. It has been removed from the article.