Stochastic Online Scheduling with Precedence Constraints

Nicole Megow* Tjark Vredeveld †

January 5, 2009

Abstract

We consider the preemptive and non-preemptive problems of scheduling jobs with precedence constraints on parallel machines with the objective to minimize the sum of (weighted) completion times. We investigate an online model in which the scheduler learns about a job when all its predecessors have completed. For scheduling on a single machine, we show matching lower and upper bounds of $\Theta(n)$ and $\Theta(\sqrt{n})$ for jobs with general and equal weights, respectively. We also derive corresponding results on parallel machines.

Our result for arbitrary job weights holds even in the more general stochastic online scheduling model where, in addition to the limited information about the job set, processing times are uncertain. For a large class of processing time distributions, we derive also an improved performance guarantee if weights are equal.

1 Introduction

One of the classical scheduling problems that has attracted research for decades is the problem of processing jobs with precedence constraints on parallel machines with the objective to minimize the sum of (weighted) completion times. We consider a stochastic online version of this problem where processing times are modelled as random variables and the jobs become known to the scheduler online.

In traditional online paradigms, i. e., the online-time and the online-list model [21, 24], it is assumed that all data about a request are revealed as soon as the request becomes known. Interpreted for an online scheduling problem with precedence constraints, this means that whenever a job arrives, a scheduler learns about its weight and processing time and – most importantly – about job dependencies. However, these dependencies occur between *two* jobs and it is not clear which job gets assigned the information about such a bilateral relation. Certainly, there are various options to specify the information that should be revealed at job arrival. However, we consider a model in which the moment of unveiling jobs and all their data is designated by other *job completions*: a scheduler learns about the existence of a job when all its predecessors have completed their processing. Then, its weight, (expected) processing time and all precedence relations to predecessors become known. This model has been used earlier for online scheduling to minimize makespan [8, 2].

1.1 Problem definition

Let $\mathcal{J} = \{1, 2, \ldots, n\}$ be a set of jobs which must be scheduled on m identical, parallel machines. Each of the machines can process at most one job at the time, and the jobs can be executed by any of the machines. All jobs must be scheduled in compliance with the given precedence constraints.

*Max-Planck-Institut für Informatik, Campus E1 4, 66123 Saarbrücken, Germany. Email: `nmegow@mpi-inf.mpg.de`. Research partially supported by the DFG Research Center MATHEON *Mathematics for key technologies* in Berlin.

†Maastricht University, Department of Quantitative Economics, P.O. Box 616, 6200 MD Maastricht, The Netherlands. Email: `t.vredeveld@ke.unimaas.nl`. Research partially supported by METEOR, the Maastricht research school of Economics of Technology and Organizations.

These constraints define a partial order $(\mathcal{J}, \prec)$ on the set of jobs $\mathcal{J}$, where $j \prec k$ implies that job k must not start processing before j has completed. If no precedence constraints are given, then we call the jobs independent.

Each job j must be processed for P_j units of time, where P_j is a non-negative random variable. By $\mathbb{E}[P_j]$ we denote the expected value of the processing time of job j and by p_j a particular realization of P_j. We assume that all random variables of processing times are stochastically independent. We may or may not allow preemption. In the preemptive setting, a job can be interrupted at any time and resume processing on the same or another machine at any time later. In the non-preemptive setting, each job must run until its completion once it has started. Additionally, each job j has associated a non-negative weight w_j.

The goal is to find a non-anticipatory scheduling policy so as to minimize the total weighted completion time of the jobs, $\sum w_j C_j$, in expectation, where C_j denotes the completion time of job j. For details on stochastic scheduling policies we refer to Möhring, Radermacher, and Weiss [19].

In this paper we consider the online version of these stochastic problems in which a job j becomes known to the scheduler when all its predecessors $k \prec j$ have completed their processing; at this point in time the weight w_j and the probability distribution of the processing time P_j are revealed. The solution of such a stochastic online scheduling problem is a non-anticipatory, online scheduling policy; for more details see [17]. We aim for approximative policies, and as suggested in [17], we use a generalized definition of approximation guarantees from traditional stochastic offline scheduling by [20]. Then, an (online) stochastic policy Π is a ρ-approximation, for some $\rho \geq 1$, if for all problem instances $\mathcal{I}$,

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

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

1.2 Previous Work

The deterministic offline problem of scheduling jobs with precedence constraints to minimize the sum of (weighted) completion times has been shown to be $\mathcal{NP}$-hard [12, 13] even if there is only a single processor. The preemptive problem is $\mathcal{NP}$-hard already on two machines when precedence constraints form chains [5] and in the weighted setting even when additionally all jobs have unit processing times [5, 27].

The deterministic single machine problem has attracted research for more than thirty years and a vast amount of results has been obtained on this problem. Several classes of scheduling algorithms based on different LP-formulations are known that achieve an approximation ratio of 2 in polynomial time whereas special cases are even solvable optimally; we refer to [4, 1] for a recent comprehensive overview. For the parallel machine variant of this problem, the currently best known approximation algorithm is by Queyranne, and Schulz [22] and yields an approximation guarantee of $4 - 2/m$. When preemption is allowed, Hall et al. [9] propose an LP-based algorithm that yields a $3 - 1/m$-approximate solution.

Despite the obvious research interest in the scheduling problem under consideration, literature is very limited when assuming uncertainty in the problem data. As far as we know, the only work that deals with precedence constraints in scheduling under uncertainty is by Skutella and Uetz [25], who consider the stochastic offline scheduling model. The performance guarantees they prove are functions of a parameter Δ that bounds the squared coefficient of variation of processing times. Their policies require to solve a linear programming relaxation in which all jobs must be known in advance. This approach is not directly applicable in our online setting.

To the best of our knowledge, we present the first results on scheduling with precedence constraints to minimize the sum of completion times when the problem instance is revealed online.

On the other hand, research has been done on the deterministic online problem when the goal is to minimize the makespan. Probably, one of the earliest publications using the online paradigm introduced above is by Feldmann et al. [8]. They consider a different scheduling model, in which parallel jobs are processed by more than one machine at the same time. Considering jobs that are processed by at most one machine at the time, no algorithm can achieve a constant competitive ratio. Azar and Epstein [2] derived a lower bound of $\Omega(\sqrt{m})$ for the competitive ratio of any deterministic or random online algorithm that schedules jobs, preemptively or not, on m related machines. This bound matches an upper bound given earlier by Jaffe [11].

