This article was downloaded by: [216.73.217.52] On: 26 November 2026, At: 21:43
Publisher: Institute for Operations Research and the Management Sciences (INFORMS)
INFORMS is located in Maryland, USA



INFORMS TutORials in Operations Research

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

# Safe Scheduling

Kenneth R. BakerDan Trietsch

To cite this entry: Kenneth R. BakerDan Trietsch. Safe Scheduling. *In* INFORMS TutORials in Operations Research. Published online: 14 Oct 2014; 79-101.
https://doi.org/10.1287/educ.1073.0031

Full terms and conditions of use: https://pubsonline.informs.org/Publications/Librarians-Portal/PubsOnLine-Terms-and-Conditions

This article may be used only for the purposes of research, teaching, and/or private study. Commercial use or systematic downloading (by robots or other automatic processes) is prohibited without explicit Publisher approval, unless otherwise noted. For more information, contact permissions@informs.org.

The Publisher does not warrant or guarantee the article's accuracy, completeness, merchantability, fitness for a particular purpose, or non-infringement. Descriptions of, or references to, products or publications, or inclusion of an advertisement in this article, neither constitutes nor implies a guarantee, endorsement, or support of claims made of that product, publication, or service.

Copyright © 2007, INFORMS

Please scroll down for article—it is on subsequent pages

With 12,500 members from nearly 90 countries, INFORMS is the largest international association of operations research (O.R.) and analytics professionals and students. INFORMS provides unique networking and learning opportunities for individual professionals, and organizations of all types and sizes, to better understand and use O.R. and analytics tools and methods to transform strategic visions and achieve better outcomes.
For more information on INFORMS, its publications, membership, or meetings visit http://www.informs.org

TUTORIALS IN
OPERATIONS RESEARCH
INFORMS 2007

© 2007 INFORMS | ISBN 13 978-1-877640-22-3
DOI 10.1287/educ.1073.0031

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

# Safe Scheduling

***Kenneth R. Baker***
Tuck School, Dartmouth College, Hanover, New Hampshire 03755,
kenneth.r.baker@dartmouth.edu

***Dan Trietsch***
College of Engineering, American University of Armenia, Yerevan, Armenia,
dan.trietsch@gmail.com

**Abstract** Safe scheduling augments the traditional approach to sequencing and scheduling models by explicitly recognizing safety times. In this tutorial, we give a brief review of the basic results in scheduling theory, describe how these results have been extended to stochastic cases, and demonstrate how safe-scheduling models sharpen our ability to formulate stochastic problems. Although scheduling theory has historically been driven by optimization methods, we demonstrate how stochastic scheduling problems can be approached with a powerful spreadsheet-based heuristic procedure.

**Keywords** sequencing; stochastic scheduling; safety time

---

Safe scheduling takes a new approach to stochastic problems in scheduling by explicitly recognizing safety times. In this tutorial, we discuss the basic principles and results of safe scheduling. We begin with a brief historical review of deterministic scheduling theory. From that starting point, we describe how the theory expanded to stochastic models. Finally, we show how safe-scheduling models fit into the picture and improve our ability to address stochastic scheduling problems.

## 1. Background: The Basic Deterministic Model

In scheduling theory, generic models are built to represent practical scheduling problems, or at least abstractions of practical problems, and those models are analyzed in an attempt to discover general rules and principles of effective scheduling. The most basic and most important model in scheduling theory is the deterministic, single-machine model. It is based on the following set of assumptions.

A1. At time zero, $n$ single-operation jobs (or tasks) are available for processing.
A2. The machine (or resource) can process at most one job at a time.
A3. Setup times for the jobs are independent of job sequence and are included in processing times.
A4. Job descriptors are deterministic and known in advance.
A5. Machines are continuously available (no breakdowns occur).
A6. Machines are never kept idle while work is waiting.
A7. Once an operation begins, it proceeds without interruption.

For the basic model, we may not need to impose the last two assumptions. In most deterministic, single-machine models, it is possible to find optimal solutions without inserted idle time, as in A6, or job preemption, as in A7. Therefore, A6 and A7 emerge as properties of optimal solutions rather than as assumptions that make the model tractable. In more complex models, however, these two properties may not characterize optimal solutions, so it is helpful to keep them visible.

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

To formalize the elements of the basic model, let $p_j$ denote the time required to process job $j$. (In some cases, we might also want to specify the job's due date, $d_j$, or the job's weighting factor, $w_j$.) When we construct a schedule, consisting of a sequence of the jobs, we effectively determine the completion time of each job, $C_j$. Next, we want to convert the set of completion times into a performance measure for the schedule.

A performance measure is a one-dimensional summary that aggregates the results for the various jobs. For example, suppose we define the *flowtime* $F_j$ as the length of time that job $j$ is in the system until it is fully processed. In the basic model, of course, we have $F_j = C_j$, but in other models the two measures could differ. A common way to aggregate over all the jobs is to take the total or the average; thus, we are interested in minimizing the total of the flowtimes, or $F = \sum F_j$. We refer to this as the $F$-problem. When we want to distinguish importance weights, $w_j$, among the various jobs, we can generalize the total flowtime measure to the *total weighted flowtime* measure, or $\sum w_j F_j$. This variation of the performance measure gives rise to what we call the $F_w$-problem.

When due dates, $d_j$, are given and we are concerned about due-date performance, we begin by measuring the tardiness of job $j$, $T_j = \max\{0, C_j - d_j\}$. Tardiness measures the amount by which a job's due date is missed, so one alternative for quantifying due-date performance is the total tardiness measure or $\sum T_j$. Alternatively, if there is a good chance that all jobs can be completed on time, or if we are mainly concerned about the worst-case outcome, we can adopt the *maximum tardiness* measure, or $T_{\max} = \max\{T_j\}$, and try to sequence the jobs so that its value is zero.

Finally, when we are mainly concerned about the efficiency of the schedule, we might want to complete all jobs as soon as possible, and so adopt the *makespan* measure, or $C_{\max} = \max\{C_j\}$. However, in the basic model, this performance measure does not help us to distinguish among schedules, because all sequences have the same makespan. The makespan measure becomes challenging for more complicated models.

Although scheduling problems can be classified in different ways, one of the most useful classifications is based on the scheduling objective. Three main objectives are prominent in scheduling theory, just as they are in scheduling practice:

- Turnaround
- Due-date performance
- Throughput

In scheduling models, these objectives are represented by such quantitative performance measures as $\sum w_j F_j$, $T_{\max}$, and $C_{\max}$, and the models with these objectives are analyzed to discover insights and relationships that might inform practical scheduling decisions. When scheduling theorists discuss the origins of their field, they usually point to three seminal papers that introduced these three types of criteria. Smith [11] addressed the single-machine sequencing problem with the objective of minimizing $\sum w_j F_j$. He demonstrated the following property.

**Theorem 1.** *To minimize $\sum w_j F_j$ in the basic single-machine model, sequence the jobs in nondecreasing order of $p_j/w_j$. This is often called the shortest weighted processing time ($SWPT$) sequence. If all weights are identical, this result dictates processing the jobs in shortest-first order, a procedure that generates the so-called shortest processing time ($SPT$) sequence.*

Jackson [5] dealt with the same model and the $T_{\max}$ objective. He demonstrated the following result.

**Theorem 2.** *To minimize $T_{\max}$ in the basic single-machine model, sequence the jobs in nondecreasing order of $d_j$. This is often called the earliest due date ($EDD$) sequence.*

Johnson [6] analyzed a two-machine model with the objective of maximizing throughput. Specifically, for a given set of tasks, throughput is maximized when all jobs are completed in the shortest possible time. Johnson derived the following rule for sequencing the jobs.

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

**Theorem 3.** *To minimize $C_{\max}$ in the two-machine flowshop (where all jobs must travel first to machine A and then to machine B), first classify the jobs according to whether the processing time on machine A is smaller than the time on machine B. Then, sequence the jobs with smaller times on A in nondecreasing order of processing times on machine A, followed by the jobs with smaller times on B in nonincreasing order of processing time on machine B.*

Theorems 1–3 represent key results in the seminal work done more than 50 years ago. Subsequent research in scheduling theory led to additional results of interest, which we highlight next.

Let $U_j$ denote a binary measure that is 1 if job $j$ is tardy and 0 otherwise. Consider the problem of minimizing the number of tardy jobs, $U = \sum U_j$, which we refer to as the $U$-problem. Moore [8] described a relatively simple solution algorithm that constructs two sets of jobs—an early set $A$, which may be sequenced according to EDD, and a late set $R$, which may be sequenced in any order. Moore's algorithm proceeds as follows.

1. Place all jobs in $A$ and order the jobs according to EDD.
2. If no jobs are tardy, then stop. The sequence is optimal.
3. Consider the jobs in sequence up to and including the first tardy job. Among the jobs in this initial sequence, remove the longest from $A$ and place it in $R$. Return to Step 2.

At the end of this procedure, the jobs in $A$ will remain in EDD order and complete by their due dates. The jobs in $R$ may be sequenced in any order because they will be tardy.

Several research efforts were devoted to the problem of minimizing total tardiness, $T = \sum T_j$, which we refer to as the $T$-problem. Although this sequencing problem exhibits special structure, and many special cases can be solved readily, the problem is NP-hard in general. Over the years, many specialized algorithms have been developed for the $T$-problem, and the state of the art now reflects methods that can solve problems with as many as a few hundred jobs (Szwarc et al. [13]).

