Stochastic Online Scheduling

Maastricht University

Tjark Vredeveld

Stochastic Online Scheduling

RM/09/052

METEOR

Maastricht University School of Business and Economics
Maastricht Research School of Economics
of Technology and Organization

P.O. Box 616
NL - 6200 MD Maastricht
The Netherlands

Stochastic Online Scheduling

Tjark Vredeveld*

November 5, 2009

Abstract

In this paper we consider a model for scheduling under uncertainty. In this model, we combine the main characteristics of online and stochastic scheduling in a simple and natural way. Jobs arrive in an online manner and as soon as a job becomes known, the scheduler only learns about the probability distribution of the processing time and not the actual processing time. This model is called the stochastic online scheduling (SOS) model. Both online scheduling and stochastic scheduling are special cases of this model. In this paper, we survey the results for the SOS model.

1 Introduction

Machine scheduling problems belong to the classical problems in combinatorial optimization. These problems play a role whenever jobs need to be processed on a limited number of machines or processors, with applications in manufacturing, parallel computing [3] or compiler optimization [5]. Machine scheduling problems have been studied since the 1950s and for a general overview of the vast amount of literature we refer to the books by Brucker [2] and Pinedo [24] and to the handbook of scheduling by Leung [17].

In standard deterministic scheduling all relevant data to the problem is known a priori. However, this assumption is not always realistic. In many scenarios, we need to find a good schedule when the data is not fully available and decisions with wide-ranging implications need to be taken in the face of incomplete data. To cope with these uncertainties, there are two major frameworks in the theory of scheduling: *online scheduling* and *stochastic scheduling*. In online scheduling models the instance is only presented to the scheduler piecewise. Jobs are either arriving one by one (online list model) or over time (online time model). The actual processing time of a job is usually disclosed upon arrival of the job and decisions must be made without any knowledge of the jobs to come. See the survey of Pruhs, Sgall, and Torng [26] for an overview on online scheduling. In stochastic scheduling, the population of jobs is assumed to be known beforehand, but in contrast to deterministic models, the processing times of jobs are only given by a probability distribution. The actual processing times become known only upon completion of the jobs. The distribution functions of the random variables that describe the processing times, or at least their first

*Maastricht University, Dept. of Quantitative Economics, P.O.Box 616, 6200 MD Maastricht, The Netherlands, t.vredeveld@maastrichtuniversity.nl

and second moment, are assumed to be known beforehand. See the survey of Pinedo [25] and the PhD theses [32, 9] for overviews on stochastic scheduling.

Recently, a combined model was introduced [7, 19] that generalizes both stochastic scheduling and online scheduling. Like in online scheduling, we assume that the instance is presented to the scheduler piecewise, and nothing is known about jobs that might arrive in the future. Once a job arrives, like in stochastic scheduling, we assume that its expected processing time, or the distribution function of the processing time, is disclosed, but the actual processing time remains unknown until the job completes. In this survey, we will review the results on this stochastic online scheduling model.

2 Model and definitions

In the machine scheduling models that we consider, we are given a set of n jobs $J = \{1, \ldots, n\}$ each of which has to be scheduling on one or all of m machines. Each machine can process at most one job at a time and is available from the beginning. Job $j \in J$ has release date r_j, which is the earliest possible time at which this job may be processed on. Moreover, with job j, we associate a nonnegative weight w_j. In this survey, we only consider machine scheduling problems in which the goal is to find a schedule that minimizes the total weighted completion time $\sum_j w_j C_j$, where C_j denotes the completion time of job j. Depending on the model, we may or may not preempt a job, that is, interrupt a job and continue its processing later on the same or another machine.