Finally, there exists relevant work on the deterministic offline problem of scheduling jobs with generalized precedence constraints, so called AND/OR-precedence relations. While ordinary precedence constraints force a job to wait for the completion of *all* its predecessors (represented as an AND-node in the corresponding precedence graph), there is an additional relaxed waiting condition that allows a job to start after at least one of its predecessors has completed (OR-node). Clearly, ordinary precedence constraints are contained as a special case in AND/OR-precedence constraints. Erlebach, Kääb, and Möhring [7] analyze the performance of a *Shortest Processing Time* (SPT) Algorithm for the deterministic offline problem of scheduling with AND/OR-precedence constraints on a single machine. The classic SPT algorithm schedules jobs in non-decreasing order of their processing times. Erlebach et al.'s variant considers at any time only jobs that are *available* for processing according to the precedence constraints and schedules one with minimal processing time. Thus, it coincides with the online SPT algorithm that knows of jobs only after all predecessors have finished. Therefore, the approximation results translate into competitiveness results in our online setting if all processing times are deterministic.

Theorem 1 (Erlebach, Kääb, and Möhring [7]). *The online version of the* SPT *algorithm has a competitive ratio of n for the deterministic, non-preemptive problem on a single machine. If all jobs have equal weights, then* SPT *is $2\sqrt{n}$-competitive.*

In their paper, Erlebach et al. [7] state (without proof) that a parallel machine version of SPT also yields an approximation guarantee of n for the deterministic parallel machine scheduling problem. This result would also hold in our model restricted to deterministic instances.

1.3 Our results

We provide the first results on online scheduling with precedence constraints to minimize the (weighted) sum of completion times.

We complement the results in Theorem 1 by Erlebach et al. [7] with matching lower bounds on the competitive ratio of any online algorithm for the single machine problem. It follows that an online SPT algorithm achieves the best possible performance for this problem which is a competitive ratio of n. If all weights are equal, then we improve this bound to $\sqrt{2n}$. On the other hand, we show that no online algorithm can have a competitive ratio less than $2/3\sqrt{n}-1$. For the corresponding scheduling problem on identical parallel machines we also provide lower and upper bounds on the competitive ratio. Here, we leave a gap growing with the number of machines m. Table 1 gives a summary of the lower and upper bounds we derived.

Notice, that on a single machine preemption does not lead to an improved schedule since online information is revealed only at the completion of jobs. Therefore, the results transfer immediately to the preemptive setting. This is also true for the bounds on parallel machine scheduling. The worst-case instances for the lower bounds are constructed on chains in which case preemption is redundant again [5] whereas in the analysis of the algorithm (for the upper bound) we use lower bounds on the optimal offline value which hold in the preemptive as well as in the non-preemptive setting.

We derive most of the performance guarantees above in a much more general model for scheduling under uncertainty, in which additionally to the lack of information about job arrivals, also processing times of known jobs are uncertain. We show that an online variant of the *Shortest Expected Processing Time* (SEPT) policy yields the best possible approximation guarantees, n, for the stochastic single machine problem, independently of the probability distribution of processing times. If the weights of all jobs are equal, then we improve this bound for processing time distributions with $\mathrm{Var}[P_j]/\mathbb{E}\left[\,P_j\,\right]^2 \leq n/(m-1)-1$. For so called NBUE distributions, that is,

	lower bound	upper bound	And/Or-prec [7]
single machine, $w_j = 1$	$\frac{2}{3}\sqrt{n} - 1$	$\sqrt{2n}$	$2\sqrt{n}$
single machine, arbitrary w_j	$n - 1$	n	n
parallel machines, $w_j = 1$	$\frac{2}{3}\sqrt{\frac{n}{m}} - 1$	$\sqrt{2mn}$	n
parallel machines, arbitrary w_j	$\frac{n-1}{m}$	n	n

Table 1: Bounds on the performance guarantee of any preemptive or non-preemptive online algorithm [lower bound] and on the performance guarantee of an online version of S(e)pt [upper bound] for non-preemptive stochastic online scheduling of jobs with precedence constraints. Upper bounds for the unweighted setting are given for the special case where processing time distributions obey $\mathrm{Var}\,[P_j] \le \mathbb{E}\,[\,P_j\,]^2$. The approximation results for arbitrary weights also hold true when job preemption is allowed. For deterministic problem instances this is true also for the improved results for trivial weights. Offline considerations in [7] for problems with And/Or-precedence relations transfer to our deterministic online setting and inspire our new results; their approximation guarantees are given in the third column.

distributions for which holds that $\mathbb{E}\,[\,P_j - t \,|\, P_j > t\,] \le \mathbb{E}\,[\,P_j\,]$, for all $t \ge 0$, and thus satisfying $\mathrm{Var}[P_j] \le \mathbb{E}\,[\,P_j\,]^2$ [10], the competitive ratio is no more than $\sqrt{2mn}$. Obviously, this holds also for deterministic processing times in which case the Sept policy coincides with the Spt algorithm.

Finally, the lower bounds on the competitive ratio for deterministic online scheduling translate directly into lower bounds on our more general model, since online scheduling instances with deterministic processing times can be seen as a special case.

In stochastic (online) scheduling preemption is a powerful tool for dealing with the uncertainty of processing times even on a single machine; see e.g. [28, 18]. However, the approximation guarantee of n for scheduling jobs with arbitrary weights by Sept holds also in the preemptive scheduling setting. This is not true for the improved result for jobs with equal job weights because we use a lower bound on the optimum value which does not hold when preemption is allowed. Actually, simple examples show that Sept as well as any other policy that does not utilize preemption may perform arbitrarily bad.

Our results are still valid when considering the more general And/Or-precedence constraints even though this is not focus of our work. Thereby we give a full proof for the approximation guarantee of n for parallel machine scheduling with And/Or-precedence constraints which was mentioned in [7]. Moreover, we improve this result for scheduling instances in which all jobs have equal weights and give a approximation guarantee of $\sqrt{2mn}$.

