The impact of design uncertainty in engineer-to-order project planning

Hajnalka Vaagen and Michal Kaut and Stein W. Wallace

in: European Journal of Operation Research. See also BibTeX entry below.

BibTeX:

```
@article{VaagenEA2017,
  author  = {Hajnalka Vaagen and Michal Kaut and Stein W. Wallace},
  title   = {The impact of design uncertainty in engineer-to-order project planning},
  journal = {European Journal of Operation Research},
  year    = {2017},
  volume  = {261},
  number  = {3},
  pages   = {1098--1109},
  doi     = {10.1016/j.ejor.2017.03.005}
}
```

This is the authors' version of a paper published in European Journal of Operational Research. The published version can be access via DOI 10.1016/j.ejor.2017.03.005.

The impact of design uncertainty in engineer-to-order project planning

Hajnalka Vaagen[1,2], Michal Kaut[2], and Stein W. Wallace[3]

[1]*Norwegian University of Science and Technology (NTNU), Ålesund, Norway*
[2]*SINTEF Technology and Society, Trondheim, Norway*
[3]*Norwegian School of Economics (NHH), Bergen, Norway*

February 1, 2018

A major driver of planning complexity in engineer-to-order (ETO) projects is design uncertainty far into the engineering and production processes. This leads to uncertainty in technical information and will typically lead to a revision of parts of the project network itself. Hence, this uncertainty is different from standard task completion uncertainty. We build a stochastic program to draw attention to, and analyse, the engineering-design planning problem, and in particular, to understand what role design flexibility plays in hedging against such uncertainty. The purpose is not to devise a general stochastic dynamic model to be used in practice, but to demonstrate by the use of small model instances how design flexibility actually adds value to a project and what, exactly, it is that produces this value. This will help us understand better where and when to develop flexibility and buffers, even when not actually solving stochastic models.

Keywords: project scheduling, engineer-to-order, design uncertainty, design flexibility

1. Problem description

We consider a project production system following the engineer-to-order (ETO) approach where design, engineering and production do not commence until after a customer order is confirmed (Rudberg and Wikner, 2004). This approach is used to create products that are tailored for each customer and is used in, for example, shipbuilding and off-shore oil and gas installations. A typical feature of ETO projects, especially in the case of

complex orders such as offshore ships, is a continuous dialogue with the customer after the order has been received. This often leads to specification changes after the design phase of the project has started, sometimes even far into the engineering and production phases.

While the flexibility is good for the customer, for the producer it represents a source of uncertainty in the technical information, which often leads to uncertainty in the project network itself. This leads to continuous adjustments in procurement, engineering and execution (Emblemsvåg, 2014), and suggests that we are dealing with a stochastic dynamic planning problem with uncertainty at two levels. Firstly, we have that task completion times are uncertain and usually correlated for any fixed design. Secondly, design uncertainty is added to this. And this latter layer is not merely a scaling of the first, but can change its structure substantially. Consequently, the resulting dependencies become very complicated.

Design uncertainty is, therefore, a major driver of planning complexity in ETO projects where advanced design and engineering is taking place concurrently with production. Obviously, concurrency is challenging only when design is uncertain. Despite this, design is most commonly separated from project scheduling (Eckert and Clarkson, 2003; Emblemsvåg, 2014), leading to plans that lack the flexibility necessary to handle the true uncertainty.

In general, what is lacking in the classical project scheduling *models* is the possibility to have decisions that are conditioned on arriving information (in our case the progress of tasks and changes in design) as well as future decisions. The major difficulty is that there *is* no arrival of information in these models. However, even though the models do not consider re-planning, this is of course performed in reality, normally by rerunning the existing models based on all new information, that is, *reactive planning*.

It is, however, well established, see for example King and Wallace (2012), that such a sequence of decisions (plans) from static models, often referred to as rolling horizon modelling, can lead to arbitrarily poor decisions. The reason is that each individual plan is inflexible; it assumes the future (in terms of decisions) cannot be changed. And a series of inflexible decisions remains inflexible.

Contrary to reactive scheduling, *proactive scheduling* would imply a scheduling that takes into account both arrival of information and future decisions that might unfold. This is discussed in Jørgensen and Wallace (2000), and might lead to plans (and decisions) that are very different from those stemming from static (non-dynamic) models. The main reason is that dynamic models will suggest decisions that are much more flexible, and that lead to future situations that are much easier to handle when something goes wrong; they include options, see Wallace (2010).

There are very few proactive approaches that discuss the two main issues, arrival of information and future decisions (see details in Section 2), and in any case, these are not very practical. As practitioners increasingly recognize the shortcomings of classical project scheduling models, these are often replaced by team-based judgemental decision processes that automatically open up for behavioural challenges (Vaagen and Aas, 2014). Dealing with the described complexity without model-based decision support is obviously not easier when we lack guidelines on where, when and how to develop flexibility and

buffers. Although buffer management is commonly used to hedge against uncertainty (Van de Vonder et al., 2006), the optimal solutions in stochastic dynamic environments are not 'the static solution plus something'. Rather, the two solutions are normally structurally different; see Wallace (2010) for a detailed discussion.

This is intuitive if we consider, for example, changing an off-shore shipbuilding process from an originally planned cable-layer to a fire-fighter, far into the production process. A fire-fighter requires specialised solutions throughout the bow of the vessel, which implies activity sequencing substantially different from other design solutions. Late adaptation to such complex outfitting designs cannot be handled by fixed schedules for other design solutions, plus some slack, as they require extensive rework.

A second, and probably less intuitive example, is uncertainty in the design of strategic components, e.g., size, technical specifications or supplier of the engine for sea exploration operations. The alternative designs often require very different piping and electro solutions, with different tasks and sequencing. Handling this type of change by adding time buffers to the design-dependent tasks is certainly possible, given that the uncertainty is identified and the buffers are sufficiently large. But with many low-probability/high-impact changes throughout the project life-cycle, in an environment where short delivery time is critical for competitiveness, adding time buffers is obviously a sub-optimal countermeasure.

This leads to our main concern: handling the uncertainty and dynamics generated by frequent design changes, where a particular technical solution may be selected/de-selected by the customer, even far into the production. For a given design, task durations are inherently uncertain and often assumed to follow known and rather simple distributions, e.g., uniform or exponential (Lambrechts et al., 2010), and are frequently handled by time buffers in proactive-reactive approaches (Van de Vonder et al., 2006). But design uncertainty differs from uncertainty in task durations, as it affects the choice (and technical sequencing) of the tasks to be executed. This suggests that the most critical variation is actually caused by design uncertainty. Obviously, this poses challenges when the managerial objective of reducing delivery times triggers concurrency in engineering and execution activities, in projects with frequent low probability/ high impact design changes. This is the case we study.