The advent of just-in-time production generated interest in performance measures based on both earliness and tardiness. A job's earliness is defined as $E_j = \max\{0, d_j - C_j\}$, which becomes a symmetric complement of the job's tardiness. The E/T-problem calls for minimizing the sum of earliness costs and tardiness costs. For example, let $\alpha_j$ denote the unit penalty for earliness and let $\beta_j$ denote the unit penalty for tardiness. Then the E/T objective function takes the form of the sum $\sum(\alpha_j E_j + \beta_j T_j)$. There are many special cases of the E/T-problem, depending on whether the due dates are different and whether the earliness and tardiness penalties differ, but the general E/T-problem is NP-hard.

The E/T-problem differs in an important way from the other problems we have mentioned. The performance measures $\sum w_j F_j$, $T_{\max}$, $\sum U_j$, and $\sum T_j$ are all *regular* measures. For a regular measure to improve, at least one job must complete earlier. Stated another way, if we modify a schedule by delaying one job's completion time without reducing some other completion time, then the measure cannot get better. Most familiar performance measures are regular measures. Single-machine models with regular measures have optimal solutions without inserted idle time or preemption. For that class of problems, therefore, assumptions A6 and A7 are redundant. The E/T-problem, by contrast, involves a nonregular measure. As a result, we may sometimes impose assumptions A6 and A7, but in doing so, we may be excluding some desirable scheduling possibilities. Nevertheless, assumptions A6 and A7 are very practical—schedules and schedulers adhere to these assumptions in the real world—so those assumptions are usually reasonable in the analysis of scheduling problems.

### 1.1. A Spreadsheet-Based Heuristic Solution Approach

Difficult problems such as the E/T-problem or the $T$-problem generally require specialized algorithms and codes for finding optimal solutions to anything more than modest-sized problems. Nevertheless, good heuristic procedures are available for solving these problems. In

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

TABLE 1. Spreadsheet layout for a 20-job tardiness problem.

| Sequencing model | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Single-machine tardiness example | | | | | | | | | | | | | | | | | | | | |
| Data | | | | | | | | | | | | | | | | | | | | |
| $j$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| $p_j$ | 60 | 71 | 71 | 76 | 81 | 82 | 93 | 104 | 108 | 108 | 108 | 109 | 113 | 115 | 116 | 118 | 118 | 120 | 122 | 145 |
| $d_j$ | 404 | 394 | 534 | 308 | 778 | 917 | 482 | 472 | 702 | 803 | 1,142 | 1,115 | 811 | 1,191 | 672 | 1,139 | 1,329 | 710 | 534 | 591 |
| Solution | | | | | | | | | | | | | | | | | | | | |
| 5,433 | | | | | | | | | | | | | | | | | | | | |
| Decisions and calculations | | | | | | | | | | | | | | | | | | | | |
| Pos | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| $j$ | 2 | 7 | 4 | 1 | 8 | 3 | 19 | 9 | 5 | 10 | 6 | 13 | 11 | 12 | 14 | 15 | 16 | 17 | 18 | 20 |
| $p_j$ | 71 | 93 | 76 | 60 | 104 | 71 | 122 | 108 | 81 | 108 | 82 | 113 | 108 | 109 | 115 | 116 | 118 | 118 | 120 | 145 |
| $C_j$ | 71 | 164 | 240 | 300 | 404 | 475 | 597 | 705 | 786 | 894 | 976 | 1,089 | 1,197 | 1,306 | 1,421 | 1,537 | 1,655 | 1,773 | 1,893 | 2,038 |
| $d_j$ | 394 | 482 | 308 | 404 | 472 | 534 | 534 | 702 | 778 | 803 | 917 | 811 | 1,142 | 1,115 | 1,191 | 672 | 1,139 | 1,329 | 710 | 591 |
| $T_j$ | 0 | 0 | 0 | 0 | 0 | 0 | 63 | 3 | 8 | 91 | 59 | 278 | 55 | 191 | 230 | 865 | 516 | 444 | 1,183 | 1,447 |

what follows, we show how a generic heuristic procedure can be implemented on a spreadsheet for the purposes of solving these kinds of difficult problems. We use the $T$-problem as an example.

Imagine that we were given a single-machine sequencing problem and asked to verify the objective function value corresponding to a particular solution. How would we organize the calculations? The given information can be displayed as a table of job numbers, processing times and due dates, one type of data in each of three rows. The sequence can be displayed in another row. For each position in sequence, we would note the job assigned to that position and its processing time, compute its completion time, compare that with its due date, and finally, calculate its tardiness. The complete layout for an example is illustrated in the spreadsheet of Table 1.

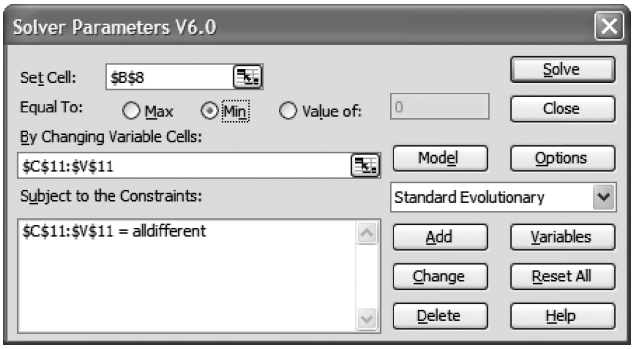
In the spreadsheet, rows 4–6 contain the given data that describe the problem. Rows 10–15 contain the calculations required to evaluate the objective function, and the value of the objective function itself is shown in row 8. Note, for example, that the table of calculations contains some entries that are drawn from the table of given information. In a spreadsheet context, these entries are table lookups and are obtained via Microsoft Excel formulas. The job completion times are easily calculated from left to right with a cumulative sum. Finally, the tardiness values are calculated from the formula that defines tardiness, $\max\{0, \textit{Completion} - \textit{due date}\}$, and these are totalled in the performance measure cell.

The optimization problem corresponding to our example is to choose the sequence in row 11 to minimize the value in the performance measure cell in row 8. An effective way of solving such a problem in Excel is to implement the evolutionary algorithm in Premium Solver. (Premium Solver is an upgraded version of the solver that comes with Excel and is available from Frontline Systems, http://www.solver.com. An educational version is common in engineering and business curricula because it accompanies more than a dozen textbooks in widespread use. Premium Solver contains four different algorithms, one of which is Evolutionary Solver, which is an advanced genetic algorithm suitable, among many other things, for sequencing problems.)

When we invoke Premium Solver, we encounter the Solver window shown in Figure 1. We select Evolutionary Solver from the pulldown menu, and the remaining entries specify (1) a cell containing the objective function, (2) a range of cells containing the decisions—in this case the assigned positions in sequence, and (3) any constraints on the selection of the decision variables. Here, the one constraint takes the form of an *all-different* constraint. This requirement forces the positions in the decision variable cells to comprise a permutation of the job indices. In other words, the decision cells must correspond to a feasible sequence.

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

FIGURE 1. Solver parameters window.




Evolutionary Solver searches for the best solution it can find, and its effectiveness is influenced by several user-determined parameters that are specified after clicking the Options button in the Solver window. The most important of these parameters set the stopping and convergence conditions that control the termination of the search. A good generic set of parameters would be the following:

Population size = 100
Mutation rate = 7.5% (the default value)
Convergence = 0.01%
Tolerance = 0
Maximum time without improvement = 15 seconds
Maximum search time = 60 seconds.

Other parameters (the number of feasible solutions, the number of subproblems, and the number of iterations) should be set to large numbers so as not to impede the search.

With these settings, the search will continue until one of the stopping conditions is met:

The search has been underway for 60 seconds;
No improvement has been encountered in the last 15 seconds;
99% of the 100 best solutions found are within 0.01% of each other.

The time limits can be adjusted according to the user's patience, but we have found that runs of roughly one minute produce good results for sequencing problems. In fact, optimal or near-optimal solutions are usually found in far less time.

In our 20-job example, a run of Evolutionary Solver produces the schedule shown in Table 1, with a total tardiness of 5,433. (This turns out to be an optimal value.)

## 2. Traditional Models in Stochastic Scheduling

Practical scheduling insights can be gleaned from relaxing any of the assumptions in the basic model and analyzing the problem that arises. Thus, for example, if we relax A1 and allow jobs to arrive over time, we confront the dynamic version of the basic single-machine model. If we relax A3 and allow changeover times that depend on job sequence, we confront the sequence-dependent setup version of the basic model. Each relaxation of this type produces a class of interesting scheduling problems, but space does not permit us to summarize the work on all of those models. For our purposes, we focus on the relaxation of A4, the assumption of deterministic job parameters.

When processing times are random, the resulting problem is called a *stochastic scheduling problem*. By relaxing A4, we permit due dates and job weights to be uncertain as well as processing times, but such models have limited practical significance. The most practical version of the model contains probabilistic processing times but treats due dates and weights as deterministic. As a consequence, the EDD sequence is well defined (except for ties). In

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

contrast, the SPT sequence is not well defined, because processing times are not known in advance. However, it is still possible to order jobs by nondecreasing *expected* processing times. This sequence is known as Shortest Expected Processing Time (SEPT). Similarly, the Shortest Weighted Expected Processing Time sequence (SWEPT) is also well defined.

Historically, research on stochastic scheduling has focused on the same performance measures considered in deterministic scheduling ($F$, $T_{\max}$, $U$, $T$, etc.) and has sought the minimization of their expected values. Thus, typical stochastic models aim to minimize $\mathrm{E}(F)$, $\mathrm{E}(T_{\max})$, $\mathrm{E}(U)$, $\mathrm{E}(T)$, etc. We refer to such models as *stochastic counterparts* of the corresponding deterministic problems. For example, the stochastic counterpart of the $F$-problem is a stochastic scheduling problem in which the objective function is the expected total flowtime, $\mathrm{E}(F)$. More generally, for deterministic models that seek to minimize the total cost or the maximum cost, stochastic counterparts seek to minimize the expected total cost or the expected maximum cost.