2 Scheduling jobs with arbitrary weights

For scheduling independent jobs, good performance guarantees have been obtained for online versions of *Smith's* classic rule [26], also known as *Weighted Shortest Processing Time* Rule (Wspt), its stochastic counterpart, and various extensions [23, 16, 17, 15]. If jobs must obey precedence relations which are revealed after all predecessors have completed, no such variant yields a bounded performance. We give a simple single machine example where all jobs have even deterministic processing times. Note, that on a single machine no waiting time will reveal new information on the online sequence and no preemption will improve the schedule.

Example 1. Consider an instance that consists of the following three jobs. The first job has processing time $p_1 = k \ge 3$ and weight $w_1 = 1$. Jobs 2 and 3 must obey the precedence constraint $2 \prec 3$; they have processing times $p_2 = 1$ and $p_3 = \varepsilon$, respectively, and their weights are $w_2 = \varepsilon$ and $w_3 = k$. Let k and ε be such that the ratios of weight over processing time of the two independent jobs 1 and 2 fulfill $w_1/p_1 > w_2/p_2$, that means $\varepsilon < 1/k$.

Then the online version of the Wspt algorithm schedules the jobs in increasing order of their indices, 1, 2, and 3 achieving an objective value of $k^2 + 2k + \varepsilon(2k + 1)$. In contrast, an optimal schedule has job 2 being processed first, followed by 3 and 1, and yields thus a value of $2k + 1 + \varepsilon(k + 2)$. For $k \geq 3$, the ratio of values of the WSPT schedule and an optimal schedule is larger than $\frac{k}{2}$ which is unbounded for increasing k.

Consider the online variant of the *Shortest Processing Time* (SPT) policy that schedules at any point in time the job with the shortest processing time among the available jobs. Even though it seems counter intuitive to ignore known job weights, Theorem 1 states that this algorithm yields a competitive ratio that matches the lower bound on the performance guarantee for any online algorithm on a single machine as we prove later in Theorems 5 and 6.

We extend this result to the more general parallel machine setting in which all jobs have stochastic processing times, without loosing in the performance guarantee. Consider the stochastic online preemptive and non-preemptive scheduling problems on parallel machines and the stochastic online policy that runs the online variant of the *Shortest Expected Processing Time* (SEPT) policy on only *one out of* m machines. This non-preemptive policy simply ignores the $m - 1$ remaining available machines. We denote this policy by 1-SEPT.

Lemma 2. *The order of jobs in a schedule obtained by 1-*SEPT *is independent of the realization of processing times.*

Proof. We claim that for any two realizations of processing times, at the completion of job j the same set of jobs is available, for any $j \in \mathcal{J}$. This implies the lemma, as 1-SEPT chooses the job to process only based on the set of available jobs and the expected processing times of these jobs.

To see the claim, consider two realizations of processing times. We assume that jobs are indexed in order in which they are processed in the first realization. First note that when no job has been processed, obviously the same set of jobs is available to 1-SEPT in both realizations. Suppose the claim is true up to job j. As 1-SEPT chooses job $j + 1$ to be processed after job j in the first realization and in the second realization the same set of jobs is available to 1-SEPT, the policy will also choose job $j + 1$ to be processed for the second realization. Hence, at the completion of job $j + 1$ the same set of jobs will be set free to 1-SEPT in the first as in the second realization. $\square$

Let $\mathbb{E}\left[\, C_j^{\Pi} \,\right]$ denote the expected completion time of a job j in the schedule obtained by policy Π. Inspired by [7], we define a stochastic version of the *threshold* ξ_j^{Π} of a job j for policy Π as the maximum expected processing time of a job that finishes in expectation no later than j. More formally,

$$\xi_j^{\Pi} \;=\; \max_{k \in \mathcal{J}} \{\, \mathbb{E}\,[\, P_k \,] \mid \mathbb{E}\left[\, C_k^{\Pi} \,\right] \leq\; \mathbb{E}\left[\, C_j^{\Pi} \,\right] \} \,.$$

Thresholds have a useful property.

Lemma 3 (Threshold-Lemma)**.** *Let* Π *be a feasible policy for the stochastic preemptive or non-preemptive scheduling problem on parallel machines. Then for any job* $j \in \mathcal{J}$ *with threshold* ξ_j^{Π} *holds*

$$\xi_j^{1-\text{SEPT}} \;\leq\; \xi_j^{\Pi} \,.$$

Proof. For any non-anticipatory policy Π, we have that $\xi_j^{\Pi} \geq \mathbb{E}\,[\, P_j \,]$ since $\mathbb{E}\,[\, P_j \,] \leq \mathbb{E}\left[\, C_j^{\Pi} \,\right]$. If $\xi_j^{\Pi} = \mathbb{E}\,[\, P_j \,]$, then the lemma holds. Suppose that $\xi_j^{1-\text{SEPT}} > \mathbb{E}\,[\, P_j \,]$. Then there exists a job k that was completed before job j by 1-SEPT, and that has expected processing time $\mathbb{E}\,[\, P_k \,] = \xi_j^{1-\text{SEPT}} > \mathbb{E}\,[\, P_j \,]$. As 1-SEPT chooses the job with smallest expected processing time, we know by Lemma 2 that in any realization of processing times, job j cannot be available to 1-SEPT when job k is started to be processed. Hence, k must be a predecessor of j. Thus, any policy processes job k before j, from which follows $\xi_j^{\Pi} \geq \mathbb{E}\,[\, P_k \,] = \xi_j^{1-\text{SEPT}}$. $\square$

Now, we can establish a performance guarantee for 1-SEPT.

