# Getting rid of stochasticity: applicable sometimes

Marjan van den Akker [*] Han Hoogeveen [†]

## Abstract

We consider the single-machine scheduling problem of minimizing the number of late jobs. We omit here one of the standard assumptions in scheduling theory, which is that the processing times are deterministic. Our main message is that in a number of cases the problem with stochastic processing times can be reformulated as a deterministic problem, which is solvable in polynomial time through the famous algorithm by Moore and Hodgson. We first review and reinterpret this algorithm as a dynamic programming algorithm. We then consider four problem classes with stochastic processing times. The first one has equally disturbed processing times, that is, the processing time consist of a deterministic part and a random component that is independently, identically distributed for each job. The jobs in the other three classes have processing times that follow: (i) A gamma distribution with shape parameter $p_j$ and scale parameter $\beta$, where $\beta$ is common to all jobs; (ii) A negative binomial distribution with parameters $p_j$ and $r$, where $r$ is the same for each job; (iii) A normal distribution with parameters $p_j$ and $\sigma_j^2$.

In this scheduling environment, the completion times will be stochastic variables as well. Instead of looking at the expected number of on time jobs, we introduce the concept of a job being 'stochastically on time', that is, we qualify a job as being on time if the probability that it is completed by the deterministic due date is at least equal to a certain given minimum success probability. We show that in case of equally disturbed processing times we can solve the problem in $O(n \log n)$ time through the algorithm by Moore and Hodgson, if we make the additional assumption that the due dates and the minimum success probabilities are agreeable, which encompasses the case of equal minimum success probabilities. The problems with processing times following a gamma or a negative binomial distribution can be solved in $O(n \log n)$ time by Moore and Hodgson's algorithm, even if the minimum success probabilities are arbitrary; based on these two examples, we characterize the properties that a distribution must possess to allow such a result. For the case with normally distributed processing times we need the additional assumption that the due dates and minimum success probabilities are agreeable. Under this assumption we present a pseudo-polynomial time algorithm, and we prove that this is the best we can hope for by establishing weak $\mathcal{NP}$-hardness. We also show that the problem of minimizing the weighted number of late jobs can be solved by an extension of the dynamic programming algorithm in all four cases; this takes pseudo-polynomial time. We further indicate how the problem of maximizing the expected number of on time jobs (with respect to the standard definition) can be tackled if we add the constraint that the on time jobs are sequenced in a given order.

**Keywords.** Scheduling, sequencing, single machine, number of late jobs, stochastic processing times, minimum success probability, dynamic programming, NP-hardness.

[*] E-mail: marjan@cs.uu.nl. Department of Computer Science, Utrecht University, P.O.Box 80089, 3508 TB Utrecht, The Netherlands. Supported by EC Contract IST-1999-14186 (Project alcom-FT).

[†] E-mail: slam@cs.uu.nl. Department of Computer Science, Utrecht University, P.O.Box 80089, 3508 TB Utrecht, The Netherlands. Supported by EC Contract IST-1999-14186 (Project alcom-FT).

# 1 Introduction

One of the standard complaints of many people nowadays is ‘so many things to do, so little time’. If it is not possible to increase the amount of time available (for instance by hiring somebody), then it may be inevitable to skip some tasks. The question then becomes of course: which tasks should be skipped? This boils down to the machine scheduling problem that was studied in the famous paper by Moore (1968). The busy person is translated into a single machine, and the things to do are called jobs. Formally, the problem is then formulated as follows. The machine is assumed to be continuously available from time zero onwards, and it can perform at most one job at a time. The machine has to execute $n$ jobs, denoted by $J_1, \ldots, J_n$. Performing task $J_j$ requires a period of length $p_j$, and the execution of this task is preferably finished by its due date $d_j$. If job $J_j$ is finished after its due date, then it is marked as late. The objective is to minimize the number of late jobs. Since for this objective it does not matter at which time a late job is finished, such a job can just as well be skipped altogether; the machine then only carries out the jobs that will finish on time. Since it is not acceptable for a customer to just hear ‘sorry, we did not make it’ at the due date, we assume that the firm has to tell the potential client at time zero whether it will honor its request. If the request is denied, then the client will go elsewhere, and the company does not have to execute this job. Note that, if all information is available at time zero in a deterministic situation, then this assumption is in fact irrelevant. Moore shows that the problem of maximizing the number of jobs that are finished on time can be solved in $O(n \log n)$ time by an algorithm that since then is known as Moore-Hodgson’s algorithm.

In this setting, each job is equally important. In many applications, however, some jobs are more important than others. This importance can be measured by assigning a positive weight $w_j$ to each job $J_j$ $(j = 1, \ldots, n)$; the objective function then becomes to minimize the total weight of the late jobs. Lawler and Moore (1969) show that this problem is solvable in $O(n \sum p_j)$ time by dynamic programming. Karp (1972) shows that pseudo-polynomial running time is unavoidable for this problem (unless $\mathcal{P} = \mathcal{NP}$) by establishing $\mathcal{NP}$-hardness in the ordinary sense, even if all due dates are equal.