Advances in scheduling theory have provided solutions to some of the prominent stochastic counterparts, but such results are often difficult to obtain. Before exploring those results, we discuss a general way to formulate stochastic sequencing problems (using spreadsheet models) along with a general heuristic procedure for obtaining solutions.

In formulating a stochastic model, the key feature is a description of the distributions of processing times. Imagine a table in which each row corresponds to a scenario and each column corresponds to a job. The body of the table contains the processing time of job $j$ in scenario $s$. In Table 2, we show an example with four scenarios and five jobs. Along the bottom of the table, we show the average processing time for each job.

We can interpret this table in two ways. First, we can think of the scenarios as states of nature. When the scenarios form an exhaustive set of states, the columns represent the population of processing time outcomes for each job. Based on those values, we could calculate the mean and variance of processing times as well as the mean and variance of virtually any measure of performance. These calculations would provide us with accurate theoretical values.

Second, we can think of the scenarios as observations drawn from the population of potential outcomes. In this situation, the columns represent samples from processing-time distributions. Based on those values, we could calculate sample averages and sample standard deviations, and we could construct estimates of any measure of performance. Moreover, statistical procedures would allow us to estimate the precision in those estimates. In this *sample-based* approach, we usually find that samples of size 1,000 are sufficient for our purposes.

In Table 3, we show a spreadsheet formulation of a small stochastic scheduling problem. This example uses the probability distributions of Table 2. Because there are four scenarios, there are four rows in each portion of the table corresponding to the various probabilistic outcomes for completion time, tardiness, etc. For the purposes of illustration, we treat the scenarios in this example as if they represent an exhaustive set of probabilistic outcomes, so that the average values correspond to the true mean of the distributions involved. Suppose

Table 2. Scenarios for a stochastic scheduling problem.

| Example problem with random processing times | | | | | |
|---|---|---|---|---|---|
| Scenario | Job 1 | Job 2 | Job 3 | Job 4 | Job 5 |
| 1 | 10 | 8 | 12 | 15 | 20 |
| 2 | 12 | 12 | 14 | 17 | 20 |
| 3 | 14 | 16 | 16 | 20 | 20 |
| 4 | 16 | 20 | 22 | 24 | 24 |
| Average | 13 | 14 | 16 | 19 | 21 |

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

TABLE 3. Spreadsheet layout for the example problem.

| Example | $U$-problem | | | | | | |
|---|---|---|---|---|---|---|---|
| Data | Job $j$ | 1 | 2 | 3 | 4 | 5 | |
| State | $d_j$ | 22 | 37 | 28 | 75 | 54 | |
| 1 | $p_j$ | 10 | 8 | 12 | 15 | 20 | |
| 2 | | 12 | 12 | 14 | 17 | 20 | |
| 3 | | 14 | 16 | 16 | 20 | 20 | |
| 4 | | 16 | 20 | 22 | 24 | 24 | |
| | E($p_j$) | 13 | 14 | 16 | 19 | 21 | |
| | | | | | | | No. tardy |
| | Sequence | 1 | 3 | 2 | 5 | 4 | 2.50 |
| Processing times | 1 | 10 | 12 | 8 | 20 | 15 | |
| | 2 | 12 | 14 | 12 | 20 | 17 | |
| | 3 | 14 | 16 | 16 | 20 | 20 | |
| | 4 | 16 | 22 | 20 | 24 | 24 | |
| Completion times | 1 | 10 | 22 | 30 | 50 | 65 | |
| | 2 | 12 | 26 | 38 | 58 | 75 | |
| | 3 | 14 | 30 | 46 | 66 | 86 | |
| | 4 | 16 | 38 | 58 | 82 | 106 | |
| Tardiness | 1 | 0 | 0 | 0 | 0 | 0 | |
| | 2 | 0 | 0 | 1 | 4 | 0 | |
| | 3 | 0 | 2 | 9 | 12 | 11 | |
| | 4 | 0 | 10 | 21 | 28 | 31 | |
| | | | | | | | Total |
| Unit tardiness | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 2 | 0 | 0 | 1 | 1 | 0 | 2 |
| | 3 | 0 | 1 | 1 | 1 | 1 | 4 |
| | 4 | 0 | 1 | 1 | 1 | 1 | 4 |

that the objective function is E($U$), the expected number of tardy jobs. As shown in Table 3, the EDD sequence 1-3-2-5-4 generates a value of 2.50.

Again, we can employ Evolutionary Solver to find good solutions for this type of problem. The objective function cell (H12 in the table), and the decision cells (C12:G12) represent the heart of the optimization problem. The only constraint is the all different constraint on the decisions. A run of Evolutionary Solver generates a solution value of 1.50, produced by the sequence 1-2-5-4-3.

Evolutionary Solver gives us a powerful and flexible solution technique that can run effectively with up to 20 jobs and up to 1,000 scenarios. Although the technique is a heuristic method and cannot guarantee optimal solutions, it seems to be very effective in problems of this size.

## 2.1. Optimal Solutions for Stochastic Scheduling

Of course, we would prefer to work with solutions that are guaranteed to be optimal in stochastic scheduling problems, but it turns out that there are very few general cases in which we can do so. We now turn to a brief summary of the theoretical state of the art in stochastic scheduling. We focus on the measures addressed earlier: E($F_w$), E($T_{\max}$), E($U$), and E($T$), and we make no specific assumptions about the processing-time distributions.

Consider the stochastic counterpart of the $F_w$-problem. In other words, processing times are random, and the objective is to minimize the expected value of total weighted flowtime.

**Theorem 4 (**Rothkopf [10]**).** *To minimize* E($F_w$) *in the basic single-machine model, sequence the jobs according to SWEPT.*

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

Note that Theorem 4 does not say that total flowtime is *always* minimized by SWEPT—that is, in every scenario. Rather, SWEPT minimizes the total *on average*. In the scenarios of Table 2, sequences other than SWEPT are sometimes optimal. In the first scenario, for example, job 2 should precede job 1, and in the fourth scenario, job 3 should precede job 2. But such observations can be made only in hindsight, and we cannot rely on hindsight for scheduling decisions. Therefore, SWEPT is the best we can do ex ante, before the realizations of the processing times are revealed.

In the example of Table 2, all weights are equal to 1, so the objective function is total expected flowtime. By Theorem 4, the optimal sequence is SEPT (1–2–3–4–5), which generates an expected total flowtime of 228. Suppose instead that we substitute the expected processing times for the random processing times and then solve this deterministic problem, thus creating what is known as the *deterministic counterpart* of the stochastic problem. We then find that the same sequence (1–2–3–4–5) is optimal, and that its total flowtime is 228. In general, the optimal sequence and the optimal value of the performance measure in the deterministic counterpart always match those of the stochastic $F_w$-problem.

Next, consider the stochastic counterpart of the $T_{\max}$-problem. The solution to this problem is well known, and again, it echoes the deterministic solution.

**Theorem 5 (**Crabill and Maxwell [4]**).** *To minimize* $\mathrm{E}(T_{\max})$ *in the basic single-machine model, sequence the jobs according to EDD.*

But an interesting feature arises with respect to the deterministic counterpart. When we suppress the random processing times and substitute expected values instead, the optimal sequence is the same (EDD does not depend on the processing times). However, the value of the objective function may not be the same for the deterministic counterpart as it is for the solution of the stochastic problem. Our example serves as an illustration. Table 4 shows that the EDD sequence produces $\mathrm{E}(T_{\max}) = 11.75$. But if we substitute expected processing

TABLE 4. The EDD sequence in the example problem.

| Example | $T_{\max}$-problem | | | | | | |
|---|---|---|---|---|---|---|---|
| Data | Job $j$ | 1 | 2 | 3 | 4 | 5 | |
| State | $d_j$ | 22 | 37 | 28 | 75 | 54 | |
| 1 | $p_j$ | 10 | 8 | 12 | 15 | 20 | |
| 2 | | 12 | 12 | 14 | 17 | 20 | |
| 3 | | 14 | 16 | 16 | 20 | 20 | |
| 4 | | 16 | 20 | 22 | 24 | 24 | |
| | $\mathrm{E}(p_j)$ | 13 | 14 | 16 | 19 | 21 | |
| | | | | | | | $\mathrm{E}(T_{\max})$ |
| | Sequence | 1 | 3 | 2 | 5 | 4 | 11.75 |
| Processing times | 1 | 10 | 12 | 8 | 20 | 15 | |
| | 2 | 12 | 14 | 12 | 20 | 17 | |
| | 3 | 14 | 16 | 16 | 20 | 20 | |
| | 4 | 16 | 22 | 20 | 24 | 24 | |
| Completion times | 1 | 10 | 22 | 30 | 50 | 65 | |
| | 2 | 12 | 26 | 38 | 58 | 75 | |
| | 3 | 14 | 30 | 46 | 66 | 86 | |
| | 4 | 16 | 38 | 58 | 82 | 106 | |
| | | | | | | | $T_{\max}$ |
| Tardiness | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 2 | 0 | 0 | 1 | 4 | 0 | 4 |
| | 3 | 0 | 2 | 9 | 12 | 11 | 12 |
| | 4 | 0 | 10 | 21 | 28 | 31 | 31 |

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

times for random processing times and solve the deterministic problem that results, we find that the EDD sequence produces $T_{\max} = 10.00$. Thus, we begin to see that we cannot go back and forth seamlessly between a stochastic-scheduling problem and a deterministic simplification based on expected values.