Theorem 4. *The* 1-SEPT *policy that utilizes only one machine is an* n*-approximation for the stochastic online scheduling problem on parallel machines with and without preemption.*

Proof. Let jobs be indexed in their order in the 1-Sept schedule. Recall from Lemma 2 that the order of jobs in the 1-Sept schedule is independent of the realization of processing times. Then $k < j$ implies $\mathbb{E}\left[C_k^{1-\textsc{Sept}}\right] < \mathbb{E}\left[C_j^{1-\textsc{Sept}}\right]$. Since there is no idle time, the expected completion time $\mathbb{E}\left[C_j^{1-\textsc{Sept}}\right]$ of a job j in the 1-Sept schedule is

$$\mathbb{E}\left[C_j^{1-\textsc{Sept}}\right] = \sum_{k=1}^{j} \mathbb{E}\left[P_k\right] \leq \; n\,\xi_j^{1-\textsc{Sept}}\,. \tag{1}$$

With the Threshold-Lemma 3 and the fact that $\xi_j^{\Pi} \leq \mathbb{E}\left[C_j^{\Pi}\right]$ holds by definition for any policy Π – thus, also for an optimal policy Opt – we conclude from inequality (1)

$$\mathbb{E}\left[C_j^{\textsc{Sept}}\right] \;\leq\; n\,\xi_j^{\textsc{Opt}} \;\leq\; n\,\mathbb{E}\left[C_j^{\textsc{Opt}}\right]\,.$$

Weighted summation over all jobs $j \in \mathcal{J}$ proves the theorem. ◻

In the following we show that no online algorithm using m machines can have a competitive ratio of less than $(n-1)/m$. Thus the analysis of the simple 1-Sept policy using one machine is tight if there is just one machine available, whereas in general it leaves a gap in the order of m. The lower bound is achieved even when the processing times are given deterministically. By definition of the model, these bounds carry over to the more general stochastic online scheduling model.

Theorem 5. *No preemptive or non-preemptive deterministic online algorithm can achieve a competitive ratio less than $(n-1)/m$ for scheduling with precedence constraints on any number of machines m.*

Proof. Consider the following instance that consists of n jobs and assume, w.l.o.g., that $n-1$ is a multiple of the number of machines m. We have $n-1$ independent jobs $1, 2, \ldots, n-1$ with weights $w_j = 0$ and unit processing time. Suppose, that the online algorithm chooses the job ℓ to be scheduled such that it completes as the last job. Then, we have one final job n in the instance with ℓ as its predecessor and with processing time zero and weight 1.

Clearly, the online algorithm can schedule the highly weighted last job only as the final job, achieving a schedule with value $(n-1)/m$. In contrast, an offline algorithm would choose job ℓ as one of the m first jobs to be processed, followed by the highly weighted job n. This yields value of 1. Thus, the ratio between both value is $(n-1)/m$. ◻

Adding a randomizing ingredient to the instance above, we extend the result to a lower bound for any randomized online algorithm. Here, we make use of Yao's principle [29] which states that a lower bound on the expected competitive ratio of any deterministic online algorithm on an appropriate input distribution also lower bounds the competitive ratio of any randomized online algorithm against an oblivious adversary. Since this is the weakest type of adversaries, these lower bounds hold against any other adversary with more power as well; see, e.g., Ben-David et al. [3].

Theorem 6. *No preemptive or non-preemptive randomized online algorithm can achieve a competitive ratio less than $(n-1)/(m+1)$ for scheduling with precedence constraints on any number of machines m.*

Proof. Consider the instance in the previous proof. When playing against a randomized algorithm, an oblivious adversary does not know which will be the last completing job ℓ among the independent jobs. Therefore, we modify the instance by adding m random precedence relations to each job $1, \ldots, n-1$ with the same probability. Let $k := (n-1)/m$ be an integral number. Then, any set of m jobs (excluding n) has probability $1/\binom{km}{m}$ for being the set of predecessors of n.

Clearly, the optimal offline solution has still value 1. Now, consider any deterministic online algorithm Alg.

$$\mathbb{E}[\,\textsc{Alg}\,] = \mathbb{E}\left[\,C_n^{\textsc{Alg}}\,\right] \geq \sum_{i=1}^{k-1} i \Pr\left[i \leq C_n^{\textsc{Alg}} < i+1\right] + k \Pr\left[k \leq C_n^{\textsc{Alg}}\right] = \sum_{k=i}^{k} \Pr\left[C_n^{\textsc{Alg}} \geq i\right]$$
$$= \sum_{i=1}^{k} 1 - \Pr\left[C_n^{\textsc{Alg}} < i\right] = k - \sum_{i=1}^{k} \Pr\left[C_n^{\textsc{Alg}} < i\right]. \tag{2}$$

Moreover, we give Alg the advantage that it schedules job n as soon as all its predecessors have completed. The number of jobs that can be completed strictly before a fixed (integral) point in time $i+1$ is at most im. Therefore, the probability that all m random predecessors of job n complete before a fixed time $2 \leq i+1 \leq k$ is

$$\Pr\left[C_n^{\textsc{Alg}} < i+1\right] \;\leq\; \frac{\binom{im}{m}}{\binom{km}{m}} \;=\; \frac{\frac{(im)!}{((i-1)m)!}}{\frac{(km)!}{((k-1)m)!}} \;\leq\; \left(\frac{i}{k}\right)^m, \tag{3}$$

and $\Pr\left[C_n^{\textsc{Alg}} < 1\right] = 0$. Now we use the bound $\sum_{i=1}^{k-1} \left(\frac{i}{k}\right)^m \leq \frac{k}{m+1}$ which we prove below. This bound combined with (2) and (3) yields

$$\mathbb{E}[\,\textsc{Alg}\,] \;\geq\; k - \sum_{i=1}^{k-1} \left(\frac{i}{k}\right)^m \;\geq\; k\left(1 - \frac{1}{m+1}\right) = \frac{n-1}{m+1}.$$

By Yao's principle [29] this gives the desired lower bound on the competitive ratio of any online algorithm.