The purpose of this paper is, therefore, to understand better how engineering design uncertainty affects project planning complexity and how this uncertainty leads to different plans and decisions when taken explicitly into account in the decision models. This will help us understand better where and when to develop flexibility and buffers, even when not actually solving stochastic models. In real projects these models will be far too heavy. We do this by solving small model instances, showing how design flexibility adds value to a project and what this flexibility actually comes from. Within this framework, we seek a methodology that captures the value of future choices on design alternatives. Stochastic programming is, in our view, a good approach for this task, despite its complexities. As we deal with a stochastic dynamic problem not yet solved in the literature, the challenges in formulating and solving the problem necessarily have to be discussed, but that is not the message of this paper. Our main concern is what we can learn about the impact of design uncertainty on planning, by analysing small model instances.

The paper is organized as follows. Section 2 discusses relevant attempts to handle uncertainty in the engineering design process, with particular focus on project management and scheduling. The stochastic modelling approach is described in Section 3. The following two sections present and analyze two test cases. The first is a design-engineering planning case that focuses on the value of flexible (two-step) design strategies. The second test case is set up to help us characterize good plans when these flexible design strategies are not available. Insights from analyzing the test cases come in Section 6, before we conclude the paper in Section 7. The detailed mathematical formulation of the stochastic dynamic project scheduling model, needed for our analysis, is given in the appendix.

2. Existing literature

The engineering design process connects the phases of basic (preliminary) design with detailed design and project planning and scheduling, where one design alternative normally excludes other alternatives. Most commonly, design planning and project scheduling are treated as separated stages. This separation is problematic in an uncertain world where speed to market drives competitiveness, and design activities are necessarily performed concurrently with planning and execution (Eckert and Clarkson, 2003; Emblemsvåg, 2014).

Decision-making trends in project management and advances in scheduling techniques are reviewed in Rolstadås et al. (2014), highlighting the need for increased use of analytical approaches to handle project uncertainty. That said, the number of model-based approaches to support project planning is substantial, but with important shortcomings in handling uncertainty, dependencies and dynamics (Herroelen, 2007; Vaagen and Aas, 2014; Van de Vonder et al., 2006). Most importantly, a large share of the research assumes a static and deterministic environment, while real project activities often are subject to substantial uncertainty, leading to schedule disruptions.

But also approaches developed to handle uncertainty fail to properly handle project uncertainty and dynamics, e.g., proactive or reactive scheduling dealing with a sequence of decisions from static models. For important work on generating robust (deterministic) baseline schedules that are sufficiently protected against (anticipated) uncertainty, and reactive policies deployed to adjust the baseline schedules after uncertainty is revealed, see Van de Vonder et al. (2006) and Herroelen (2007). Here, statistical information about possible disruptions is used to create baseline (deterministic) schedules, which are revised/reoptimized when necessary. The underlying idea is to create a 'solution robust' baseline schedule, normally by adding time buffers (Herroelen and Leus, 2005), or by developing multiple baselines before and during the project execution, and responding to anticipated events by switching to the schedule that corresponds to the event that occurred (see, e.g., Artigues et al., 2005). The latter is also called *contingent scheduling*, as it focuses on alternatives. These solutions are, however, not flexible, despite the alternatives provided, as the approach applied consists of a series of inflexible decisions.

A second research path to handle uncertainty is the stochastic resource-constrained

project scheduling problem, with the most common objective to minimize the project completion time (see the classification of Herroelen et al. (2001)), determining at each stage which activities are to be started next. The most known proactive baseline scheduling approach is perhaps the critical path method, CPM (Morris, 1994), balancing time and cost while resource-oriented, and the rather similar PERT, with a focus on stochastic activity durations. The PERT method aims to assist managers by identifying the activities with greatest impact on overall project duration. The potentials and shortcomings of these methods are discussed in, e.g., Herroelen et al. (2002), concluding that the insertion of time buffers on critical chains may generate unnecessarily high project due dates, and may also fail to prevent the propagation of uncertainty throughout the schedule. Related to this, Wynn and Clarkson (2009) show how simulation can be used to align design activities and information flow with project targets (e.g., milestone delivery dates), and how monitoring and re-planning is supported by using non-ideal practitioner metrics, revealed by the study. PERT and other simulation approaches provide a picture of project risk and simulate the effects of options for decisions (before decisions are made). These facilitate better planning, but *still lack decisions*. Wynn and Clarkson (2009) point to the need for future research on how to select the 'best' schedule from a set of alternatives (i.e., contingent scheduling) generated by simulation approaches. Most decision makers still choose the decisions that fit the expected (or sometimes most likely) outcome, and overlook the potentially high costs of adapting to a different scenario.

Attempts to overcome these shortcomings exist by adding decisions within the simulation model: e.g., the decision of increasing resources if we are late relative to the plan (Steinhauer and Soyka, 2012; Steinhauer and Wagner, 2008). Future decisions, e.g., on a future design alternative, are however not explicitly taken into account, as this cannot be done within a simulation model. To do this, stochastic dynamic decision models are needed.

An interesting alternative can be found in Deblaere et al. (2011). They operate with a decision rule, and use simulation to estimate the effects of specific parameter settings. By intelligently updating the parameters, near-optimal settings can be found. Obviously, it is not possible to say exactly how good an optimized decision rule is, one can only compare with others. They show, however, that their approach beats many alternatives. More importantly for us, such an approach handles the two issues we are concerned with: arrival of information and future decisions. This comes at the cost of not knowing how good the decision rule is. Our approach needs optimal decisions, otherwise it is not clear if we discuss the model at hand or the decision rule, but for us it comes at the price of not having a model that is usable in practice. Rather, we end up with a tool for a principal study of the problems at hand.

Finding a way to formulate the general stochastic dynamic scheduling problem is difficult (Jørgensen and Wallace, 2000), mainly because the order of decisions is not fixed but depends on previous decisions and the realization of random variables (Kall and Wallace, 1994). Jørgensen and Wallace (1998) present a stochastic dynamic model with an independent decision maker allocated to each node, with information only about the local (nodal) state of the project; i.e., no information about the actual decision of

the other decision makers. This model is solved for a small example in Jørgensen and Wallace (2000), concluding that little is gained by the proactive schedule compared to the stochastic static model in reactive fashion.