The mechanism operating in this example is an instance of Jensen's Inequality:

$$\text{If } h(Z) \text{ is convex and } Z \text{ is random, then } \mathrm{E}[h(Z)] \geq h[\mathrm{E}(Z)].$$

For example, taking $Z$ to be a set of processing times and $h(Z)$ to be the maximum tardiness function, we obtain $\mathrm{E}[\max(T_j)] \geq T_{\max}$ in the deterministic counterpart. Therefore, we can conclude that, although it delivers the optimal sequence to the stochastic problem, the deterministic counterpart systematically underestimates the value of the objective function in the stochastic problem.

In such a case, we say that there is a nonnegative *Jensen gap* between the objective function of the deterministic counterpart and the objective of the stochastic problem. Jensen gaps are often positive, although they are zero for linear functions, such as $F_w$. To distinguish between piecewise linear convex functions (such as the max function) and linear ones, we define any convex function that is not linear as *meaningfully convex*. In models with meaningfully convex objective functions, we can find instances with strictly positive Jensen gaps. If we are interested in the optimal value of the objective function as well as the optimal sequence, the deterministic counterpart may be biased, due to the Jensen gap, even in otherwise well-solved problems.

When we consider the stochastic counterpart of the $U$-problem, things get a little more complicated. Although the deterministic counterpart can be solved efficiently using Moore's algorithm, the stochastic $U$-problem is essentially unsolved. In other words, the deterministic counterpart may not even reveal the optimal sequence to the stochastic problem. We return to this model later on.

Next, consider the stochastic $T$-problem. The deterministic counterpart has attracted a lot of attention over the years, and as mentioned earlier, the most recent algorithmic advances can solve versions of the problem with hundreds of jobs. However, there has been little progress on the stochastic problem. One reason is the difficulty of finding dominance conditions, which are often the key to effective performance of deterministic solution procedures. As a consequence, the branch-and-bound approaches that work well in the deterministic case have not been successful in the stochastic case. (Because the tardiness function is meaningfully convex, we should not be surprised to discover that a Jensen gap exists here, too.) The stochastic $T$-problem is therefore largely unsolved.

To reinforce our point, consider the problem of sequencing two jobs with stochastic processing times and with the objective of minimizing the expected total tardiness $\mathrm{E}(T)$.

| Job $j$ | 1 | 2 |
|---|---|---|
| $d_j$ | 2 | 3 |
| $\mathrm{E}(p_j)$ | 2.5 | 2.5 |

| State | Job $j$ | 1 | 2 | Probability |
|---|---|---|---|---|
| A | $p_j$ | 1 | 2 | 0.5 |
| B | $p_j$ | 4 | 3 | 0.5 |

We can easily enumerate the possible solutions to this problem. The sequence 1–2 generates $\mathrm{E}(T) = 4.5$, whereas the sequence 2–1 generates $\mathrm{E}(T) = 3.5$. Therefore, the latter sequence is optimal. However, instead of analyzing the stochastic problem, suppose we rely on the deterministic counterpart, in which the expected processing times are 2.5 for each job. When

we substitute this value for the processing times, we find that the minimum tardiness is 2.5, achieved by the sequence 1–2. Thus, the deterministic counterpart leads us to a suboptimal sequence as well as to a biased value for the objective function.

Thus, as we review the general results in stochastic scheduling, we find that the deterministic counterpart is not reliable. The bias inherent in such an approach has a potentially devastating consequence for practice. Over time, as organizations learn that their deterministic counterpart solutions underestimate actual performance, they sense the need for time buffers in the schedule. Lacking any explicit guidance for buffering schedules, practitioners tend to make pessimistic estimates of processing times. This approach is ultimately inefficient. For practical solutions of stochastic scheduling problems, we must turn to a different form of analysis. Heuristic solutions are available, using the sample-based technique described above, but optimization approaches require a new way of thinking about the stochastic model.

## 3. The Safe-Scheduling Approach

We believe the extensive literature on stochastic scheduling models has unfortunately focused too often on stochastic counterparts and overlooked the practical importance of safety time. As a result, traditional stochastic models fail to consider service levels in scheduling. We advocate an approach to the stochastic problem that recognizes safety time. As an analogy, imagine inventory analysts trying to build stochastic inventory models by relying only on deterministic analysis of average behavior and making no provisions for safety stock. Just as safety *stocks* are vital to practical inventory policies, safety *time* is vital to practical scheduling policies. However, the optimal determination of safety time has no counterpart in deterministic scheduling. *Safe scheduling* departs from the dominant paradigm in stochastic scheduling by considering safety time explicitly.

In stochastic inventory theory there are two general ways to determine safety stocks—by meeting service-level constraints or by minimizing the expected total costs due to overstocking and understocking. We use analogous approaches in safe scheduling. To incorporate service-level constraints, we replace the deterministic definition of "on time" by a stochastic one. Define the *service level* for job $j$ as $\Pr\{C_j \le d_j\}$, the probability that job $j$ completes by its due date. We sometimes denote this probability $SL_j$. Let $b_j$ denote a given target for the service level. Then the form of a *service-level constraint* for job $j$ is

$$SL_j = \Pr\{C_j \le d_j\} \ge b_j.$$

We say that job $j$ is *stochastically on time* if its service-level constraint is met; otherwise, the job is *stochastically tardy*. A complete sequence is called *stochastically feasible* if all jobs are stochastically on time. The use of service-level constraints is widely accepted in practice, especially when it is difficult to estimate the relevant costs, but it has seldom been exploited in the stochastic scheduling literature. An early exception was the observation by Banerjee [3] that the EDD sequence maximizes the minimum service level. (Equivalently, EDD minimizes the maximum probability that a job will be tardy.)

The alternative to imposing service-level constraints is to explicitly consider economic factors. If we can model the true economic costs of various outcomes with a total penalty function, we can then look for a schedule that minimizes the expected total penalty. Because the penalty function includes the cost of creating a buffer as well as the cost of failing to meet due dates, the minimization of the penalty function automatically dictates optimal safety. Unfortunately, the difficulty with this approach lies in the practical problems of acquiring good cost data, especially when some cost elements are subjective. When costs are hard to identify, we fall back on the service-level approach.

Both alternatives allow us to incorporate considerations of safety, but they do not specify the scheduling problem completely. Thus, there are two major formulations of the safe-scheduling problem. One formulation treats due dates (and possibly release dates) as given

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

FIGURE 2. Classification of safe-scheduling examples.

| | **Accept/reject decision** | **Due-date decision** |
|---|---|---|
| **Service-level constraints** | $U$-problem with service-level constraints | Due-date tightness and tardiness trade-off |
| **Costs for all outcomes** | $U$-problem with three rewards | Optimizing earliness and tardiness penalties |

and determines which jobs to accept, which to reject, and how to sequence the accepted jobs. We illustrate this formulation in the context of the $U$-problem. The other formulation treats due dates as decisions and adjusts these parameters in the process of minimizing expected total penalty. Typically, when due dates are decisions, we assume that all jobs will be processed, but we could accommodate the accept/reject decision as well. In either case, optimal safety time is a by-product of the analysis. We illustrate the second formulation in the context of the E/T-problem.

Considering the two approaches to acknowledging safety in conjunction with the two problem formulations, we obtain four combinations, which we discuss in the following subsections. Figure 2 depicts the main examples.

Our illustrations are representative of problems that could be addressed by ongoing research, but many other problems could be pursued from the lens of safe scheduling.

### 3.1. Service-Level Approach to the $U$-Problem

Earlier, we presented the deterministic $U$-problem as one of postponing the jobs assigned to set $R$ so that jobs in set $A$ can be completed on time. Because jobs in $R$ are late, they can be postponed indefinitely without altering the number of late jobs. Thus, we may equivalently consider jobs in $R$ to be rejected, and Moore's algorithm can be viewed as a procedure for minimizing the number of rejected jobs. Similarly, we treat the stochastic $U$-problem as a problem of accepting or rejecting jobs. From the safe-scheduling perspective, the stochastic $U$-problem comes with service-level constraints and calls for minimizing $|R|$ subject to stochastic feasibility of the accepted jobs. The general version of this problem is known to be NP-hard (Kise and Ibaraki [7]), and no computationally practical solution procedure has been developed.

One complication that arises in the problem relates to the sequencing of the on-time jobs—those in set $A$. In Moore's algorithm, we can rely on EDD sequencing of set $A$, but the optimality of EDD does not generalize to the stochastic case, as the following example demonstrates.

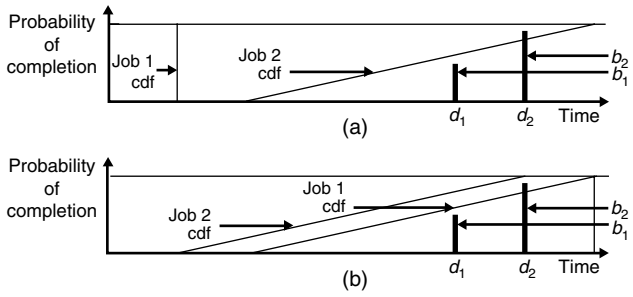
| Job $j$ | 1 | 2 |
|---|---|---|
| $d_j$ | 5.0 | 6.0 |
| $b_j$ | 0.5 | 0.9 |
| $\mathrm{E}(p_j)$ | 1 | 3.5 |

Job 1 has a deterministic processing time, but Job 2 has a processing time that follows a uniform distribution on the interval $(1, 6)$.

Figure 3(a) describes the situation for the EDD sequence. Two vertical segments with height 1, occurring at times 0 and 1, mark the start and finish of job 1. The uncertainty in $C_2$ is represented by a linear cdf. Finally, the two service-level requirements are represented by vertical bars with heights $b_j$ at the respective due dates. Clearly, Job 1 exceeds its service-level requirement because its cdf reaches a height of 1 prior to $d_1$. But Job 2 fails to meet its requirement because by $d_2$ its cdf does not reach the height of $b_2$. Figure 3(b) demonstrates,

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

FIGURE 3. Graph for Example: (a) sequence 1-2, (b) sequence 2-1.




however, that if we interchange the jobs and sequence against EDD order, then both service-level constraints are met. The example shows that, contrary to the deterministic case, a stochastically feasible set of jobs is not necessarily feasible in EDD sequence.