It is left to show $\sum_{i=1}^{k-1} \left(\frac{i}{k}\right)^m \leq \frac{k}{m+1}$. We prove the bound by induction on k; it is certainly true for $k = 2$.

$$\sum_{i=1}^{k} \left(\frac{i}{k+1}\right)^m \;=\; \sum_{i=1}^{k-1} \left(\frac{i}{k}\right)^m \left(\frac{k}{k+1}\right)^m \;+\; \left(\frac{k}{k+1}\right)^m \;=\; \left(\frac{k}{k+1}\right)^m \left[\,\sum_{i=1}^{k-1} \left(\frac{i}{k}\right)^m + 1\,\right]$$
$$\leq\; \left(\frac{k}{k+1}\right)^m \left[\,\frac{k}{m+1} + 1\,\right] \;=\; \frac{k^{m+1} + (m+1)k^m}{(k+1)^m(m+1)} \tag{4}$$
$$\leq\; \frac{(k+1)^{m+1}}{(k+1)^m(m+1)} \;=\; \frac{k+1}{m+1}. \tag{5}$$

Inequality (4) follows from the induction hypothesis. The second inequality (5) follows from

$$(k+1)^n \;=\; \sum_{i=0}^{n} \binom{n}{i} k^i \;>\; k^n + n\,k^{n-1},$$

which concludes the inductive step. $\square$

2.1 Scheduling Jobs with equal Weights

If all job weights are equal then, intuitively, the 1-Sept policy should perform better than in the general setting. We show a performance bound for non-preemptive scheduling which improves the n-approximation in Theorem 4 for a large class of problem instances; in particular, for instances with NBUE distributed processing times, which include deterministic instances, it is $\sqrt{2nm}$. We achieve this bound by extending ideas of Erlebach et al. [7] to the stochastic online parallel-machine setting. Moreover, we apply the following lower bound on the expected optimal value for the relaxed problem without precedence constraints given by Möhring, Schulz, and Uetz [20].

Lemma 7 (Möhring et al. [20]). *Consider the stochastic scheduling problem on parallel machines to minimize the expected total weighted completion time in which all jobs are available for processing from the beginning. Assume that the jobs are indexed in non-decreasing order of expected processing times* $\mathbb{E}[P_j]$*. Then, an optimal policy* Opt *yields a value*

$$\sum_j \mathbb{E}\left[C_j^{\text{OPT}}\right] \;\geq\; \sum_j \sum_{k=1}^{j} \frac{\mathbb{E}[P_k]}{m} \;-\; \frac{(m-1)(\Delta-1)}{2m} \sum_j \mathbb{E}[P_j]\,,$$

where Δ *bounds the squared coefficient of variation of the processing times, that is,* $\mathrm{Var}[P_j] \leq \Delta\,\mathbb{E}[P_j]^2$ *for all jobs* $j = 1,\ldots,n$ *and some* $\Delta \geq 0$.

Theorem 8. *The 1-*Sept *policy achieves an approximation guarantee of*

$$\rho \;=\; \frac{1}{2}(m-1)(\Delta-1) \;+\; \frac{1}{2}\sqrt{[(m-1)(\Delta-1)]^2 \,+\, 8mn}\,,$$

with $\mathrm{Var}[P_j] \leq \Delta\,\mathbb{E}[P_j]^2$ *for any instance of the non-preemptive stochastic online problem on parallel machines.*

Proof. Consider an 1-Sept schedule. Let $\alpha > (m-1)(\Delta-1)/\sqrt{n}$ be a parameter that will be specified later. For notational convenience, we define for each job j the set of jobs completed no later than job j in the 1-Septschedule as $\mathcal{B}(j) = \{k \in \mathcal{J} \mid \mathbb{E}\left[C_k^{1-\text{SEPT}}\right] \leq \mathbb{E}\left[C_j^{1-\text{SEPT}}\right]\}$. Let x be the last job in the 1-Sept schedule such that all jobs scheduled before this job have an expected processing time of at most $\mathbb{E}\left[C_x^{1-\text{SEPT}}\right]/(\alpha\sqrt{n})$, that is,

$$x \;:=\; \arg\max\left\{\mathbb{E}\left[C_j^{1-\text{SEPT}}\right] \;\middle|\; j \in \mathcal{J} \text{ and } \mathbb{E}[P_k] \leq \frac{\mathbb{E}\left[C_j^{1-\text{SEPT}}\right]}{\alpha\sqrt{n}} \text{ for all } k \in \mathcal{B}(j)\right\}.$$

This designated job x is used to partition the set of jobs into two disjunctive subsets: $\mathcal{J}^{\leq}$ denotes the set of jobs that complete before x in the 1-Sept schedule, that is, $\mathcal{J}^{\leq} = \{j \in \mathcal{J} \,|\, \mathbb{E}\left[C_j^{1-\text{SEPT}}\right] \leq \mathbb{E}\left[C_x^{1-\text{SEPT}}\right]\}$, and $\mathcal{J}^{>}$ consists of the remaining jobs $\mathcal{J} \setminus \mathcal{J}^{\leq}$. Obviously, the expected completion time of job x is $\mathbb{E}\left[C_x^{1-\text{SEPT}}\right] = \sum_{j\in\mathcal{J}^{\leq}} \mathbb{E}[P_j]$. Now, the expected value of the Sept schedule can be expressed as

$$\sum_{j\in\mathcal{J}} \mathbb{E}\left[C_j^{1-\text{SEPT}}\right] \;=\; \sum_{j\in\mathcal{J}^{\leq}} \mathbb{E}\left[C_j^{1-\text{SEPT}}\right] \;+\; \sum_{j\in\mathcal{J}^{>}} \mathbb{E}\left[C_j^{1-\text{SEPT}}\right].$$

We bound the expected completion times of jobs of both job sets separately. To bound the contribution of the jobs in $\mathcal{J}^{\leq}$, assume that $\mathcal{J}^{\leq} \neq \emptyset$.

Let Opt be an optimal policy for all jobs $j \in \mathcal{J}$ and Opt' an optimal policy that schedules only the jobs in $\mathcal{J}^{\leq}$. Clearly,

$$\sum_{j\in\mathcal{J}^{\leq}} \mathbb{E}\left[C_j^{\text{OPT}}\right] \;\geq\; \sum_{j\in\mathcal{J}^{\leq}} \mathbb{E}\left[C_j^{\text{OPT}'}\right]. \tag{6}$$