The time it takes to process job j on machine i is denoted by the random variable P_{ij}. In this survey, we consider several machine environments each with there own restrictions on the processing times. In single machine models, there is only one machine to process all jobs and therefore, we denote the processing time by P_j instead of P_{1j}. In the identical parallel machine model, each job has to be processed by only one machine and as the machines are identical, we have that the processing time of a job is not machine dependent, i. e., $P_{ij} = P_j$. Also in the uniformly related machine environment, jobs need to be processed by only one machine. Each machine has a certain speed denoted by s_i and the processing time of job j on machine i is given by $P_{ij} = P_j/s_i$, where P_j is the random variable denoting the processing requirement of job j. The last model that we will consider in this survey is the flow shop. In the flow shop problem a set of n jobs needs to be processed non-preemptively on m machines. Each machine can process at most one job at a time and each job can be processed by at most one machine at a time. Each job must be processed by each machine in the same order.

The goal is to find a *stochastic online scheduling* (SOS) policy that minimizes the objective function in expectation. The definition of an SOS policy extends the traditional definition of stochastic scheduling policies by Möhring, Radermacher, and Weiss [22] to the setting where jobs arrive online. A scheduling policy specifies *actions* at decision time t. An action is a set of jobs that is started or, in case of preemption, interrupted at time t and a next decision time $t' > t$ at which the next action is taken, unless some job is released or ends at time $t'' < t'$. In that case, t'' becomes the next decision time. To decide, the policy may utilize the complete information contained in the partial schedule up to time t, as well as information about unscheduled jobs that have arrived

at or before t. However, a policy is required to be *online*, thus at any time, it must not utilize any information about jobs that will be released in the future. Moreover, it needs to be *non-anticipatory*, thus at any time, it must not utilize the actual processing times of jobs that are scheduled or unscheduled but not yet completed. An *optimal scheduling policy* is defined as a non-anticipatory scheduling policy that minimizes the objective function value in expectation. Note that we do not assume that an optimal policy needs to be online. Also notice that even an optimal scheduling policy generally fails to yield an optimal solution for all realizations of the processing times; this is because it is non-anticipatory.

For an instance I, consisting of the number of machines m, the set of jobs J together with their release dates r_j, weights w_j, and processing time distributions P_{ij}, let $C_j^{\Pi}(I)$ denote the random variable for the completion time of job j under policy Π. When the instance is clear from the context, we write C_j^{Π} for short. Let

$$\mathbb{E}\left[\Pi(I)\right] = \mathbb{E}\left[\sum_{j\in J} w_j C_j^{\Pi}(I)\right] = \sum_{j\in J} w_j \mathbb{E}\left[C_j^{\Pi}(I)\right]$$

denote the expected performance of a scheduling policy Π on instance I.

Generalizing the definitions of by Möhring, Schulz, and Uetz [23] for traditional stochastic scheduling, we define the performance guarantee of an SOS policy as follows.

Definition 1 *An SOS policy Π is a* ρ-approximation *if, for some $\rho \geq 1$, and all instances I of the given problem,*

$$\mathbb{E}\left[\Pi(I)\right] \leq \rho \mathbb{E}\left[OPT(I)\right].$$

Here, $OPT(I)$ denotes an optimal stochastic scheduling policy on the given instance I, assuming a priori knowledge of the set of jobs J, their weights w_j, release dates r_j and processing time distributions P_{ij}. The value ρ is called the *performance guarantee* or *approximation ratio* of policy Π. The *asymptotic approximation ratio* of a policy Π is given by

$$\rho^{\infty} = \inf\left\{\rho \geq 1 \,:\, \exists N_0 \text{ s.t. } \frac{\mathbb{E}\left[\Pi(I)\right]}{\mathbb{E}\left[OPT(I)\right]} \leq \rho, \text{ for all instances } I \text{ with } |J| \geq N_0\right\}.$$

The asymptotic approximation ratio characterizes the maximum relative deviation from optimality for all sufficiently large instances. If a stochastic scheduling policy has an asymptotic approximation ratio of one, then we say it is asymptotically optimal.

3 Single machine

The first results on stochastic online scheduling have been obtained for the nonpreemptive single machine environment [7, 19], although [19] also considered the identical parallel machine environment.