Figure 3(a) is an example of a predictive Gantt chart. In a regular Gantt chart, jobs are always depicted as rectangles. Job 1 is depicted this way in Figure 3(a) because there is no uncertainty about its start and finish times. The height of its rectangle is equal to 1, and we can interpret the vertical line at time 1 as the cdf of $C_1$. The same job appears in Figure 3(b), but in that case its start time and finish time are uncertain and represented by cdfs. The area between the start cdf and the completion cdf of job 1 is the same in both figures and equals the expected processing time of the job. The horizontal line at the top of the figure can be interpreted as part of the cdf of the start time of the activity. As the figure demonstrates, a predictive Gantt chart shows the probability of completion as a function of time (because it involves cdfs) and can also be used to check whether particular service levels are met.

As we shall see, an important condition for solving this problem is stochastic ordering. The processing times of two jobs, $j$ and $k$, are said to be *stochastically ordered* if $\Pr\{p_j \le t\} \ge \Pr\{p_k \le t\}$ for any value of $t$. This does not mean that job $j$ is always shorter than job $k$. Rather, for any duration $t$, the completion of $j$ by this amount of time is more likely than the completion of $k$ by this amount of time, and this relation holds for any duration $t$. In our example, we have a somewhat extreme case of stochastic ordering, because job 2 is always at least as long as job 1. Thus, our example illustrates that the stochastic infeasibility of EDD may occur even if processing times are stochastically ordered.

To make progress for our formulation of the $U$-problem, we need a procedure for determining whether a feasible sequence exists for any subset of accepted jobs. As it happens, a relatively simple procedure is available (van den Akker and Hoogeveen [14]). This *feasibility check* examines whether any job would satisfy its service-level constraint if scheduled last. (The probability distribution of the last job's completion time can be determined without knowing the full job sequence because the last completion time does not depend on the job sequence.) Any such job may be scheduled last and removed from further consideration. The procedure is then repeated for the remaining jobs. Because the procedure is constructive, it builds a sequence from back to front and yields a feasible sequence whenever the set of jobs is feasible.

Next, we imbed the feasibility check in the logic for accepting or rejecting jobs. If the jobs are stochastically ordered, we can use the logic of Moore's algorithm, with the feasibility check in place of EDD sequencing. The steps are as follows.

*SEPT-Based Feasibility Check*

1. Sequence the jobs by SEPT (ties may be broken arbitrarily) and place all jobs in the unresolved set (sets $A$ and $R$ are empty).

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

TABLE 5. Data for a five-job example.

| | Job $j$ | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| | $d_j$ | 7.8 | 7.5 | 17 | 20 | 12 |
| State | $b_j$ (%) | 90 | 60 | 50 | 80 | 60 |
| 1 | $p_j$ | 2.6 | 3.5 | 3.8 | 4.5 | 6.4 |
| 2 | | 2.8 | 3.9 | 4.4 | 5.5 | 6.6 |
| 3 | | 3.2 | 4.1 | 5.6 | 6.5 | 7.4 |
| 4 | | 3.4 | 4.5 | 6.2 | 7.5 | 7.6 |
| | $\mathrm{E}(p_j)$ | 3.0 | 4.0 | 5.0 | 6.0 | 7.0 |

2. Tentatively add the first unresolved job to $A$ and apply the feasibility check. If the result is feasible, record the sequence and add the job to $A$ permanently. Otherwise, add the job to $R$.

3. If the unresolved set is not empty, return to Step 2. Otherwise, stop. The last recorded sequence of the jobs in $A$ is optimal.

To illustrate the application of the algorithm, we consider another five-job example, with service-level targets $b_j$ as specified in Table 5.

When we consider the set $\{1\}$, we find that job 1 meets its target service level because it is certain to complete by its due date ($d_1 = 7.8$), so in Step 2, we add job 1 to set $A$. When we consider the set $\{1, 2\}$, we find that job 1 cannot meet its target if it follows job 2, but job 2 meets its target if it follows job 1. Therefore, in Step 2, we add job 2 to $A$. Next, job 3 is feasible in the third position and is added to $A$. When we try to add job 4, it is not feasible in the fourth position, but the initial sequence 1-2-4-3 is feasible, so we add job 4 to $A$. Lastly, adding job 5 leads to infeasibility—no job can be feasibly scheduled in the last position. Therefore, job 5 is rejected, and the optimal solution is $|R| = 1$.

Note what happens if we use Moore's algorithm, which relies on EDD sequencing of the jobs in set $A$. The EDD sequence is 2-1-5-3-4. jobs 2 and 1 are added to $A$, but job 5 cannot be feasibly added next, so it is assigned to set $R$. Then job 3 is added, but job 4 cannot be feasibly added after job 3, so job 4 is assigned to $R$, leaving only three jobs in $A$. As shown above, the optimal number of jobs in $A$ is 4.

We are left with the question whether special cases exist in which we can rely on the EDD sequence for the stochastically on-time jobs. The answer is affirmative. For example, suppose that all jobs have the same service-level target—that is, $b_j = b$. In this case, a set of jobs is feasible if each one has a service level no lower than $b$. But as we mentioned earlier, the EDD sequence maximizes the minimum service level, so it will generate a sequence, if one exists, in which each service level meets or exceeds $b$. In other words, if a feasible sequence exists, the EDD sequence will be feasible.

We can also generalize the case of equal service levels. For any two jobs $i$ and $k$, if $d_i \le d_k$ and $b_i \ge b_k$, then these parameters are said to be *agreeable*. When (1) every pair of jobs has agreeable due dates and service-level targets and (2) the processing times are stochastically ordered, the EDD sequence generates a feasible sequence if one exists. Thus, when those two conditions hold, Moore's algorithm can be used to minimize the number of stochastically tardy jobs.

To summarize, the problem of minimizing the number of stochastically tardy jobs is NP-hard in general. If we know that the processing times are stochastically ordered, then we can find solutions with an algorithm based on a feasibility check, and if we also know that due dates and service-level targets are agreeable, then we can find solutions with Moore's algorithm, which is slightly simpler. Using a safe-scheduling approach—in this case, making service levels part of the problem—we can at least find efficient solutions to special cases of a problem that is otherwise quite difficult to solve.

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

### 3.2. Economic Approach to the $U$-Problem

The economic approach involves specifying the costs for various outcomes and then minimizing an objective corresponding to expected total penalty. If we can specify costs, then the minimization of total penalty may reflect reality better than a traditional summary measure such as $U$.

In the stochastic case, there is a conceptual difference between a job that is rejected by design and a job that is tardy by chance. That is, we may accept a job with the intention of completing it on time, but the stochastic nature of its processing time (and those of earlier jobs) may result in tardiness in spite of our intention. This structure leads us to a model that distinguishes three types of outcomes rather than two. For every job that is completed early or on time, we obtain a reward of $R_E$, a tardy job generates a reward of $R_T < R_E$, and the reward for rejecting a job is $R_R$. We require

$$R_E > R_R > R_T.$$

Here, our objective is to maximize the expected total reward. The assumption in the ordering of the three reward parameters is economically sound: If $R_R$ were not strictly higher than $R_T$, we would gain nothing by rejecting a job—that is, we would want to process all jobs even if they are in $R$ because the reward would be at least as good as rejecting them. Similarly, if $R_R$ were not strictly lower than $R_E$, we would reject all jobs immediately. By subtracting $R_R$ from all rewards, we change the total reward by a constant, but the optimal sequence does not change. Therefore, without loss of generality, we may assume that $R_E > 0$, $R_R = 0$, and $R_T < 0$. After this adjustment, any optimal solution must have a nonnegative expected reward because rejecting all jobs would guarantee a total reward of zero.

To facilitate the incorporation of explicit rejection decisions into our model, we turn to the service-level approach and derive a constraint that all accepted jobs must satisfy. As introduced earlier, $SL_j$ denotes the probability that job $j$ is on time. Then the expected reward $\mathrm{E}(R_j)$ for job $j$ when it is accepted becomes:

$$\mathrm{E}(R_j) = R_E SL_j + R_T(1 - SL_j).$$

Note that this contribution is not positive unless

$$\mathrm{E}(R_j) = R_E SL_j + R_T(1 - SL_j) > 0$$

or

$$SL_j > -R_T/(R_E - R_T).$$

The right-hand side of this inequality serves as a legitimate probability because $R_T < 0$. Furthermore, rejecting any job can only help reduce the tardiness of other jobs, so the optimal solution cannot call for accepting any job whose expected reward is negative. Therefore, if the inequality is violated for any accepted job at any sequence position, that sequence cannot be optimal. The inequality is necessary for each job, but not sufficient for optimality. In practice, these constraints are not likely to be tight except for jobs that are sequenced at the end of the schedule. In other words, early in the sequence it may even be suboptimal to accept a job whose service level barely satisfies the inequality. Although accepting such a job would generate a positive expected reward for the job itself, the consequence may be to reduce the service levels of several other jobs and ultimately lead to a net loss.

The economic approach to the $U$-problem is new, and no optimizing algorithm has yet been developed. Nevertheless, the problem can be attacked using the spreadsheet-based heuristic algorithm demonstrated earlier. Table 6 shows a spreadsheet layout and an initial solution for the five-job example introduced in the previous subsection with rewards of $R_E = 5$ and $R_T = -10$. Once again, a good heuristic solution can be obtained with Evolutionary Solver, which generates an expected reward of 15 in this example.