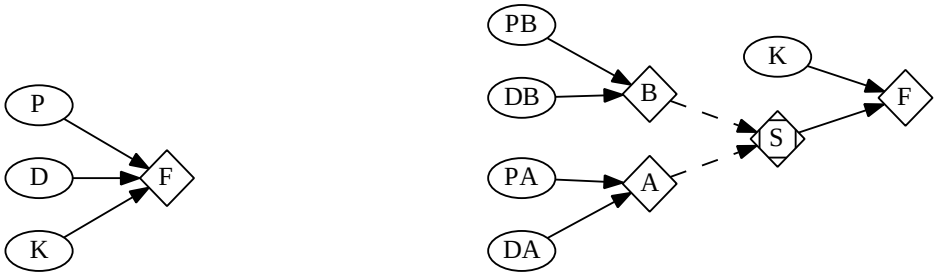
None of the referred papers considers a two-level uncertainty problem identified in this paper, where design uncertainty influences the project network itself. A two-level stochastic problem of conceptual relevance for this paper is treated in Vaagen et al. (2011), showing that hedging in a product portfolio problem with substantial uncertainty is mainly driven by the two possible design states (preferred/not preferred by the market), which is very similar to our choice of design.

The difficulties in modelling and solving large mixed integer stochastic problems also motivated academics to discuss whether uncertainty should be ignored in planning, or, if not, when and how it should be included in the planning process. Discussion on the use of deterministic solutions in stochastic settings are provided in Thapalia et al. (2011, 2012a,b) and Maggioni and Wallace (2012). Within the setting of a stochastic mixed-integer network design model, the authors show that deterministically chosen edges, combined with edge capacities set by a stochastic *linear* model, can lead to very good solutions of the overall stochastic mixed-integer model. The planning problem of this paper can be seen as a related scheduling problem, and it may be tempting to look at similar approaches. We deal, however, with a two-level uncertainty structure, where we anticipate deterministically-chosen sequences not to deliver good solutions for different design alternatives, and our focus is on the design uncertainty, not the task uncertainty for a given design.

3. Stochastic programming formulation

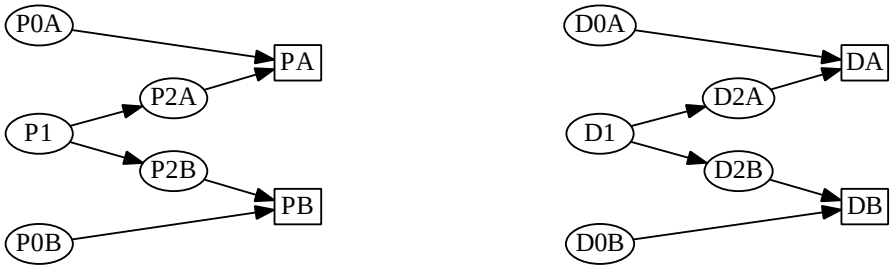
In this section, we build a model for the case of stochastic changes in design specifications, while keeping the task durations deterministic. The main reason for this separation is that our goal is to study the impact of design uncertainty on planning, and adding uncertainty in task durations would just make the results more difficult to interpret. As discussed in Section 1, our problem understanding (supported by contextual exploratory studies) suggests that the most critical variation is caused by design uncertainty, as a particular technical solution may be selected/de-selected by the customer, even far into the production, resulting in a change in the network itself.

The study of how design uncertainty and uncertain task durations *together* affect plans is left for future research. For valuable insights into how task duration uncertainty affects plans for a given design, as well as for more general proactive approaches to handle this type of uncertainty, we refer at this point to the body of literature on proactive-reactive project scheduling discussed in Van de Vonder et al. (2006), Herroelen (2007) and Deblaere et al. (2011).



(a) Initial project.

(b) Adding design choice for activities P and D.



(c) A flexible version of activity P.

(d) A flexible version of activity D.

Figure 1: Step-by-step construction of the motivating example. Real activities are depicted by ellipses, indicator activities with *and*-dependency by diamonds, *or*-dependency by rectangles, and the stochastic dependency by a combination of the two.

3.1. Motivating example

We introduce the most important problem elements and concepts by constructing an example project, which we support by a real-life example from shipbuilding for complex sea-exploration operations. Consider uncertainty in engine design, an example introduced in Section 1. Assume we are in the engineering design phase, a phase which is divided into basic and detail design. The overall hull structure, with strategic scope outfitting (like the engine), is modelled in 3D and split into blocks/units by the basic design team. Units are then complemented with production details (e.g., piping, electro) by the detail design team.

We start with a simple project, installation of the engine, consisting of three activities (tasks), namely piping (P), electro (D) and accommodation (K), depicted in Figure 1a. There, we have introduced an indicator activity F depending on the three tasks, P, D and K. The diamond shape shows that the dependency is of type 'and', i.e., the activity needs all its predecessors to finish in order to start.

Now, let us assume that the activities piping (P) and electro (D) are design-dependent, and the customer can choose between two design alternatives, A and B (referring to different sizes and technical specifications, from distinct suppliers), but the choice is not taken at the time the basic design model is transferred to the detail design team. We are, hence, facing design-dependent uncertainty in tasks P and D.

This gives the network in Figure 1b, where we have introduced three new indicator

activities A, B, and S. The latter is of a special type, since S depends on either A *or* B, dependent on what the customer decides. For example, if the customer chooses design A, this would translate into activity S depending on activity A and hence on activities PA *and* DA.

Finally, we assume that the design-dependent activities P and D can be run as activities specialized for a given design from the start (call it the 'one-step' version), *or* in a modularized fashion with some common parts for the two designs A and B, where we can start with the common part and postpone the specialization[1] (call it the flexible two-step version). For project P, this means replacing nodes PA and PB from Figure 1b by the network of nodes depicted in Figure 1c. There, activities PA and PB become indicator activities with dependencies of type 'or', i.e., they can start when at least one of their predecessors has finished. Figure 1d shows the corresponding network for activity D. The complete project graph is shown in Figure 2 at the start of Section 4, where we use this project for our numerical tests.

3.2. Modelling requirements

Based on the example presented above, we can now create a list of features that our model has to have in order to be able to model the described project:

- Indicator (dummy) activities, i.e., activities that take zero time and consume no resources.
- Activity dependencies of type *'and'* (wait for all the specified activities) and *'or'* (wait for at least one activity).
- Design dependencies, controlled by the customer. We will come back to these later, when we discuss uncertainty.
- The possibility to stop an ongoing activity, so we can react to design changes.
- The ability to require that we undo one activity before we can start another one. This is needed for activities representing different designs/solutions of the same element. For example, in the network from Figure 1c, we might require that to start P0B, one has to undo P0A and P2A, if any of them has been started. Moreover, since it is possible to stop an activity before it finishes, the duration of an undo activity should depend on how far we have come with the activity being undone.

In addition, we need a convex resource-cost function to model the possibility of hiring extra resources at extra costs. However, unlike Jørgensen and Wallace (2000), we let the resource-usage per activity be fixed. If we want to have the option to speed up an activity using extra resources, we have to create a copy of the activity with higher resource usage and shorter duration, and connect it to the original version with an 'or'

[1] In most cases, this extra flexibility will come at a cost

dependency. In other words, the extra resources that we model, can be used to run more activities, or more resource-hungry activities, but they do not change the duration of the activities.