Whenever all release dates are the same, i. e., $r_j = 0$ for all $j = 1, \ldots, n$, then the deterministic as well as the stochastic single machine problem can be solved to optimality by a simple rule, called the Weighted Shortest (Expected)

Processing Time rule (WSEPT): process the jobs in non-increasing order of weight over (expected) processing time [30, 27]. A straightforward extension to the problem in which there are non-trivial release dates, is the policy that whenever the machine is idle it will process a job that has highest ratio of weight to expected processing time, $w_j/\mathbb{E}[P_j]$. Chou, Liu, Queryanne, and Simchi-Levi [7] named this policy the Weighted Shortest Processing Time among Available jobs (WSEPTA) rule and performed an asymptotic analysis for this rule. They showed that whenever the weights are bounded from above and below by some arbitrary constants, i. e., there are constants $\underline{w}$ and $\overline{w}$ such that $\underline{w} \leq w_j \leq \overline{w}$ for all $j \in J$, and there are some upper and lower bounds on the possible realizations of the processing times, i. e., there are some constants $\underline{x}$ and $\overline{x}$ such that $\Pr[\underline{x} \leq P_j \leq \overline{x}] = 1$ for all $j \in J$, the ratio between the expected performance of the WSEPTA rule and the expected total weighted completion time of the optimal policy tends to 1, when the number of jobs tends to infinity. To show their results they first prove that the value of the LP-relaxation from Goemans [14, 15] on expected processing times yields a lower bound on the optimal value of the stochastic scheduling problem. This lower bound for stochastic scheduling has first been obtained by Möhring et al. [23]. Then Chou et al. show that the gap between the expected value of WSEPTA and the lower bound is relatively small, that is, $o(n^2\overline{w}\,\overline{x})$, whereas the expected performance of the optimal policy is at least $n(n+1)\underline{w}\,\underline{x}/2$. This last bound can be easily obtained by computing the optimal value for an instance with n jobs, all having processing time $\underline{x}$, weight $\underline{w}$ and release date 0. When the assumption that the processing are bounded from below by a positive constant is not satisfied, then any non-idling, and thus WSEPTA, can be arbitrarily bad.

The asymptotic analysis of Chou et al. has been extended by Chen and Shen [6] to an asymptotic analysis for any non-idling policy. However, Chen and Shen not only assume uniform bounds on the weights and processing times, but they also assume that the processing times are i.i.d. with mean μ and the interarrival times are also i.i.d. with mean $\lambda > \mu$. Under these assumptions, they show that any non-idling policy is optimal for the non-preemptive single machine stochastic scheduling problem as to minimize the expected total weighted completion time. They showed that when the interarrival times are larger than the processing times on average, the total flow time is insignificant compared to the sum of the release dates. Using the same kind of reasoning, Chen and Shen also show that any non-idling policy is asymptotically optimal for the flowshop and uniformly related machine environment.

The first non-asymptotic analysis for stochastic online scheduling on a single machine as to minimize the expected total weighted completion time has been given by Megow, Uetz, and Vredeveld [19]. They consider a modified version of the WSEPT rule, the α-shift WSEPT policy: Given fixed $\alpha > 0$, when a job arrives, modify its release date to $r'_j = \max\{r_j, \alpha\mathbb{E}[P_j]\}$. At any time t, when the machine is idle, start processing the job with the highest ratio $w_j/\mathbb{E}[P_j]$ among all available jobs with $t \geq r'_j$. The deterministic version of this policy has been proposed by Megow and Schulz [18]. To analyze this policy, Megow et al. [19] introduce the concept of δ-NBUE random variables, extending the notion of NBUE (new better than used in expectation)random variables. A random variable X is said to be δ-NBUE if for all $x > 0$ it holds that $\mathbb{E}[X - x \,|\, X > x] \leq \delta\,\mathbb{E}[X]$. An NBUE random variable is 1-NBUE. Given that all processing times are δ-NBUE, they show that the α-shift WSEPT policy is

a $(2+\delta)$-approximation for $\alpha = 1$. The intuition behind the proof of this approximation ratio is that as soon as a job j becomes available according to its modified release date, only higher priority jobs, i. e., jobs with higher ratio $w_k/\mathbb{E}[P_k]$, will be processed up to the start of this job plus possibly a job, ℓ that is in process at its modified release date. To bound the remaining processing time of this job ℓ, we use the property of δ-NBUE random variables and the fact that due to the modification of the release dates this job satisfies $\mathbb{E}[P_\ell] \leq \mathbb{E}[P_j]$.