By ignoring the release dates, we can use Lemma 7. Assuming that the jobs in $\mathcal{J}^{\leq}$ are indexed, $1,\ldots,|\mathcal{J}^{\leq}|$, in non-decreasing order of their expected processing times, we obtain:

$$\sum_{j\in\mathcal{J}^{\leq}} \mathbb{E}\left[C_j^{\text{OPT}'}\right] \;\;\geq\;\; \sum_{j\in\mathcal{J}^{\leq}} \sum_{k=1}^{j} \frac{\mathbb{E}[P_k]}{m} \;-\; \frac{(m-1)(\Delta-1)}{2m} \sum_{j\in\mathcal{J}^{\leq}} \mathbb{E}[P_j]\,, \tag{7}$$

where $\mathrm{Var}[P_j] \leq \Delta\,\mathbb{E}[P_j]^2$ for all jobs $j \in \mathcal{J}$ and some $\Delta \geq 0$. We claim that $\sum_{j\in\mathcal{J}^{\leq}} \sum_{k=1}^{j} \mathbb{E}[P_k]$ is bounded from below by

$$\frac{\alpha\sqrt{n}}{2}\,\mathbb{E}\left[C_x^{1-\text{SEPT}}\right].$$

To see this claim, note that

$$\sum_{j\in\mathcal{J}^{\leq}}\sum_{k=1}^{j}\mathbb{E}\left[\,P_k\,\right] = \sum_{k\in\mathcal{J}^{\leq}}\left(|\mathcal{J}^{\leq}| - k + 1\right)\mathbb{E}\left[\,P_k\,\right].$$

This value can not be less than the minimum of this value over all possible expected processing times for jobs in $j \in \mathcal{J}^{\leq}$ satisfying $\sum_{j\in\mathcal{J}^{\leq}}\mathbb{E}\left[\,P_j\,\right] = \mathbb{E}\left[\,C_x^{1-\textsc{Sept}}\,\right]$ and $\mathbb{E}\left[\,P_j\,\right] \leq \mathbb{E}\left[\,C_x^{1-\textsc{Sept}}\,\right]/(\alpha\sqrt{n})$. This minimum is obviously obtained by setting $\mathbb{E}\left[\,P_j\,\right] = \mathbb{E}\left[\,C_x^{1-\textsc{Sept}}\,\right]/(\alpha\sqrt{n})$ for $j = |\mathcal{J}^{\leq}| - b + 1, \ldots, |\mathcal{J}^{\leq}|$, $\mathbb{E}\left[\,P_j\,\right] = (1 - b/(\alpha\sqrt{n}))\mathbb{E}\left[\,C_x^{1-\textsc{Sept}}\,\right]$, for $j = |\mathcal{J}^{\leq}| - b$, and $\mathbb{E}\left[\,P_j\,\right] = 0$ for all other j, where $b = \lfloor\alpha\sqrt{n}\rfloor$. This proves the claim and with $\sum_{j\in\mathcal{J}^{\leq}}\mathbb{E}\left[\,P_j\,\right] = \mathbb{E}\left[\,C_x^{1-\textsc{Sept}}\,\right]$ we have

$$\begin{aligned}\sum_{j\in\mathcal{J}^{\leq}}\mathbb{E}\left[\,C_j^{\textsc{Opt}}\,\right] \quad &\geq \quad \frac{\alpha\sqrt{n}}{2m}\mathbb{E}\left[\,C_x^{1-\textsc{Sept}}\,\right] \;-\; \frac{(m-1)(\Delta-1)}{2m}\mathbb{E}\left[\,C_x^{1-\textsc{Sept}}\,\right]\\ &= \quad \frac{\alpha\sqrt{n}-(m-1)(\Delta-1)}{2m}\mathbb{E}\left[\,C_x^{1-\textsc{Sept}}\,\right].\end{aligned}$$

With this estimate of the relevant portion of the expected optimal value we can bound the value achieved by the 1-Sept policy

$$\sum_{j\in\mathcal{J}^{\leq}}\mathbb{E}\left[\,C_j^{1-\textsc{Sept}}\,\right] \;\leq\; n\,\mathbb{E}\left[\,C_x^{1-\textsc{Sept}}\,\right] \;\leq\; \frac{2\,m\,n}{\alpha\sqrt{n}-(m-1)(\Delta-1)}\sum_{j\in\mathcal{J}^{\leq}}\mathbb{E}\left[\,C_j^{\textsc{Opt}}\,\right], \tag{8}$$

if and only if $\alpha\sqrt{n} > (m-1)(\Delta-1)$.

Consider now jobs in the remaining job set $\mathcal{J}^{>}$; by definition, there exists for each job $j \in \mathcal{J}^{>}$ a job k that completes in 1-Sept earlier than j and has processing time $\mathbb{E}\left[\,P_k\,\right] > \mathbb{E}\left[\,C_j^{1-\textsc{Sept}}\,\right]/(\alpha\sqrt{n})$. We conclude from this fact and the Threshold-Lemma 3 (including the notion of the threshold ξ_j^S) that for all $j \in \mathcal{J}^{>}$ holds

$$\mathbb{E}\left[\,C_j^{\textsc{Opt}}\,\right] \;\geq\; \xi_j^{\textsc{Opt}} \;\geq\; \xi_j^{1-\textsc{Sept}} \;\geq\; \mathbb{E}\left[\,P_k\,\right] \;>\; \frac{\mathbb{E}\left[\,C_j^{1-\textsc{Sept}}\,\right]}{\alpha\sqrt{n}}.$$

Summation over all jobs $j \in \mathcal{J}^{>}$ yields a bound on the completion times in the 1-Sept schedule,

$$\sum_{j\in\mathcal{J}^{>}}\mathbb{E}\left[\,C_j^{1-\textsc{Sept}}\,\right] \;\leq\; \alpha\sqrt{n}\sum_{j\in\mathcal{J}^{>}}\mathbb{E}\left[\,C_j^{\textsc{Opt}}\,\right].$$

Finally, combination with Equality (8) yields the bound

$$\sum_{j\in\mathcal{J}}\mathbb{E}\left[\,C_j^{1-\textsc{Sept}}\,\right] \;\leq\; \max\left\{\frac{2\,m\,n}{\alpha\sqrt{n}-(m-1)(\Delta-1)},\; \alpha\sqrt{n}\right\}\sum_{j\in\mathcal{J}}\mathbb{E}\left[\,C_j^{\textsc{Opt}}\,\right].$$

The performance bound is minimized when choosing the parameter $\alpha := ((m-1)(\Delta-1) + \sqrt{[(m-1)(\Delta-1)]^2 + 8mn})/(2\sqrt{n})$ which gives the desired approximation guarantee

$$\rho = \frac{1}{2}\,(m-1)(\Delta-1) + \frac{1}{2}\,\sqrt{[(m-1)(\Delta-1)]^2 + 8mn}\,.$$

Observe that the optimal choice of α fulfills the condition $\alpha\sqrt{n} > (m-1)(\Delta-1)$ in equality (8). $\square$

In contrast to the previous, more general approximation guarantee of value n in Theorem 4, this result depends on the variance of processing times. In particular, the performance guarantee ρ grows with the parameter Δ. However, for instances with distributions of small relative variance, this bound improves on the n-approximation for the general weighted problem in Theorem 4. More precisely, for instances with an upper bound on the squared coefficient of processing times

$$\Delta \;\leq\; \frac{n}{m-1} - 1$$

the performance guarantee ρ in Theorem 8 is at most n. Moreover, this theorem leads immediately to the following result for a restricted class of probability distributions – the NBUE distributions, which imply $\Delta \leq 1$ [10].