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

TABLE 6. Spreadsheet solution to the example problem.

| Example | | Expected rewards with A/R decisions | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Data | Job $j$ | 1 | 2 | 3 | 4 | 5 | $R_E$ | 5 |
| Scenario | $d_j$ | 22 | 37 | 28 | 75 | 54 | $R_T$ | −10 |
| GG | $p_j$ | 10 | 8 | 12 | 15 | 20 | | |
| GB | | 12 | 12 | 14 | 17 | 20 | | |
| BG | | 14 | 16 | 16 | 20 | 20 | | |
| BB | | 16 | 20 | 22 | 24 | 24 | | |
| | E($p_j$) | 13.0 | 14.0 | 16.0 | 19.0 | 21.0 | | |
| | Sequence | 5 | 4 | 3 | 2 | 1 | E(Reward) | |
| | Accept | 1 | 1 | 1 | 0 | 1 | −10.00 | |
| Processing times | | | | | | | | |
| GG | | 20 | 15 | 12 | 0 | 10 | | |
| GB | | 20 | 17 | 14 | 0 | 12 | | |
| BG | | 20 | 20 | 16 | 0 | 14 | | |
| BB | | 24 | 24 | 22 | 0 | 16 | | |
| Completion times | | | | | | | | |
| GG | | 20.0 | 35.0 | 47.0 | 47.0 | 57.0 | | |
| GB | | 20.0 | 37.0 | 51.0 | 51.0 | 63.0 | | |
| BG | | 20.0 | 40.0 | 56.0 | 56.0 | 70.0 | | |
| BB | | 24.0 | 48.0 | 70.0 | 70.0 | 86.0 | | |
| Rewards | | | | | | | Total | |
| GG | | 5.0 | 5.0 | −10.0 | 0.0 | −10.0 | −10.00 | |
| GB | | 5.0 | 5.0 | −10.0 | 0.0 | −10.0 | −10.00 | |
| BG | | 5.0 | 5.0 | −10.0 | 0.0 | −10.0 | −10.00 | |
| BB | | 5.0 | 5.0 | −10.0 | 0.0 | −10.0 | −10.00 | |

### 3.3. Due Dates as Decisions in the Tightness-Tardiness Trade-off

Due dates are considered given information in the basic model. However, some research (e.g., Baker and Bertrand [1]) has also explored models with due dates as decisions. This treatment is justified by the fact that due dates are sometimes negotiated with customers, so that the scheduling agent has some control over setting due dates. In addition, in multistage processes, due dates are often set internally by a control system, as a means of establishing progress guidelines while the jobs are in process. In what follows, we examine models that treat due dates as decisions.

In the first of these models, we are concerned about E($T$) as a performance measure. When due dates are controllable, an inherent trade-off occurs. We can make tardiness small (even zero) by choosing the due dates to be sufficiently loose. On the other hand, tight due dates are desirable because they impose some discipline on the progress of due-date-oriented jobs. Therefore, we consider an objective function that combines due-date tightness and tardiness performance. More specifically, the objective function takes the form

$$f(S) = \sum d_j + \gamma \sum \mathrm{E}(T_j),$$

where $S$ represents the set of scheduling decision variables, here including the due dates. Instead of finding the optimal due dates directly, we can solve the problem by determining the optimal values of the service-level targets $b_j$. Once those optimal targets are found, they dictate the optimal due dates. Note that the objective function is additive: Each job makes a separate contribution to each of the two sums. Changing $d_j$ affects these two contributions but not the contributions of any other job. Therefore, we can solve for the optimal due dates separately.

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

Because the effects of the due dates are separable, we can analyze the choice for job $j$. Suppose that we choose a due date of $d_j$, and then we observe a completion time of $C_j$. In retrospect, we can ask whether it would be desirable to have increased $d_j$ by 1 initially. A unit increase in $d_j$ would increase $\sum d_j$ by 1 and would reduce tardiness if the job finished late—that is, if $C_j > d_j$. Thus, the net effect on the objective function would be $1-\gamma \Pr\{C_j > d_j\}$. It follows that we should increase the due date as long as this expected incremental cost drops; that is, while

$$1-\gamma(\Pr\{C_j > d_j\}) \le 0.$$

Note that this inequality can never be satisfied unless $\gamma > 1$. For those values, we can equivalently write the required condition as increasing $d_j$ while:

$$1-\gamma(1-\Pr\{C_j \le d_j\}) \le 0.$$

Therefore, we should set $d_j$ equal to the smallest value for which this inequality fails and

$$\Pr\{C_j \le d_j\} \ge (\gamma-1)/\gamma.$$

This condition is a version of the *critical fractile rule* and justifies the following result.

**Theorem 6.** *Suppose the objective is to minimize $\sum d_j + \gamma \sum \mathrm{E}(T)$ for a given sequence. Then, if $\gamma \le 1$ we should set all due dates to 0. Otherwise, for job $j$, it is optimal to set $d_j$ equal to the smallest value that satisfies the condition*

$$SL_j = \Pr\{C_j \le d_j\} \ge (\gamma-1)/\gamma.$$

As an example, consider the following five-job problem. The processing times are independent and each is drawn from a normal distribution with the mean and standard deviation shown in the table below. Take the parameter $\gamma = 10$.

| Job $j$ | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| $\mathrm{E}(p_j)$ | 5 | 6 | 7 | 8 | 9 |
| $\sigma_j$ | 1.8 | 1.0 | 0.3 | 0.8 | 0.2 |

In this instance, we have $(\gamma-1)/\gamma = 0.9$, and the corresponding $z$-value in the normal table is $z = 1.282$. From this value we can calculate the due date for each of the five jobs, as shown in Table 7. We then calculate the mean tardiness for each of the jobs, using the algebra of normal distributions, and finally, the value of our objective function. In the example, the optimal value of this objective function is 112.851. In more detail, $\sum d_j = 109.118$ whereas $\mathrm{E}(\sum C_j) = 99$, and the difference is the combined safety time of 10.118. The expected tardiness cost is 3.733, for a total of 13.851 above $\mathrm{E}(\sum C_j)$. Without safety time, however, the expected tardiness cost grows to 31.487 and the total cost to 130.487. Thus, a plan for 10.118 units of safety time leads to a net savings of 17.635 in the objective function.

Theorem 6 demonstrates that when due dates are decisions, they can be set to produce optimal service levels. That choice produces an optimal level of safety time for each job, although the specific level depends on the other jobs in the schedule.

Finally, whereas the optimal sequence to minimize $\sum d_j + \gamma \mathrm{E}(T)$ is not always easy to find, the SEPT sequence—which is optimal for the deterministic counterpart—is often a good heuristic. Indeed, if processing times are stochastically ordered, then it is possible to show that SEPT is optimal.

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

TABLE 7. Spreadsheet layout for the tightness-tardiness trade-off.

| Example | $\sum d+\gamma\sum \mathrm{E}(T)$ | Sequence given *Normal case* | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Data | | | | | | | | |
| | Job $j$ | 1 | 2 | 3 | 4 | 5 | $\gamma$ | 10 |
| | $\mathrm{E}(p_j)$ | 5 | 6 | 7 | 8 | 9 | | |
| | $s_j$ | 1.8 | 1.0 | 0.3 | 0.8 | 0.2 | $(\gamma-1)/\gamma$ | 0.9 |
| | Sequence | 3 | 2 | 1 | 4 | 5 | | |
| | $\mathrm{E}(p_j)$ | 7 | 6 | 5 | 8 | 9 | | |
| | $s_j$ | 0.3 | 1.0 | 1.8 | 0.8 | 0.2 | $\sum d+\gamma\sum \mathrm{E}(T)$ | 112.851 |
| Calculations | | | | | | | | |
| | Variance | 0.090 | 1.000 | 3.240 | 0.640 | 0.040 | | |
| | Cumulative | 0.090 | 1.090 | 4.330 | 4.970 | 5.010 | | |
| | Sq. root | 0.300 | 1.044 | 2.081 | 2.229 | 2.238 | | |
| | Cum. mean | 7 | 13 | 18 | 26 | 35 | | |
| | $z$ value 1.282 | 1.282 | 1.282 | 1.282 | 1.282 | | | |
| | Normal pdf | 0.175 | 0.175 | 0.175 | 0.175 | 0.175 | | |
| | Due date | 7.385 | 14.338 | 20.668 | 28.858 | 37.870 | 109.118 | |
| | Mean Tardiness | 0.014 | 0.049 | 0.098 | 0.105 | 0.106 | 0.373 | |

### 3.4. Due Dates as Decisions in the Stochastic Makespan Problem

We next turn to a general economic approach in scheduling problems that treat due dates as decisions. We consider the stochastic version of the E/T-problem, and we begin with the special case of a common due date. The E/T-problem with a common due date arises when customers want their jobs completed at the same time. A more realistic application of a common due date arises when all jobs go to the same customer, feeding an assembly operation. In this case, the completion of the last job determines performance. The unit earliness cost $\alpha$ and the unit tardiness cost $\beta$ apply to the difference between the makespan ($C_{\max}$) and the due date ($d$). Thus, we let

$$f(S)=\alpha\max\{0,d-C_{\max}\}+\beta\max\{0,C_{\max}-d\}.$$

The deterministic version of this problem is trivial because the makespan is constant, and we can optimize $f(S)$ by setting $d=C_{\max}$. However, in the stochastic case, the solution is nontrivial. For now, we assume no inserted idle time, and our objective is to minimize $\mathrm{E}[f(S)]$. Then the job sequence is not at issue, and we can determine the optimal due date.