The literature for deterministic online scheduling when there are precedence relations among the jobs is rather limited. A natural online paradigm for online scheduling with precedence relations is given by Feldmann, Kao, Sgall, and Teng [11]: a job becomes known to the online scheduler as soon as all its predecessors have finished. Feldmann et al. study the online problem to minimize the makespan, i. e., the latest completion time of a job. Erlebach, Kääb, and Möhring [10] studied the deterministic online problem for and/or-precedence relations, which is a generalization of the general precedence constraints, when the goal is to minimize the total weighted completion time. They analyzed the performance of the Shortest Processing Time (SPT) rule that schedules the jobs in order of non-decreasing processing times and showed that it is a $2\sqrt{n}$-approximation if all weights are equal and an n-approximation for arbitrary weights. Megow and Vredeveld [21] extended these results to the stochastic online setting and also improved on them. They showed that the Shortest *Expected* Processing Time (SEPT) rule attains a performance guarantee of $\sqrt{2n}$ if all jobs have equal weights and n for arbitrary weights. To prove the upper bound of n, Erlebach et al. introduced the concept of a threshold of a job j, which is the largest processing time of a job that is completed before job j. They then showed that for each job j the algorithm that minimizes the threshold is the SEPT rule and thus is the completion time of job j at least its threshold. Megow and Vredeveld extended the definition of the threshold to the stochastic setting and showed that a similar property on the thresholds hold true for the SEPT policy. Megow and Vredeveld also gave lower bounds on the approximation ratio for any stochastic online scheduling policy: for arbitrary weights the lower bound is $n-1$, whereas no online policy can be better than a $2\sqrt{n}/3-1$ approximation when all weights are equal.

Let us now consider single machine scheduling when preemption is allowed. Some of the first results in this setting that can be found in the literature are by Chazan, Konheim, and Weiss [4] and Konheim [16]. They formulated sufficient and necessary conditions for a policy to solve optimally the single machine problem in which all jobs have the same release date. Later Sevcik [29] developed an intuitive method for creating optimal schedules in expectation. He introduces a priority policy that relies on an index which can be computed for each job based on the properties of a job, but not on other jobs. Gittins [12] showed that this priority index is a special case of his Gittins index [12, 13]. Twenty years after Sevcik presented the priority policy, Weiss [33] formulated Sevcik's priority index again in terms of the Gittins index and provided a different proof of the optimality of the priority policy. Weiss named this policy a Gittins Index Priority Policy (GIPP).

GIPP computes an index, or *rank*, for each job. This rank is not dependent on other jobs, but changes with the amount of processing a job has received, and thus also changes over time. We thus can compute a tentative schedule, assuming that no job will ever finish before it received its maximum possible

processing time. This tentative schedule can be seen as an ordered list of job pieces. This amount of processing time for each piece is the time spent on the job before it will be preempted. GIPP then always processes the next uncompleted job in the list for the specified amount of time, or up to completion, whatever comes first.

Megow and Vredeveld [20] formulated two variants of this GIPP that work online. The first one, called Follow GIPP (F-GIPP), computes at any release date t the ordered list of job pieces and their processing times for the jobs that have been released at or before time t. Note that this list does not take the release dates into account. It then deletes all the pieces that have been processed up to time t, possibly decreasing the amount of processing time for certain job pieces. It then follows the GIPP until the next release date. It can easily be shown that this is 2-approximation policy. For any realization it can be shown that between the release date and the completion time of a job j, only job pieces are processed that also would have been processed before the completion of job j in the schedule obtained by the optimal policy for the relaxed problem in which the jobs have no release date. Moreover, no job can complete before its release date.