Corollary 9. *If all jobs have processing times that follow a NBUE distribution, that is, $\Delta \leq 1$, the 1-*Sept *policy that utilizes only one machine is a $\sqrt{2mn}$-approximation for the non-preemptive stochastic online scheduling problem on parallel machines with equal job weights. This includes deterministic instances.*

It follows from the analysis that the result holds also if And/Or-precedence constraints are present. Thus, we improve the approximation factor of n for the offline scheduling problem where jobs have equal weights and processing times are deterministic in [7] even though we consider a more general model.

Corollary 10. *The 1-*Spt *algorithm that utilizes only one machine is a $\sqrt{2mn}$-approximation for the scheduling problem on parallel machines with* And/Or*-precedence constraints with equal job weights.*

For preemptive scheduling of jobs with trivial weights, the analysis of 1-Sept above does not improve the general n-approximation of Theorem 4. The reason is, that the lower bound on the expected optimum value in Lemma 7 does not hold when jobs are allowed to be preempted. In fact, the Sept policy which does not preempt a job performs arbitrarily bad in the preemptive stochastic scheduling environment as simple examples show. However, for deterministic instances without precedence constraints preemption is redundant; see McNaughton [14]. Therefore, the deterministic version of the lower bound on the optimum value in Lemma 7 holds also for preemptive scheduling. In fact, in that case it coincides with the classical result by Eastman, Even, and Isaacs [6]. Now, the analysis of 1-Spt as in the proof of Theorem 8 holds true also for deterministic online scheduling with precedence constraints. Choosing $\alpha = \sqrt{2mn}$ leads to the corollary.

Corollary 11. *The 1-*Spt *algorithm is $\sqrt{2mn}$-competitive for the deterministic preemptive online scheduling problem on parallel machines, with the equal job weights*

Finally, we complement the new improved performance guarantees by a lower bound which leaves a gap in the order of m.

Theorem 12. *The competitive ratio of any preemptive and non-preemptive, deterministic online algorithm for scheduling jobs with precedence constraints has a lower bound of $\frac{2}{3}\sqrt{n/m} - 1/3$ and any randomized online algorithm has a lower bound of $\frac{2}{3}\sqrt{nm/(m+1)^2} - 1/3$ on the competitive ratio.*

Proof. We have mk independent jobs with processing times $p_j = 1$ for all $j = 1, \ldots, mk$ where $k >> m$. Moreover, there are $mk^2 - mk$ jobs that have length 0 and which must obey precedence constraints that form one long chain $mk+1 \prec mk+2 \prec \ldots$. Let $\ell \in \{1, \ldots, mk\}$ be the job to be scheduled last by the online algorithm among the independent jobs. This job ℓ is a predecessor of $mk+1$, the first job of the chain. Note, that an online algorithm Alg cannot start the job chain with $mk^2 - mk$ jobs earlier than time k. Alg yields a schedule of value

$$\textsc{Alg} \quad \geq \quad m\sum_{i=1}^{k} i + (mk^2 - mk)k \;=\; \frac{1}{2}\,mk\,(1 + (2k-1)\,k) \;\geq\; \frac{1}{2}\,mk^2(2k-1)\,. \tag{9}$$

In contrast, an optimal offline algorithm Opt knows the sequence in advance. By processing job ℓ at time 0 and starting the chain of zero length jobs at time 1, it can achieve an objective value

$$\text{Opt} \quad \leq \quad m\sum_{i=1}^{k} i + mk^2 - mk \;=\; \frac{1}{2}\,mk\,(3k-1) \;\leq\; \frac{3}{2}\,mk^2\,. \tag{10}$$

The ratio of the bounds in (9) and (10) combined with the number of jobs, $n = mk^2$, gives the lower bound on the competitive ratio of any deterministic online algorithm.

$$\frac{\text{Alg}}{\text{Opt}} \;\geq\; \frac{2k-1}{3} \;\geq\; \frac{2}{3}\sqrt{\frac{n}{m}} - \frac{1}{3}\,.$$

We achieve almost the same bound for randomized online algorithms by randomizing the deterministic instance in the same way as in the proof of Theorem 6. We consider the instance above and replace the precedence constraint $\ell \prec mk+1$ by m random precedence constraints. That means, any set of m jobs precedes job $mk+1$, the first job of the job chain, with the same probability. Then with the same arguments as in the Theorem 6, the expected completion time of the jobs in the chain is $k(1-1/(m+1))$ which gives the result. □

3 Conclusion

We presented first results for (stochastic) online scheduling with precedence constraints to minimize the (expected) sum of weighted completion times. The bounds for the single machine setting are tight whereas in the parallel machine setting we leave a gap of $\mathcal{O}(m)$.

For closing this gap, it is not sufficient to run simply a parallel version of the Sept or Spt algorithm. The following deterministic instance shows that the competitive ratio of Spt is also at least in the order of n for the problem with arbitrary job weights.

Example 2. Consider an instance with n jobs and m machines. Job 1 has processing time ε and weight 0. Jobs $2, \ldots, n-m$ have unit processing time and weight 0, and they can start only after job 1 has completed. Job $n-m+1$ has m jobs out of $2, \ldots, n-m$ as direct predecessors and $w_{n-m+1} = 1$ and $p_{n-m+1} = 0$. The adversary chooses the m precedence relations such that these are the latest finishing jobs. Finally there are $m-1$ large independent jobs with processing times $n-m$ and weight 0. When ε tends to 0, the parallel online version of Spt has value $n-m$ whereas an optimal solution has value 1.