**Theorem 7.** *Assume that all jobs are processed with no inserted idle time and that the objective is to minimize* $\alpha\mathrm{E}[\max\{0,d-C_{\max}\}]+\beta\mathrm{E}[\max\{0,C_{\max}-d\}]$. *Then the optimal due date corresponds to a service level of* $\beta/(\alpha+\beta)$. *That is, it is optimal to set* $d$ *to the smallest value that satisfies the condition*

$$\Pr\{C_{\max}\le d\}\ge\beta/(\alpha+\beta).$$

This theorem, like the previous one, can be proved by applying critical fractile reasoning. In this case, $\alpha$ represents the unit cost of overestimating the length of the makespan and $\beta$ represents the unit cost of underestimating it. The resulting critical fractile, which appears in the theorem, is $\beta/(\alpha+\beta)$. A numerical illustration follows.

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

Consider a problem containing $n = 5$ jobs with independent processing times. The processing time of each job follows a normal distribution, with mean and variance shown in the table.

| Job $j$ | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| $\mu_j$ | 5.0 | 7.0 | 6.0 | 4.0 | 8.0 |
| $\sigma_j$ | 1.0 | 1.5 | 1.2 | 0.8 | 2.0 |

The unit penalties for earliness and tardiness are $\alpha = 1$ and $\beta = 4$, respectively. The objective is to schedule the jobs so that the stochastic makespan is optimized.

In this example, we can model the properties of the makespan by drawing on the fact that the sum of five independent normal distributions follows a normal distribution, with mean and standard deviation denoted $m$ and $s$, respectively

$$m = \mu_1 + \mu_2 + \cdots + \mu_5 = 30$$

and

$$s = (\sigma_1^2 + \sigma_2^2 + \cdots + \sigma_5^2)^{1/2} = 3.054.$$

The critical fractile is $\beta/(\alpha+\beta) = 0.8$, corresponding to a standard normal $z$-value of 0.8416, for which the optimal value of $d = m + sz = 32.57$. Thus, the asymmetry in $\alpha$ and $\beta$ leads to an optimal due date that is larger than the expected makespan. The difference is safety time.

The stochastic makespan model is highly specialized, however, and it is relatively simple because the job sequence is not at issue: Any job sequence generates the same makespan. Next, we address problems in which each job contributes separately to the E/T objective function and the job sequence matters.

### 3.5. Due Dates as Decisions in the Stochastic E/T-Problem

We now allow jobs to have different due dates as well as different unit costs. The objective function in the stochastic E/T-problem can be expressed as:

$$\mathrm{E}[f(S)] = \sum_{j=1}^{n} [\alpha_j \mathrm{E}(E_j) + \beta_j \mathrm{E}(T_j)]$$

with due dates treated as decisions. The optimal choice of due dates is again determined by the critical fractile rule, as stated in the following result.

**Theorem 8.** *Assume all jobs are processed with no inserted idle time and the objective is to minimize the total expected E/T penalty. Then, for any given sequence, the optimal due date of job $j$ corresponds to a service level of $\beta_j/(\alpha_j + \beta_j)$. That is, it is optimal to set $d_j$ to the smallest value that satisfies the condition*

$$\Pr\{C_j \le d_j\} \ge \beta_j/(\alpha_j + \beta_j).$$

For a given job sequence, we can apply Theorem 8 separately to all jobs and thereby minimize the total expected E/T penalty. However, it is algebraically challenging to apply the theorem unless we assume that the processing times follow a tractable probability distribution, such as the normal. Furthermore, it is important to remember that the theorem does not completely solve our problem: Although we can set the due dates optimally, there is no efficient procedure yet known for determining the optimal job sequence. Thus, the stochastic E/T-problem remains largely unsolved.

For some insight into a heuristic approach to solving this problem, assume that the processing times follow independent normal distributions. Let the processing time of the $j$th job

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

in sequence have mean $\mu_j$, and variance $\sigma_j^2$, and let the variance of the completion time of job $j$ be $s_j^2$, where $s_j^2$ depends on the sequence. Then, because the processing times are independent, we can calculate the mean ($m_j$) and variance ($s_j^2$) of the completion time for job $j$ as follows:

$$m_j = \mu_1 + \mu_2 + \cdots + \mu_j,$$
$$s_j^2 = \sigma_1^2 + \sigma_2^2 + \cdots + \sigma_j^2.$$

We define the standard normal variate $z_j = (d_j - m_j)/s_j$, and we use an asterisk to denote optimal values. Thus, for example, $z_j^* = (d_j^* - m_j)/s_j$ is the $z_j$-value that satisfies Theorem 8. In symbols,

$$z_j^* = \Phi^{-1}[\beta_j/(\alpha_j + \beta_j)],$$

where $\Phi$ denotes the cdf of the standard normal distribution. That is, a safety time of $(z_j^* s_j)$ must yield a service level of $\beta_j/(\alpha_j + \beta_j)$, but note that $m_j$ and $s_j$ are not required to calculate $z_j^*$. Instead, we can calculate it directly from the normal cdf, setting the cumulative probability equal to the critical fractile. We then obtain:

$$d_j^* = \mathrm{E}(C_j) + z_j^* s_j = m_j + z_j^* s_j.$$

Using the algebra of normal distributions, we can then write,

$$\mathrm{E}[f(S)] = \sum_{j=1}^{n} \{(\alpha_j + \beta_j) s_j \varphi(z_j^*)\},$$

where we use $\varphi$ to denote the standard normal density function. In the spirit of the deterministic counterpart, however, suppose that we also consider setting due dates equal to expected completion times:

$$d_j = \mathrm{E}(C_j) = m_j.$$

(These due dates are equivalent to using safety times of zero.) Drawing on the algebra of normal distributions once more, we can show that the expected E/T cost for job $j$ is $(\alpha_j + \beta_j) s_j \varphi(0)$. Therefore, the expected-value approach reduces to finding a sequence that minimizes

$$\mathrm{E}[f(S)] = \sum_{j=1}^{n} (\alpha_j + \beta_j) s_j \varphi(0).$$

In the optimal schedule, the expected E/T cost associated with job $j$, $(\alpha_j + \beta_j) s_j \varphi(z_j^*)$, cannot exceed $(\alpha_j + \beta_j) s_j \varphi(0)$ because $\varphi(z)$ is maximized at $z = 0$. The two ways of setting due dates may lead to different optimal sequences, and unless $\alpha_j = \beta_j$ for all $j$, they lead to different objective function values. This analysis can become complicated, so we illustrate the concepts with a two-job example.

Consider a problem containing $n = 2$ jobs, in which the processing times of the jobs follow independent normal distributions, and the objective is to minimize total expected E/T penalty.

| Job $j$ | 1 | 2 |
|---|---|---|
| $\mu_j$ | 10 | 10 |
| $\sigma_j$ | 3 | 4 |
| $\alpha_j$ | 3 | 9 |
| $\beta_j$ | 27 | 36 |

Here, $\alpha_1 + \beta_1 = 30$ and $\alpha_2 + \beta_2 = 45$. The optimal service levels are:

For job 1, $\beta_1/(\alpha_1 + \beta_1) = 27/30 = 0.9$ (corresponding to $z_1 = 1.282$),

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

For job 2, $\beta_2/(\alpha_2+\beta_2) = 36/45 = 0.8$ (corresponding to $z_2 = 0.842$).

We solve for the best sequence by complete enumeration. For the sequence 1–2 we have $s_1 = 3$ and $s_2 = (3^2+4^2)^{1/2} = 5$. Thus,

$$f(S) = \sum_{j=1}^{n}(\alpha_j+\beta_j)s_j\varphi(z_j) = 30(3)\varphi(1.282) + 45(5)\varphi(0.842)$$
$$= 30(3)(0.1755) + 45(5)(0.2800) = 78.79.$$

For the sequence 2–1, we have $s_2 = 4$ and $s_1 = (3^2+4^2)^{1/2} = 5$. Thus,

$$f(S) = \sum_{j=1}^{n}(\alpha_j+\beta_j)s_j\varphi(z_j) = 45(4)\varphi(0.842) + 30(5)\varphi(1.282)$$
$$= 45(4)(0.2800) + 30(5)(0.1755) = 76.72.$$

Thus, the sequence 2–1 is optimal, and the corresponding due dates are as follows:

$$d_2 = 10 + 4(0.842) = 13.368,$$
$$d_1 = 20 + 5(1.282) = 26.410.$$

Job 2 should complete around time 10 and job 1 around time 20. But if we follow the deterministic counterpart and set the due dates equal to the expected completion times of 10 and 20 for this sequence, the expected penalty would be 131.65. However, without safety times, the opposite sequence (1–2) would be better, with an expected penalty of 125.67, still about 70% above optimal. The implication is that the expected-value approach may not only lead to excessive penalty but also mislead the search for an optimal sequence.

Although there is no known optimizing algorithm short of complete enumeration, a good heuristic comes from a sorting rule. In particular, we can sort the jobs by nondecreasing ratio of $\sigma_j^2/(\alpha_j+\beta_j)\varphi(z_j^*)$, with ties broken in favor of the smallest $\sigma_j$. This heuristic is often optimal for small $n$ (Soroush [12]).

For large $n$, this heuristic is asymptotically optimal. A heuristic is *asymptotically optimal* if, as $n$ grows large, the relative difference between the heuristic solution and the optimum becomes negligible. More formally, let $f(S^*)$ denote the objective function value based on the optimal sequence, $S^*$, and let $f(S^H)$ be the value associated with a heuristic. We say that the heuristic is asymptotically optimal if, in the limit as $n \to \infty$, we have

$$[f(S^*) - f(S^H)]/f(S^*) \to 0.$$

Furthermore, for independent processing times, the ratio heuristic is asymptotically optimal even if the jobs are not distributed normally. This result follows from the fact that for large $n$ the central limit theorem holds, and nearly all completion times are normally distributed. Among all possible sorting rules, only those consistent with this heuristic rule are asymptotically optimal. The tie breaker is not necessary for asymptotic optimality but it is useful because for the normal distribution, if the heuristic rule is consistent with nondecreasing $\sigma_j$, then it is known to be optimal (Portougal and Trietsch [9]).