Therefore, we add the following requirements, not directly deducible from the example:

- Each activity has a specified resource usage per period, to model access to labour, space, and equipment.
- The possibility to use a convex resource-cost function, to model the ability to hire extra resources at extra costs.

These requirements result in a rather complex stochastic mixed-integer optimization model. Since its formulation is not necessary for understanding the results, we present it in Appendix A. The model has been implemented in the GNU MathProg modelling language and the test instances were solved using Fico™ Xpress Optimizer.

4. Test case 1 – The value of flexible (two-step) designs

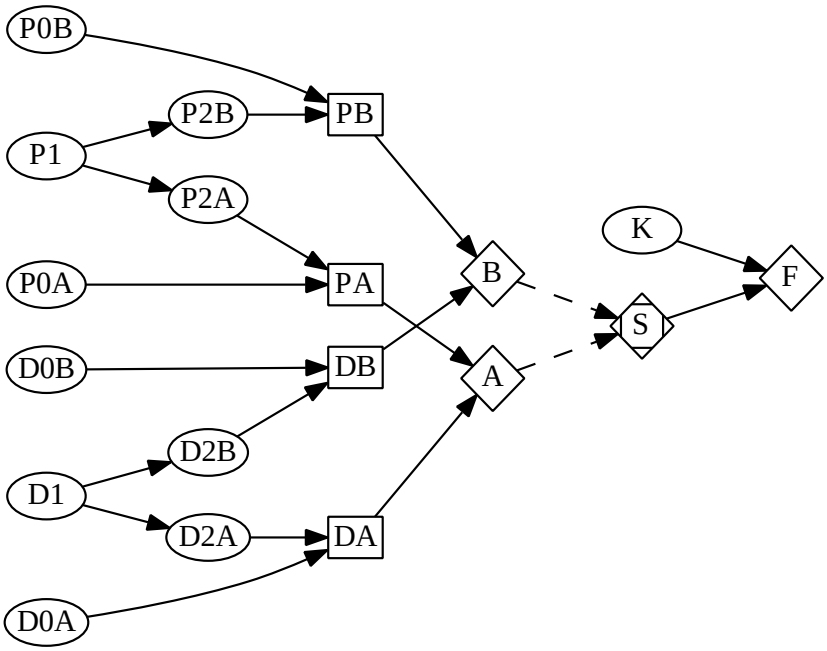


Figure 2: Example engineering-design planning problem. Real tasks are depicted by ellipses, indicator tasks with *and*-dependency by diamonds, *or*-dependency by rectangles, and the stochastic dependency by a combination of the two.

For the first test case, we use the project presented in Section 3.1. The dependency graph is presented in Figure 2, where we omit all the undo activities for the sake of

Table 1: Activity durations of the base case.

activity	P0A	P0B	P1	P2A	P2B	D0A	D0B	D1	D2A	D2B	K
duration	4	3	2	3	2	3	4	2	2	3	2

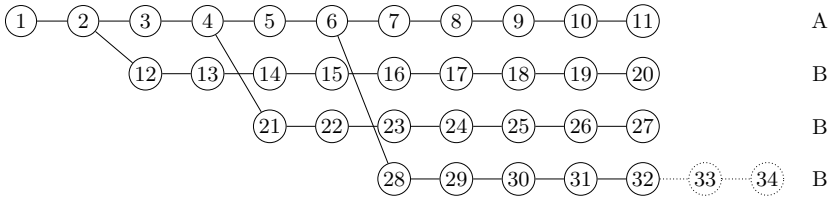


Figure 3: Scenario tree for text case 1, with specified design variant next to each scenario. The dotted part represents two periods added to the last scenario in some of the tests.

readability. The activities' durations are presented in Table 1. Note that the flexible two-step paths to PA, PB, DA, and DB take one period longer than the non-flexible direct activities.

The planning horizon consists of 11 half-week periods, so the maximal duration is 5.5 weeks. We assume that the customer has asked for design variant A, but there is a chance that the specification can be changed to B during the duration of the project. We allow the change to happen after one, two, and three weeks, i.e., after periods 2, 4, and 6. The resulting scenario tree is presented in Figure 3.

We have only one resource r and each real activity uses one unit of the resource per period. We can use up to four units of the resource in each period, where the first two units cost 1.0, the third unit 1.5, and the fourth 2.0.[2]

Since we want the project to finish as soon as possible, we use an increasing penalty for the overall finish time: the penalty is zero for the first five periods, then increases by 0.5 for the next two periods, and after that continues to increase with a gradient that doubles every two periods. See Figure 4 for the resulting penalty function.

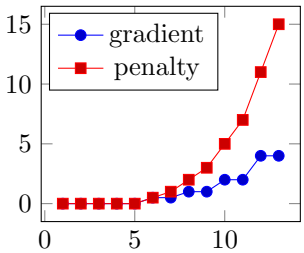


Figure 4: Penalty for finishing the project in a given period.

We have run the test with an increasing probability of a switch to B: 1%, 5%, 10% and 20% at each branching. This means that the probability of no change decreases

[2]Using the model notation from Appendix A, this means $\bar{L}_{r,1} = 2$, $C^R_{r,1} = 1.0$, $\bar{L}_{r,2} = 1$, $C^R_{r,2} = 1.5$, $\bar{L}_{r,3} = 1$, and $C^R_{r,3} = 2.0$.

from 97% to 85.7%, 72.9%, and finally 51.2% in the last case. We have also solved the case without uncertainty, i.e., with only the first scenario. In this case, the total project cost is 9. This is a *lower bound* on the expected projects cost in the stochastic case, as any late design change can only increase costs, given the data and structures we have presented.

4.1. Comparing the proactive and reactive strategy

The *reactive strategy* amounts to starting with the plan corresponding to design A and updating it whenever new information arrives, i.e., in scenario-tree nodes 12, 21, and 28 (see Figure 3). The *proactive strategy* corresponds to solving the complete stochastic model. This strategy not only adapts to changes as they arrive, but anticipates them and is prepared to react to them. In other words, this strategy takes the future uncertainty into account from the very beginning, while the reactive approach only reacts to it when something happens, without any preparations.

It turns out that the reactive strategy is infeasible in the last scenarios, since five periods is not enough time to both unistall A and install B. We therefore add two periods (one week) to the last scenario, as depicted in Figure 3.