The second policy defined by Megow and Vredeveld is easier to formulate: it always processes the job that has currently the highest rank. We call this policy Gen-GIPP. Unfortunately, for this policy we cannot claim that in any realization of the processing times, between release and completion of a job only pieces are processed that also would have been processed in the schedule constructed by the optimal policy for the problem without release dates. That is, compared to the schedule obtained by F-GIPP, some jobs may be delayed. However, the expected gain from a job j that delays a certain (piece) of job k is more than the expected loss of the delay of job k. Therefore, this policy is also a 2-approximation.

4 Multiple machines

Besides the asymptotic analysis for the non-preemptive single machine problem as discussed in the previous section, Chen and Shen [6] also considered two multiple machine models: uniform related machines and the flow shop problem. Recall that in the flow shop problem, each job must be processed by each machine in the same order, $1, \ldots, m$. Chen and Shen made the following assumptions for the flow shop problem: there exist uniform bounds on the processing times and weights, i. e., there exist constants $\underline{w}, \overline{w}$ and $\underline{x}, \overline{x} > 0$ such that $\underline{w} \leq w_j \leq \overline{w}$ and $\Pr[\underline{x} \leq P_{ij} \leq \overline{x}] = 1$. Moreover, they assumed that the interarrival times are i.i.d. with mean λ, the processing times P_{ij} are i.i.d. with mean $\mu < \lambda$. They then showed that for any non-idling policy the sum over all jobs of the time between the completion of the first operation and the last operation is insignificant compared to the sum of the weighted release dates. Therefore, they can use the result for the single machine case and thus is any non-idling policy asymptotically optimal.

In the uniformly related parallel machine setting, each job needs to be processed by exactly one of m machines and a machine i has constant speed $s_i > 0$ and processing a job j on machine i takes P_j/s_i time, where P_j is a random variable denoting the processing requirement of job j. Again, Chen and Shen assume

that there exists uniform bounds on the weights and processing requirements as well as i.i.d. distributed interarrival times and i.i.d. distributed processing requirements. The mean of the processing requirements, μ, is assumed to be larger than the sum of all machine speeds times the average interarrival time, $\lambda \sum_i s_i$. Under these assumptions, they show that the non-idling policy First Come First Serve (FCFS) is asymptotically optimal. To come to this result, Chen and Shen first bound the average waiting time of the jobs by the average waiting time of a specific *fixed assignment policy*. In a fixed assignment policy a job is assigned to a machine as soon as it arrives. Then, each machine follows its own policy, in this case FCFS, for the jobs that are assigned to it. Due to the assumptions on the interarrival times and processing requirements and due to the way of assigning the jobs to the machines, the problem for each machine satisfies the assumptions for the single machine problem, and therefore it can be shown that the total weighted waiting time as well as the total weighted processing times are insignificant compared to the total weighted release date.

As mentioned in the previous section, the first result for parallel machines in the stochastic online setting is by Megow, Uetz, and Vredeveld [19] for the problem in which preemption is not allowed. They consider a fixed assignment policy in which each machine schedules the jobs assigned to it according to the α-shift WSEPT. By assigning the jobs in a greedy manner to the machines, i. e., assigning a job to the machine on which it has the minimal expected increase in the objective function, we can show that this policy is a ρ-approximation, for $\rho = 1 + \max\{1 + \delta/\alpha, \alpha + \delta + (m-1)(\Delta+1)/2m\}$, for δ-NBUE processing times. Here Δ is a bound on the squared coefficient of variation of the processing times, $Var[P_j]/\mathbb{E}[P_j]^2 \leq \Delta$. For NBUE processing times, where $\Delta = \delta = 1$, we obtain a performance guarantee which is less than $(5+\sqrt{5})/2 - 1/(2m) \approx 3.62 - 1/(2m)$, when we choose the right α. This bound is better than the previously best known bound of $4 - 1/m$ of Möhring et al. [23], even though their policy was not an online policy. The assignment strategy of the jobs to the machines in the above described policy can be viewed as a derandomization of the strategy in which each job is assigned uniformly at random to one of the m machines. This random strategy has the same worst-case performance ratio as the derandomized version.