In this paper, we look at the problems described above, but we abolish one of the common assumptions of scheduling theory, which is that the data are deterministic. We consider four specific classes of instances. In the first one the processing times are stochastic variables that are distributed according to a gamma distribution with parameters $p_j$ (which varies per job) and $\beta$ (which is equal for all jobs). The gamma distribution is often applied to model the processing time of a task (see for instance Law and Kelton, 2000). The second class of processing times is used to model a production process where items are produced that work well with probability $r$ and malfunction with probability $(1 - r)$; a job $J_j$ corresponds then to an order of $p_j$ correctly functioning items. The corresponding processing time then follows a negative binomial distribution with parameters $p_j$ and $r$. In the third class the processing times consist of a deterministic component $p_j$ and a random disturbance, which we assume to be identically distributed for each job. This can be used to model the situation that the disturbances in the production process are not job-related but due to some side-equipment that is used by each job in the same way. In the last case, we assume that the processing times follow a normal distribution with known expected value $p_j$ and known variance $\sigma_j^2$.

We suppose that each due date is deterministic, which is reasonable, as they are specified by the customer issuing the request. More importantly, we further assume that this customer is willing to accept a delayed completion of his/her order, if the company can convince him/her that the planning is such that the probability that the order is delayed is 'small enough'. This is achieved by guaranteeing that the probability that the order is on time is at least equal to some given lower bound value, which we define as the *minimum success probability*, and which we denote by $y_j$ $(j = 1, \ldots, n)$. If the customer prefers to be convinced by hard cash, then you can agree that he/her will be compensated if the completion is delayed; when the probability distribution of the completion time of job $J_j$ is known, then working with a minimum success probability boils down to specifying an upper bound for the expected compensation payment, which corresponds to a lower bound on the expected profit, and vice versa.

The remainder of the paper is organized as follows. In Section 2 we review the problem of minimizing the number of late jobs with deterministic processing times, and we explain Moore-Hodgson's algorithm as a dynamic programming algorithm. In Section 3 we discuss the consequences of working with stochastic processing times. We show that the first and second class of processing times can be reformulated as deterministic problems, and hence can be dealt with by the traditional algorithms. We further specify a number of constraints such that, if a probability distribution satisfies these, then the problem with stochastic processing times following this distribution is solvable in $O(n \log n)$ time irrespective of the minimum success probabilities; the first two classes of instances satisfy these conditions. For the other two classes of processing times, we need the additional assumption that the minimum success probabilities and the due dates are *agreeable*, which here implies that the jobs can be numbered such that $i < j$ implies that $d_i \leq d_j$ and $y_i \geq y_j$. We develop dynamic programming algorithms that minimize the (weighted) number of late jobs for these instances; these algorithms are based on the insight gained in Section 2. We further discuss the problems that we face in case of general minimum success probabilities. In Section 4, we show that the problem with stochastic processing times that follow a normal distribution is fundamentally more difficult than the problem with deterministic processing times by establishing ordinary $\mathcal{NP}$-hardness for the problem of minimizing the number of late jobs. In Section 5, we address the problem of minimizing the expected value of the (weighted) number of late jobs under the side-constraint that the on time jobs are executed in any given order.

# References

[1] J.M. van den Akker and J.A. Hoogeveen (2004). Minimizing the number of tardy jobs. In: J.Y.-T. Leung (ed.), *Handbook of Scheduling*, Algorithms, Models, and Performance Analysis, pp. 227–243, CRC Press, Inc. Boca Raton, Fl, USA.

[2] J.R. Jackson (1955). Scheduling a production line to minimize maximum tardiness. Research Report 43, Management Science Research Project, University of California, Los Angeles.

[3] R.M. Karp (1972). Reducibility among combinatorial problems, in: R.E. Miller and J.W. Thatcher (eds.) (1972). *Complexity of Computer Computations*, Plenum Press, New York, pp. 85-103.

[4] A.M. Law and W.D. Kelton (2000). *Simulation modeling and analysis.* McGraw Hill.

[5] E.L. Lawler (-). Scheduling a single machine to minimize the number of late jobs. Unpublished manuscript.

[6] E.L. Lawler and J.M. Moore (1969). A functional equation and its application to resource allocation and sequencing problems. *Management Science 16*, 77-84.

[7] E.L. Lawler (1973). Optimal sequencing of a single machine subject to precedence constraints. *Management Science 19*, 544-546.

[8] J.M. Moore (1968). An $n$ job, one machine sequencing algorithm for minimizing the number of late jobs. *Management Science 15*, 102-109.