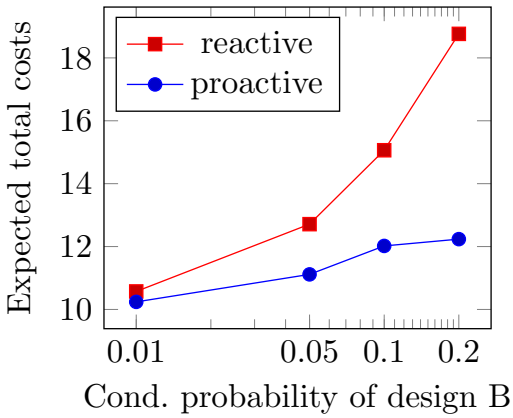


Figure 5: Comparison of the expected total costs of the reactive (deterministic) and proactive (stochastic) strategies, for the case with an extra week in the last scenario.

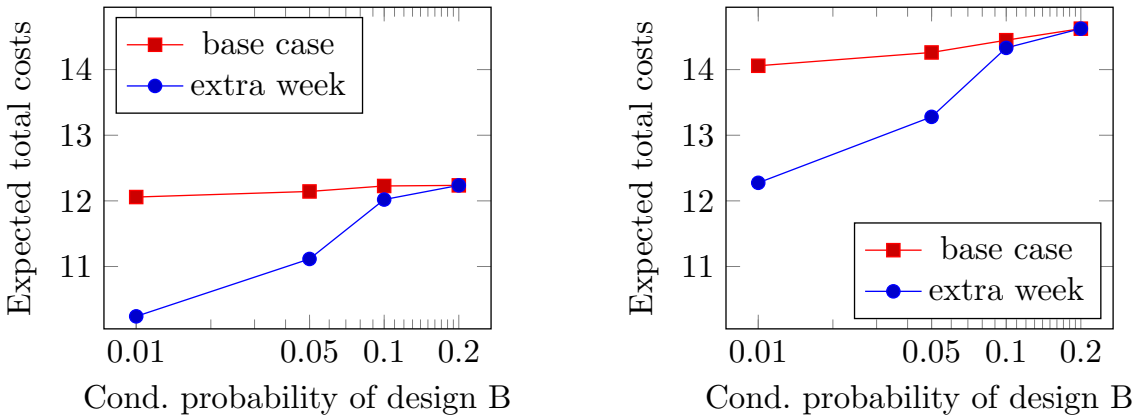
The total expected costs of both strategies are presented in Figure 5. There, we can see the cost of the reactive strategy increases from 9 in the case of no design changes to 10.6 in the case of a 1% probability of a switch to design B, and then increases to 18.8 as the switch probability approaches 20% per branching. That is, we observe more than **100% increase in costs**, a potentially severe situation, *as we move from full certainty (which rarely, if at all, happens) to a significant uncertainty in design, in a reactive fashion.* Recall that in the scenario tree in Figure 4, we allow a switch from A to B after 2, 4 or 6 periods, which means that with 20% chance for B per branching, the probability of no change is only 51.2%, i.e., the designs are almost equally likely.

The *proactive strategy*, on the other hand, handles the uncertainty much better, since the expected costs with 20% design-switch probability are only 12.3, an increase of 36% compared to the case without uncertainty. In total, this means that the proactive strategy results in about **35% lower expected costs** than the reactive strategy, when the probability of a switch to design B is 20% per branching—this is the *value of the proactive strategy*.

Looking at the optimal solutions, the major difference between the reactive and proactive approach is that the former starts with the one-step versions of design A (activities P0A and D0A) and changes to design B when necessary—regardless of the probability of change. The proactive version, on the other hand, reacts to the increasing uncertainty by postponement and by switching to the flexible two-step version P1 and D1.

4.2. The impact of an extended time horizon

To analyse the expected project costs when we have more freedom with respect to completion time (i.e., time is not critical), we compare the costs of the base case with and without the extra week in the last scenario. The results are presented in Figure 6a. Observe that without an extra week, the expected costs are stable around 12, for all the tested probabilities of design B. When we add the extra week (as we did in the previous test), the expected costs fall significantly with a decreasing probability of a switch to design B. This indicates that under relatively low design uncertainty, the potential to reduce costs is high if we allow for a less tight time horizon, but only if we apply the flexible versions of the design dependent activities P and D.



(a) Base case **(b)** Non-parallel P and D

Figure 6: Comparison of the expected total costs of the first data variant of the base case and the case with an extra week in the last scenario, for the base case (6a) and the variant where P and D cannot run in parallel (6b).

4.3. The impact of non-parallel activities

In this test, we added the requirement that no activities related to the uncertain-design specific tasks P and D can run in parallel (at the same time). This is done to avoid collisions between virtual or real objects in space; on the 3D model in the design-engineering phase, or on parts of the vessel in the production. This challenge is imposed by the connection of uncertain design to project scheduling. One example of activities requiring this type of constraint is the design of engine-specific piping and electro solutions in the motor casing (i.e., two systems occupying the same physical space), when the design of the engine is not predefined.

The expected total costs for this case, both with and without the extra week in the last scenario, are presented in Figure 6b. We can see that the advantage of the extra week is similar as in the base case, while the total costs are, naturally, higher than they were in the base case.

A closer look at the solution shows that the extra requirement leads to increased use of the flexible two-step version of the design-dependent activities; even when the probability of B is as low as 5% per branching. The extra constraints make it harder to finish the project within the deadline, and without this flexibility we would face infeasibility within the defined time limit. A second consequence is that the reactive approach becomes infeasible (again, within the defined time limit) in all the stochastic cases, even those with an extra week. This shows that using the flexible approaches of the design-dependent activities is the only way to meet the deadline in this version of the problem.

4.4. Additional tests with differentiated completion times of the two-step design variants

We have repeated all the above test with a variant of the test case where the first step of the flexible design-dependent activities P and D takes one period more, while the second steps are correspondingly shorter:

activity	P0A	P0B	P1	P2A	P2B	D0A	D0B	D1	D2A	D2B	K
base case	4	3	2	3	2	3	4	2	2	3	2
variant			3	2	1			3	1	2	

Our results show that a long completion time of the 1st step (common for both designs A and B), combined with a short completion time of the 2nd step (short reaction time for specialisation) implies more use of flexible approaches. On the other hand, when the time used on the 1st step is low and the completion time of the 2nd step specialisation (reactivity) is high, postponement is preferred with one-step (non-flexible) versions and time buffers.

The model was also tested with a symmetric case with postponed specification. For this case we built a scenario tree with 50% probability of design A and B per branching.

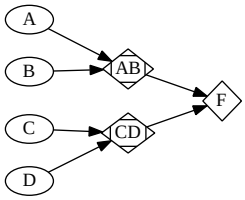


Figure 7: Test case 2 – structure of the project.