Megow et al. [19] also show a lower bound on the performance ratio that can be obtained by a fixed assignment policy. The show that if all processing times are i.i.d. and exponentially distributed, that there exist instances such that any fixed assignment policy on these instances have an expected solution value of at least $3(\sqrt{2}-1)$ times as large the expected solution value of an optimal policy.

Schulz [28] improved on these results. He gave a randomized online policy that achieves a bound of $2 + \Delta$. His policy is also a fixed assignment policy and is an extension of an online algorithm proposed by Correa and Wagner [8] for deterministic scheduling to the stochastic scheduling setting. This policy first computes a virtual preemptive fast single machine schedule for jobs with deterministic processing times equal to the expectation of the processing times based on the ideas of Goemans [14, 15] and uses the concept of α-points, introduced by Sousa [31] to determine the time at which a job becomes available for scheduling on a randomly selected machine. A derandomized version, which is not a fixed assignment policy, attains a approximation ratio of $\max\{\phi + 1, ((\phi+1)\Delta + \phi + 3)/2\}$, where again Δ is a bound on the squared coefficient of variation and $\phi = (1+\sqrt{5})/2$ is the golden ratio.

When precedence constraints are present, Megow and Vredeveld consider an version of the SEPT policy that utilizes only one machine, which they call the 1-SEPT policy. They prove that this is a n-approximation policy and also show a lower bound of $(n-1)/m$ for any online policy. To prove the upper bound of n, they basically use the same technique as in the single machine case. In case that the weights are all equal, they show that this policy is a $\sqrt{2mn}$-approximation and that no online policy can have an approximation ratio that is less than $(2\sqrt{n/m})/3 - 1$.

For the preemptive problem, Megow and Vredeveld [20] extend the F-GIPP policy to identical parallel machines: at any time process the m first jobs in the list of job pieces, or if less than m uncompleted jobs are present process all uncompleted jobs. They show that this policy is a 2-approximation by bounding the expected value of the optimal policy by the optimal value of a single machine stochastic scheduling problem in which the processing times are a factor m smaller. Besides the extension of the F-GIPP policy to multiple machines, they also provide a randomized fixed assignment policy with a performance guarantee of 2: assign each job uniformly at random to one of the m machines and run on each machine Gen-GIPP.

It is worth noticing that, unlike the known results for non-preemptive scheduling, the approximation guarantees for these preemptive policies are not dependent on properties of the probability distribution, such as the squared coefficient of variation. Actually, the guarantee for F-GIPP is the same as its deterministic counterpart that at any moment in time processes the at most m jobs with highest ratio of weight to processing time, see [18]. On the other hand, the non-preemptive policies work well if only information about the first and second moment of the processing times are given, whereas our preemptive policies need to know the complete probability distribution.

5 Concluding remarks

The area of stochastic online scheduling is relatively new. However, several results have been obtained so far, but many more open problems remain. Up to now, only results are known when the objective is to minimize the total weighted completion time and it would be interesting to see what results can be obtained for other objective functions, like minimizing the expected makespan or the expected total flow time.

A big difference between stochastic online scheduling and deterministic online scheduling can be found in the single machine problem in which we schedule the jobs preemptively to minimize the total completion time. In the deterministic setting, an optimal solution can be found by a simple online algorithm that always processes the job with shortest remaining processing time. On the other hand, we have shown that the optimal stochastic scheduling policy for this problem cannot be an online one [1].

Another interesting question comes from the difference in the results for the preemptive and non-preemptive problems. We have seen that for the non-preemptive problem on identical machines, the proposed stochastic scheduling policies only need to know the first and second moment of the probability distributions of the processing times, whereas the proposed policies for the non-preemptive problems need to have full knowledge of the random variables and

their distribution functions. Then again, the performance guarantee of these policies is independent of the properties of the distribution functions, whereas the obtained performance guarantees for the non-preemptive problem depend properties like the coefficient of variation and δ-NBUE. This raises the question what the influence of knowledge of the probability distributions is on the performance guarantee.