A slightly modified instance in which the job with high weight is substituted by a long chain of jobs with zero processing time shows that in the scheduling setting where all jobs have equal weights, the parallel Spt algorithm yields a competitive ratio $\Omega(\sqrt{n})$.

Our final remark concerns preemptive stochastic online scheduling. We mentioned that simple examples show that the Sept policy and actually any other policy that does not utilize preemption can perform arbitrarily bad when jobs are allowed to be interrupted. Therefore other policies must be considered. But still Lemma 7, which provided one of the lower bounds that were used in the analysis, does not do so in the preemptive scheduling environment. However, [18] provides a general lower bound for the preemptive stochastic online scheduling problem on parallel machines. Applied to a variant of the preemptive online policy *Generalized Gittins Index Policy* also proposed in [18] with a modification such that it respects precedence constraints, this could lead to an improved approximation result.

Acknowledgements. We thank Jiří Sgall for pointing out that the randomized lower bounds in previous versions of Theorems 6 and 12 could be strengthened.

References

[1] C. Ambühl and M. Mastrolilli. Single machine precedence constrained scheduling is a vertex cover problem. In Y. Azar and T. Erlebach, editors, *Proceedings of 14th European Symposium on Algorithms*, number 4168 in Lecture Notes in Computer Science, pages 28–39, Zurich, Switzerland, 2006. Springer.

[2] Y. Azar and L. Epstein. On-line scheduling with precedence constraints. *Discrete Applied Mathematics*, 119:169–180, 2002.

[3] S. Ben-David, A. Borodin, R. M. Karp, G. Tardos, and A. Wigderson. On the power of randomization in on-line algorithms. *Algorithmica*, 11:2–14, 1994.

[4] J. R. Correa and A. S. Schulz. Single-machine scheduling with precedence constraints. *Mathematics of Operations Research*, 30(4):1005–1021, 2005.

[5] J. Du, J. Y.-T. Leung, , and G. H. Young. Scheduling chain-structured tasks to minimize makespan and mean flow time. *Information and Computation*, 92(2):219–236, 1991.

[6] W. L. Eastman, S. Even, and I. M. Isaacs. Bounds for the optimal scheduling of n jobs on m processors. *Management Science*, 11:268–279, 1964.

[7] T. Erlebach, V. Kääb, and R. H. Möhring. Scheduling AND/OR-networks on identical parallel machines. In K. Jansen and R. Solis-Oba, editors, *Proceedings of the First International Workshop on Approximation and Online Algorithms, WAOA 2003*, volume 2909 of *Lecture Notes in Computer Science*, pages 123–136, Budapest, Hungary, 2004. Springer.

[8] A. Feldmann, M.-Y. Kao, J. Sgall, and S.-H. Teng. Optimal online scheduling of parallel jobs with dependencies. *Journal of Combinatorial Optimization*, 1(4):393–411, 1998.

[9] 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.

[10] W. J. Hall and J. A. Wellner. Mean residual life. In M. Csörgö, D. A. Dawson, J. N. K. Rao, and A. K. Md. E. Saleh, editors, *Proceedings of the International Symposium on Statistics and Related Topics*, pages 169–184, Ottawa, ON, Canada, 1981.

[11] J. M. Jaffe. Efficient scheduling of tasks without full use of processor resources. *Theoretical Computer Science*, 12:1–17, 1980.

[12] E. L. Lawler. Sequencing jobs to minimize total weighted completion time subject to precedence constraints. *Annals of Discrete Mathematics*, 2:75–90, 1978.

[13] J. K. Lenstra and A. H. G. Rinnooy Kan. Complexity of scheduling under precedence constraints. *Operations Research*, 26(1):22–35, 1978.

[14] R. McNaughton. Scheduling with deadlines and loss functions. *Management Science*, 6:1–12, 1959.

[15] N. Megow. *Coping with incomplete information in scheduling – stochastic and online models*. Dissertation, Technische Universität Berlin, 2006.

[16] N. Megow and A. S. Schulz. On-line scheduling to minimize average completion time revisited. *Operations Research Letters*, 32(5):485–490, 2004.

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

[18] N. Megow and T. Vredeveld. Approximation in preemptive stochastic online scheduling. In Y. Azar and T. Erlebach, editors, *Proceedings of 14th European Symposium on Algorithms*, number 4168 in Lecture Notes in Computer Science, pages 516–527, Zurich, Switzerland, 2006. Springer.

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

[20] R. H. Möhring, A. S. Schulz, and M. Uetz. Approximation in stochastic scheduling: the power of LP-based priority policies. *Journal of the ACM*, 46:924–942, 1999.

[21] K. R. Pruhs, J. Sgall, and E. Torng. Online scheduling. In J. Y.-T. Leung, editor, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, chapter 15. Chapman & Hall/CRC, 2004.

[22] M. Queyranne and A. S. Schulz. Approximation bounds for a general class of precedence constrained parallel machine scheduling problems. *SIAM Journal on Computing*, 35(5):1241–1253, 2006.

[23] A. S. Schulz and M. Skutella. The power of α-points in preemptive single machine scheduling. *Journal of Scheduling*, 5:121–133, 2002.

[24] J. Sgall. On-line scheduling – a survey. In A. Fiat and G. J. Woeginger, editors, *Online Algorithms: The State of the Art*, volume 1442 of *Lecture Notes in Computer Science*, pages 196–231. Springer, Berlin, 1998.

[25] M. Skutella and M. Uetz. Stochastic machine scheduling with precedence constraints. *SIAM Journal on Computing*, 34(4):788–802, 2005.

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

[27] V. Timkovsky. Identical parallel machines vs. unit-time shops and preemptions vs. chains in scheduling complexity. *European J. Oper. Res.*, 149(2):355–376, 2003.

[28] G. Weiss. On almost optimal priority rules for preemptive scheduling of stochastic jobs on parallel machines. *Advances in Applied Probability*, 27:827–845, 1995.

[29] A. C.-C. Yao. Probabilistic computations: toward a unified measure of complexity (extended abstract). In *Proceedings of the 18th IEEE Symposium on the Foundations of Computer Science*, pages 222–227, 1977.