We compared the results with those using the Figure 3 scenario tree under 20% probability for B per branching, as this case gives a total probability of 51.2% of no change from design A. The test results did not provide us with additional findings, so we leave it out of the paper.

Summarizing the findings of test case 1, in most situations, the *flexible two-step versions* of the design-dependent activities are preferred to postponements combined with one-step versions; this is true particularly when design uncertainty is high and time is critical. Under low design uncertainty (the probability of a switch to B is below 5% per branching), we observe non-flexible one-step versions that are un-installed when the design is changed from A to B. This can only be observed in the base case with an extra week, as without this time buffer there is not enough time to un-install the 'wrong' activities, and two-step approaches become the only way to achieve solution within the defined time frame. Flexible two-step versions of the design-dependent activities are, therefore, more valued *when time is critical*.

5. Test case 2 – Flexible (two-step) design is not available

Our second test case is motivated by a situation where we are planning outfitting a vessel with four possible equipment designs A, B, C, and D, but we lack the option of a flexible (two-step) design solution. This can be seen as an example of uncertainty where the outfitting equipment differs substantially with respect to the scope of the vessel. This is a situation faced by shipowners when ordering a vessel before the exact nature of the sea operations is fixed. Assume we know that the shipowner will choose one of A and B (modelled as a logical node AB), and one of C and D (modelled as a logical node CD); see Figure 7. The exact choice is not known and will be revealed first some time after the project start.

Unlike the first test case, the activities differ not only in duration, but also in their resource usage. We use a single resource, labour, and the activities resource usage and durations are presented in Table 2. The numbers have been chosen such that the choice between A and B implies uncertainty in duration, with the expected value equal to the duration of C and D. Analogously, the choice between C and D represents uncertainty in labour usage, again with the expected usage equal to the one of A and B. Note that A and C are the simpler options, while B and D are the more complex variants; B taking

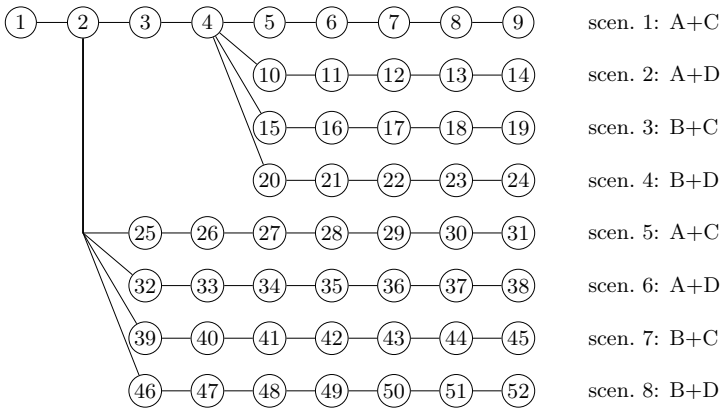


Figure 8: Scenario tree for test case 2, with selected activities in each scenario.

more time than A and D requiring more labour than C. In other words, B and D are *high-impact* variants, while A and C are *low impact*.

Table 2: Activity properties for test case 2.

task	A	B	C	D
duration	2	4	3	3
labour	2	2	1	3

There are four units of labour available in each period, the first two costing 1 per period and the rest respectively 1.5 and 2 per period. Note that this implies that activity D cannot run in parallel with activities A or B. The model has nine periods. To encourage early finish, there is a penalty of 0.5 for finishing in the penultimate period and 1.0 for the last one.

The uncertainty can be revealed either after two or after four periods, with both being equally likely. Furthermore, all the four variants (A+C, A+D, B+C, and B+D) are equally likely. This implies eight scenarios with probability 0.125 each and scenario-tree structure as shown in Figure 8.

Because of the tight schedule, we have to consider a situation where we have started work on an activity, only to find that it is no longer needed. For this, we add extra undo-tasks for each of the main activities A, B, C, and D. These use the same amount of resources as the activities they undo and their duration is equal to the time spent on the activity. In addition, there is a penalty of 10 for not undoing an unwanted activity. The value has been chosen so high that this option is used only as a last resource.

5.1. Results

The optimal expected costs of the *proactive strategy* (the stochastic dynamic case) are presented in Table 3. There, we show costs for the base version described above, as well as versions where we have reduced the time horizon to 8 and 7 periods, by increasing the relevant finish-time penalties to infinity. In addition, we present the optimal costs

of each of the variant solved deterministically. Note that the average of those costs is 12.75, so the expected cost of uncertainty is $16.19 - 12.75 = 3.44$.

Table 3: Optimal expected costs of the proactive strategies with different time horizons, and deterministic solutions with known designs.

variant	base	8 per.	7 per.	A+C	A+D	B+C	B+D
exp. cost	16.19	21.38	24.13	7	14.5	11	18.5

The optimal proactive strategy in the base case starts with postponement, i.e., we do nothing in the first two periods, waiting for the possibility to learn the preferred equipment design after two periods. If this happens, we simply install two selected designs, in an arbitrary order. If we do not learn, i.e., if we move to node 3 of the scenario tree, we start with activity B (one of the high-impact variants) and wait until we learn more in period 5. Then, if the selection happens to be A+C or A+D, we have to undo B in addition to finishing the selected activities.

When we reduce the project time to 8 periods (11% reduction in project completion time), the optimal solution starts with postponement for one period and then initiates activity D (the other high-impact variant). This should be undone and replaced by C in all odd-numbered scenarios, but there is not time for that in scenarios 1 and 3. In those cases, we have to leave D in place and pay the penalty, hence the increase in costs.

With time horizon reduced to 7 periods (22% reduction in project completion time) there is not enough time to postpone, and we start with activity B already in the first period. Another change is that we start A in node 3, so by the end of the fourth period in node 4, we have both A and B in place, even if we know we will need only one of them. The unwanted activity gets undone only in scenario 3 (case B+C), while in the other three it is left in place.

We have also tested the cost of the *reactive strategy*, i.e., implementing a deterministic solution and adjusting it as the information changes. This is easily done by solving the stochastic model with fixed decisions in nodes 1–4 (before revealing the uncertainty). Since we assume that all four variants are equally likely, there is no single 'deterministic solution'. Instead, we test solutions for each of the four cases. Moreover, since the order of the activities does not matter in the deterministic solutions, we test both variants of each case, eight variants in total.

The results are presented in Table 4. There, we see that three of the variants do not have a feasible solution at all and the rest are significantly more expensive than the stochastic solution. This again shows that ignoring the uncertainty and adjusting first when something changes is a costly approach. Note that all the tree infeasible variants start with a low-impact activity. Moreover, the fourth variant starting with a low-impact activity, 'A,D', is feasible only because it initializes a high-impact activity (D) already in the third period.