References

[1] L. Becchetti, A. Marchetti-Spaccamela, G. Schäfer, and T. Vredeveld. On scheduling stochastic jobs to minimize the expected total flow time. unpublished manuscript, 2006.

[2] P. Brucker. *Scheduling Algorithms*. Springer, 4th edition, 2004.

[3] S. Chakrabarti and S. Muthukrishnan. Resource scheduling for parallel database and scientific applications. In *Proceedings of the 8th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 329–335, 1996.

[4] D. Chazan, A. G. Konheim, and B. Weiss. A note on time sharing. *Journal of Combinatorial Theory*, 5:344–369, 1968.

[5] C. Chekuri, R. Johnson, R. Motwani, B. Natarajan, B. Rau, and M. Schlansker. An analysis of profile-driven instruction level parallel scheduling with application to super blocks. In *Proceedings 29th IEEE/ACM Int. Symp. on Microarchitecture*, pages 58–69, Paris, France, 1996.

[6] G. Chen and Z.-J. M. Shen. Probabilistic asymptotic analysis of stochastic online scheduling problems. *IIE Transactions*, 39:525–538, 2007.

[7] C.-F. M. Chou, H. Liu, M. Queyranne, and D. Simchi-Levi. On the asymptotic optimality of a simple on-line algorithm for the stochastic single machine weighted completion time problem and its extensions. *Operations Research*, 54(3):464–474, 2006.

[8] J. Correa and M. Wagner. LP-based online scheduling: from single to parallel machines. *Mathematical Programming*, 119:109–136, 2009.

[9] B. C. Dean. *Approximation Algorithms for Stochastic Scheduling Problems*. PhD thesis, Massachusetts Institute of Technology, 2005.

[10] 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.

[11] 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.

[12] J. C. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society, Series B*, 41:148–177, 1979.

[13] J. C. Gittins. *Multi-armed Bandit Allocation Indices.* Wiley, New York, 1989.

[14] M. X. Goemans. Improved approximation algorithms for scheduling with release dates. In *Proceedings of the 8th ACM-SIAM Symposium on Discrete Algorithms*, pages 591–598, New Orleans, LA, USA, 1997.

[15] M. X. Goemans, M. Queyranne, A. S. Schulz, M. Skutella, and Y. Wang. Single machine scheduling with release dates. *SIAM Journal on Discrete Mathematics*, 15:165–192, 2002.

[16] A. G. Konheim. A note on time sharing with preferred customers. *Probability Theory and Related Fields*, 9:112–130, 1968.

[17] J. Y.-T. Leung. *Handbook of Scheduling: Algorithms, Models, and Performance Analysis.* Chapman & Hall/CRC, 2004.

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

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

[20] 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.

[21] N. Megow and T. Vredeveld. Stochastic online scheduling with precedence constraints. Technical Report 029-2007, Technische Universität Berlin, 2007.

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

[23] 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.

[24] M. Pinedo. *Scheduling: Theory, Algorithms, and Systems.* Springer Verlag, Heidelberg, 3rd edition, 2002.

[25] M. Pinedo. Off-line deterministic scheduling, stochastic scheduling, and online deterministic scheduling: A comparative overview. In J. Y.-T. Leung, editor, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, chapter 38. Chapman & Hall/CRC, 2004.

[26] 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.

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

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

[29] K. C. Sevcik. Scheduling for minimum total loss using service time distributions. *Journal of the ACM*, 21:65–75, 1974.

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

[31] J. Sousa. *Time Indexed Formulations of Non-Preemptive Single-Machine Scheduling Problems*. PhD thesis, Université Catholique de Louvain, 1989.

[32] M. Uetz. *Algorithms for Deterministic and Stochastic Scheduling*. Cuvillier Verlag, Göttingen, Germany, 2002.

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