Because asymptotic optimality does not require normal processing times, the heuristic sorting rule is effective for any processing-time distribution. Furthermore, we can use the sorting heuristic to find an initial seed, and then perform neighborhood searches on the first few jobs (say 5 to 10) to see if an even better sequence can be found. For subsequent jobs, we can rely on the asymptotic optimality of the heuristic. We can use this sorting rule as a crude heuristic even when jobs are not statistically independent. To estimate job parameters for that purpose, we use their marginal distributions. After the sequence is determined, we can set the due dates using the critical fractile rule, which does not require statistical independence or normality. Although the use of marginal distributions is not theoretically correct, this method can at least generate a reasonable seed for a neighborhood search.

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

## 4. The Benefit of Inserted Idle Time

Consider the stochastic E/T-problem with a common due date, as it is a special case that can teach us something about the more general case with distinct due dates. The deterministic counterpart of this problem exhibits three general properties (Baker and Scudder [2]). These are: (1) inserted idle time provides no benefit, (2) a V-shaped schedule (processing times are first nonincreasing, then nondecreasing) is optimal, and (3) one job completes at the due date. In the stochastic case, of course, we would not expect the last condition to hold, but we might hope the other properties apply. However, as we might guess from examples with the normal distribution, V-shaped schedules may not be optimal because they do not account for variance. The remaining question involves inserted idle time. Unfortunately, we may encounter limited success relying on this feature of the deterministic counterpart to solve the stochastic problem. To illustrate, we consider an example with a common due date and symmetric earliness and tardiness costs that are the same for all jobs.

| Job $j$ | 1 | 2 | 3 |
|---|---|---|---|
| $\mathrm{E}(p_j)$ | 3.4 | 1 | 1 |
| $d_j$ | 10 | 10 | 10 |
| $\alpha_j$ | 1 | 1 | 1 |
| $\beta_j$ | 1 | 1 | 1 |

The processing times depend on which of two states of nature occur, as described in the following table.

| State | Job $j$ | 1 | 2 | 3 | Probability |
|---|---|---|---|---|---|
| $S_1$ | $p_j$ | 1 | 1 | 1 | 0.2 |
| $S_2$ | $p_j$ | 4 | 1 | 1 | 0.8 |

Note that $p_1$ is a random variable, but the other two jobs have deterministic processing times. In the deterministic counterpart, job 1 comes first, and the other jobs follow in either order. The sequence begins at time 5.6, so that the second job completes at time 10, and the total E/T penalty is 2.

Now suppose we implement the sequence 1–2–3 in the stochastic case. If we start job 1 at time 5.6, the expected total penalty is 3.52. If we explore other starting times, we find that starting job 1 at time 5.0 leads to an expected total penalty of 3.4, which is the best objective for this sequence. If we schedule job 1 last, the best we can do is to start the schedule at time 8, leading to an expected total penalty of 4.4.

Next, we explore the possibility of inserting idle time in the sequence 1–2–3. Suppose we start the schedule at time 5 but prevent the second job from starting earlier than time 9. In other words, when job 1 completes at time 6 (which occurs with probability 0.2), we allow idleness until time 9, when job 2 starts. This schedule achieves an expected total penalty of 2.6, which is better than we could achieve with no inserted idle time.

This example reveals a complicating factor in stochastic problems with E/T criteria: It may be helpful to allow inserted idle time between jobs, even though (for the case of a common due date) such idle time would not be beneficial in the deterministic counterpart. Thus, we must pay attention to the general case in which inserted idle time is permitted.

### 4.1. Setting Release Dates with Known Due Dates

As the previous example shows, inserted idle time can be beneficial in the stochastic case. The constraint on the start time for the second job is essentially a *release date*, $r_j$, for job $j$. If the machine is available before $r_j$, the machine must wait to start job $j$; but if the machine becomes free after $r_j$, the job can start immediately.

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

It is not always necessary to assign an explicit release date to each job. We may start a search for optimal release dates under the assumption that each job has its own release date, but it is ultimately sufficient to describe a schedule by specifying only release dates that have a positive probability of actually delaying a job. We refer to such release dates as *active*. A release date that is not active is redundant because it never causes a machine to wait.

Release dates define blocks. A *block* is a sequence of jobs processed without delay. If no release date is specified for a job, it is in the same block as the preceding job. (The only exception would be at the start: If no release date is specified for the first job, then processing starts at time 0.) In the stochastic case, adjacent blocks may be processed with or without a gap between them, but the expected size of the gap is positive. In the optimal schedule for the three-job example above, we place job 1 first in sequence and take $r_2 = 9$. Job 1 thus belongs to block 1, whereas the other two jobs make up block 2 and the expected gap is $0.2 \times 3 = 0.6$.

Suppose we are given a set of jobs with distinct due dates and E/T costs, and suppose further that the job sequence is given. The task then is to set release dates that minimize the total expected E/T penalty. It is possible to show that the total expected E/T penalty is a convex function of the release dates. Essentially, we need to search for the best combination of release dates to minimize this total expected penalty.

When we use a sample-based approach, we can find the best combination of release dates by a numerical search, because the problem is convex and thus not difficult in practice. Our model determines optimal release dates for a given sequence, but we still do not have an efficient algorithm for finding the best sequence. For the time being, a heuristic procedure, such as a neighborhood search algorithm, represents the state of the art for finding the optimal sequence.

In this section, we have examined the assumption of no inserted idle time and discovered that it may be restrictive. Optimal solutions to the stochastic E/T-problem may require that idle time be inserted in the schedule, or equivalently, that explicit release dates be set for some of the jobs. At this stage, we have not pursued the full implications of this observation, leaving that task to future research.

## 5. Summary and Conclusions

In this tutorial, we presented the basic single-machine model, which is central to most of scheduling theory, and we gave an overview of traditional extensions to stochastic versions of the basic model. We discussed in detail the shortcomings of using the deterministic counterpart as a device for attacking stochastic problems. First, it ignores the Jensen gap, which is the source of bias in predicted performance. Second, it ignores the practical value of safety time. Although deterministic models may be very useful as sequencing heuristics, they tend to be inadequate unless safety time is addressed correctly.

In practical applications, the bias due to the Jensen gap often has serious consequences. Because of the optimistic bias in deterministic counterparts, the natural way to use a deterministic model for a stochastic situation is to build hidden buffers into each job's estimated processing time. Such hidden safety time is ultimately wasteful. For this reason, traditional stochastic-scheduling models have not been suited to practical applications, even though real problems are often stochastic.

Safe scheduling calls for recognition of safety time, often expressed in service levels or critical-fractile results. We introduced safe scheduling and discussed four general types of models that arise in safe scheduling. Using an analogy to stochastic inventory theory, we identified two approaches to determining safety time: meeting service-level constraints or minimizing the expected value of total economic cost. We considered each of these approaches with two options for achieving safety: (1) Rejecting jobs whose inclusion could undermine the safety of other jobs or (2) accepting all jobs but assigning due dates in an optimal fashion. The combination of the economic approach and due-date setting can be viewed as the

Downloaded from informs.org by [216.73.217.52] on 26 November 2026, at 21:43 . For personal use only, all rights reserved.

stochastic counterpart of deterministic models with E/T penalties. In our coverage of this combination, we assumed no inserted idle time, but we discussed the fact that this assumption may be restrictive. An open area for research appears to be the analysis of release dates to control inserted idle time.

We foresee the themes illustrated in these types of models as giving rise to new opportunities for research and application in stochastic scheduling. Our main conclusion is this: If we wish to fully utilize our historic investment in scheduling theory, we must pay more attention to safe-scheduling formulations.

## References

[1] K. R. Baker and J. W. M. Bertrand. A comparison of due-date selection rules. *AIIE Transactions* 13:123–131, 1981.

[2] K. R. Baker and G. D. Scudder. Sequencing with earliness and tardiness penalties: A survey. *Operations Research* 38:22–36, 1990.

[3] B. P. Banerjee. Single facility sequencing with random execution times. *Operations Research* 13:358–364, 1965.

[4] T. B. Crabill and W. L. Maxwell. Single-machine sequencing with random processing times and random due dates. *Naval Research Logistics Quarterly* 16:549–555, 1969.

[5] J. R. Jackson. Scheduling a production line to minimize maximum tardiness. Research Report 43, Management Sciences Research Project, University of California at Los Angeles, Los Angeles, CA, 1955.

[6] S. M. Johnson. Optimal two- and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly* 1:61–68, 1954.

[7] H. Kise and T. Ibaraki. On Balut's algorithm and NP-completeness for a chance constrained scheduling problem. *Management Science* 29:384–388, 1983.

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

[9] V. Portougal and D. Trietsch. Setting due dates in a stochastic single-machine environment. *Computers & Operations Research* 33:1681–1694, 2006.

[10] M. H. Rothkopf. Scheduling with random service times. *Management Science* 12:707–713, 1966.

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

[12] H. M. Soroush. Sequencing and due-date determination in the stochastic single-machine problem with earliness and tardiness costs. *European Journal of Operations Research* 113:450–468, 1999.

[13] W. Szwarc, A. Grosso, and F. Della Croce. Algorithmic paradoxes of the single-machine total tardiness problem. *Journal of Scheduling* 4:93–104, 2001.

[14] J. M. van den Akker and J. A. Hoogeveen. Minimizing the number of late jobs in case of stochastic processing times with minimum success probabilities. *Journal of Scheduling*. Forthcoming.