Finally, we test what happens when we decrease the probability of activity B, i.e., the activity that we otherwise start with, for the case with 9 periods. The results are

Table 4: Expected costs of the reactive strategy of Test case 2

solution	A,C	A,D	B,C	B,D	C,A	C,B	D,A	D,B
exp. cost	inf.	22.75	19.88	19.88	inf.	inf.	24.25	23.75

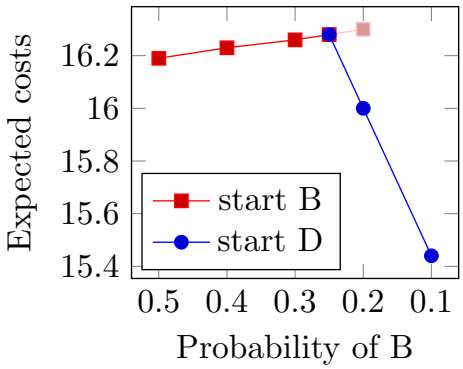


Figure 9: Expected costs of proactive strategies starting with the two high-impact activities B and D, as a function of the probability of activity B (vs. A).

presented in Figure 9. We can see that starting with B (after the initial postponement) stays optimal as long as its probability is high enough, in our case above 25%). Below this threshold, the optimal strategy switches to starting with D, the second high-impact activity. Note that even though decreasing the probability of B means a decrease in the overall uncertainty, the expected costs increase as long as we keep the same solution. This is natural, since the solution is 'betting' on a variant that is less and less likely. The fact that the same solution nevertheless prevails for such a range of probabilities indicates that prioritizing this high-impact activity is indeed advantageous.

Summarizing Test case 2, we conclude, again, that proactive and reactive approaches are different from the very beginning. In proactive scheduling, postponement is preferred whenever possible, combined with prioritization of activities with high impact on the project. The reactive strategy, on the other hand, starts with the assumed design combination and then adapts to the chosen design when necessary. In other words, neither postponement nor impact-based prioritization has value in a deterministic setting.

6. Managerial implications – Guidelines on where and when to develop flexibility and time buffers

The results indicate that the optimal objective function value in a static scheduling model cannot be trusted as a reliable estimate for project costs. An update to new customer requirements is obviously necessary, and using deterministic static models for budgeting and scheduling purposes will lead to potentially high cost overruns (up to 100% in our cases). A proactive strategy that captures the value of future design decisions improves the expected project costs, and we show such cost improvements of up to 50%; i.e., *the*

value of a proactive strategy is high. As 'costs' in our model reflect engineering time used, the implications for human resource planning are important.

High value of a proactive strategy is not unexpected from a stochastic programming point of view. From a project management and practitioner point of view, it motivates further efforts to understand better where to develop flexibility in plans to handle design uncertainty. As the model we developed is far too heavy to implement in real projects, our goal is to understand under which circumstances what kind of flexibility would perform better and by how much, and how design flexibility adds value to the project and what, exactly, it is that produces this value. We discuss this below.

We show that good approaches to situations with uncertain designs are *flexible hedging strategies*, where 'hedging' means developing activities that are common for alternative designs, and 'flexible' refers to the option to postpone design specific decisions to a later point in time. Strategies with options have higher initial costs than those without options (i.e., creating flexibility is usually not for free), but enable adaptation to the different future situations with lower total expected costs (in our case, up to 50% lower, as compared to a deterministic reactive strategy). Note that the trade-off between the fixed costs of creating flexibility (e.g., the costs of developing a modular architecture based design) *and* the benefits of such a strategy is not considered in this paper.

In a deterministic setting, flexibility and hedging designs do not have any value (and meaning), as we know from the start that one of the 'investments' will be discarded. This becomes obvious if we evaluate the strategy *after the fact*, that is, once we know which design was chosen. Then the direct (inflexible) way of implementing that design will be seen as the best thing we could have done. Ahead of planning, though, we don't know which one will be chosen, and flexibility created by a proactive strategy allows for both feasibility and reduced adaptation costs.

In most cases, the *flexible two-step versions* of the design-dependent activities are preferred to postponement combined with one-step versions; this is true particularly when design uncertainty is high and time is critical. Under low design uncertainty, we observe non-flexible one-step versions that are un-installed when the design is changed.

Guidelines derived from our results on what kind of flexibility would perform better in which circumstances are summarized below.

Flexibility enabled by two-step versions of the design-dependent activities – with a first step common for both design alternatives and specialization in a second step; e.g., modular design – has high value under the following conditions:

- When time to market is critical. This confirms research on the value of modularized design (or processes) to enable shorter lead times.
- Under requirements that some uncertain design-specific activities cannot be run in parallel. In such cases, flexible versions of these activities may be the only way to meet the project deadline.
- When two-step versions of the uncertain activities ensure high reactivity, by relatively low completion times of the specialization (step 2) tasks, compared to the

common platform (step 1) tasks.

Postponement of design specific decisions combined with one-step versions of the uncertain tasks (e.g., integral design) is to be preferred under the following conditions:

- When completion time of the specialization (step 2) tasks is high, compared to the common platform (step 1) tasks. In other words, a flexible design approach has low value when the specialization time is high.
- When the uncertainty in design is relatively low.
- When time buffers can be added to the project completion time.

When a modular design-strategy is not available, or not wanted (e.g., because of high development costs), *flexibility through postponement of uncertain decisions combined with prioritization of the activities with highest impact on the project* shows to be valuable. Postponement is preferred to impact-based prioritization whenever possible. For an innovative methodology to develop impact-based prioritization rules to handle uncertainty that is difficult to quantify and therefore difficult to manage (e.g., low probability/high impact design changes discussed in this paper), see Simchi-Levi et al. (2015). Although this method is not developed within a project (but in a global automotive supply chain context), it augments our findings on prioritizing high impact activities when uncertainty is a major element of the planning problem but its probability difficult to estimate.

Finally, although we indicate high potential to reduce the expected project costs by a less tight time horizon, industry trends move in the opposite direction by a continuous urge to reduce lead times. Turning, therefore, our results for the first example from Section 5.2 the other way around, we demonstrate *the impact of design uncertainty on plans when the reduction of project completion time is a managerial objective*: We show that such strategies (i) may be costly (here, about 20% cost increase for about 15% reduction in completion time), and (ii) may require prior investments in flexible versions of the uncertain design specific activities, as there is no way to achieve solutions within the defined time limit, without this type of flexibility. These implications trigger a move from integral design to platform based modular design, and emphasize the necessary extra costs imposed by a shorter building period. In our second example, these extra costs are 32% and 49% for project time decrease of 11% and 22%, respectively.

7. Conclusions

With this paper we extended the scope of research on project scheduling, by connecting design to project planning in a stochastic dynamic model, representing *a proactive strategy*. Our main motivation was to understand the impact of design uncertainty on project planning, as without this knowledge it is difficult to achieve good solutions for concurrency in design, engineering and execution. To deal with the problem, we developed a stochastic programming model. For practical reasons, we focused on small model

instances of the true problem, to demonstrate how design flexibility adds value to the project and what, exactly, it is that produces this value. This has helped us understand better where and when to develop flexibility and buffers, even when not actually solving stochastic models. Citing King and Wallace (2012): *"Understanding why we need stochastic programs, being able to formulate them, and, finally, finding out what it is that makes solutions good can help us find good solutions without actually solving the stochastic programs."*

Since such models are difficult to solve for large projects, the insights presented in this paper are potentially valuable to improve judgemental decision-making when planning dynamic and uncertain projects. In practice, many companies need planning guidelines to understand where to develop flexibility. The insights also provide useful learning to simulation approaches, as we show what solution structures are important to investigate when a full scenario tree evaluation is impossible due to the size of the problem.

The solutions imply high value in an early identification of design alternatives *and* activities that have most impact on project completion time. These often carry the most uncertainty and may require flexibility in task solutions and hedging plans in early phases.

Our goal is to extend the model with uncertainty in completion times (even for a fixed design) and correlations, to understand what type of uncertainty has high impact on plans.

Acknowledgments

The authors thank anonymous reviewers for valuable comments. This paper is part of the competence-building research project NextShip, under Norwegian Research Council grant agreement 216418/O70.

References

C. Artigues, J.-C. Billaut, and C. Esswein. Maximization of solution flexibility for robust shop scheduling. *European Journal of Operational Research*, 165(2):314–328, 2005. doi: 10.1016/j.ejor.2004.04.004.

F. Deblaere, E. Demeulemeester, and W. Herroelen. Proactive policies for the stochastic resource-constrained project scheduling problem. *European Journal of Operational Research*, 214(2):308–316, 2011. doi: 10.1016/j.ejor.2011.04.019.

C. M. Eckert and P. J. Clarkson. The reality of design process planning. In A. Folkeson, K. Gralen, M. Norell, and U. Sellgren, editors, *Proceedings of ICED 03, the 14th International Conference on Engineering Design*, pages 91–92, Stockholm, Sweden, 2003. Design Society.

J. Emblemsvåg. Lean project planning in shipbuilding. *Journal of Ship Production and Design*, 30(2):79–88, 2014. ISSN 2158-2874. doi: 10.5957/jspd.30.2.130054.

W. Herroelen. Generating robust project baseline schedules. In T. Klastorin, P. Gray, and H. J. Greenberg, editors, *OR Tools and Applications: Glimpses of Future Technologies*, TutORials in Operations Research, pages 124–144. INFORMS, 2007. doi: 10.1287/educ.1073.0038. Presented at the INFORMS Annual Meeting, November, 2007.

W. Herroelen and R. Leus. Project scheduling under uncertainty: Survey and research potentials. *European Journal of Operational Research*, 165(2):289–306, sep 2005. doi: 10.1016/j.ejor.2004.04.002.

W. Herroelen, E. Demeulemeester, and B. D. Reyck. A note on the paper "resource-constrained project scheduling: Notation, classification, models and methods" by brucker et al. *European Journal of Operational Research*, 128(3):679–688, feb 2001. doi: 10.1016/s0377-2217(99)00392-6.

W. Herroelen, R. Leus, and E. Demeulemeester. Critical chain project scheduling – Do not oversimplify. *Project Management Journal*, 33(4):46–60, 2002. URL https://lirias.kuleuven.be/handle/123456789/122992.

T. Jørgensen and S. W. Wallace. Distributed project management. In A. Løkketangen, editor, *Proceedings, fifth meeting of the Nordic section of the Mathematical Programming Society*, Molde, Norway, 1998.

T. Jørgensen and S. W. Wallace. Improving project cost estimation by taking into account managerial flexibility. *European Journal of Operational Research*, 127:239–251, 2000. doi: 10.1016/S0377-2217(99)00484-1.

P. Kall and S. W. Wallace. *Stochastic Programming*. John Wiley & Sons, Chichester, 1994.

A. J. King and S. W. Wallace. *Modeling with Stochastic Programming*. Springer Series in Operations Research and Financial Engineering. Springer, 2012. doi: 10.1007/978-0-387-87817-1.

O. Lambrechts, E. Demeulemeester, and W. Herroelen. Time slack-based techniques for robust project scheduling subject to resource uncertainty. *Annals of Operations Research*, 186(1):443–464, 2010. doi: 10.1007/s10479-010-0777-z.

F. Maggioni and S. W. Wallace. Analyzing the quality of the expected value solution in stochastic programming. *Annals of Operations Research*, 200(1):37–54, 2012. doi: 10.1007/s10479-010-0807-x.

P. W. G. Morris. *The Management of Projects*. Thomas Telford, London, 1994.

A. Rolstadås, J. K. Pinto, P. Falster, and R. Venkataraman. *Decision Making in Project management*. NTNU Engineering Series. Fagbokforlaget, 2014.

M. Rudberg and J. Wikner. Mass customization in terms of the customer order decoupling point. *Production Planning & Control*, 15(4):445–458, 2004. ISSN 1366-5871. doi: 10.1080/0953728042000238764.

D. Simchi-Levi, W. Schmidt, Y. Wei, P. Y. Zhang, K. Combs, Y. Ge, O. Gusikhin, M. Sanders, and D. Zhang. Identifying risks and mitigating disruptions in the automotive supply chain. *Interfaces*, 45(5):375–390, 2015. doi: 10.1287/inte.2015.0804.

D. Steinhauer and M. Soyka. Development and applications of simulation tools for one-of-a-kind production processes. In *Proceedings of the 2012 Winter Simulation Conference (WSC)*, pages 1–11, Berlin, Germany, 2012. IEEE. doi: 10.1109/wsc.2012.6465221.

D. Steinhauer and L. Wagner. Simulation-supported optimisation in maritime production planning. In M. Rabe, editor, *Advances in Simulation for Production and Logistics Applications*, pages 459–468. Fraunhofer IRB Verlag, 2008.

B. K. Thapalia, T. G. Crainic, M. Kaut, and S. W. Wallace. Single-commodity stochastic network design with multiple sources and sinks. *INFOR: Information Systems and Operational Research*, 49(3):193–211, 2011. doi: 10.3138/infor.49.3.003.

B. K. Thapalia, T. G. Crainic, M. Kaut, and S. W. Wallace. Single-commodity network design with random edge capacities. *European Journal of Operational Research*, 220 (2):394–403, 2012a.

B. K. Thapalia, T. G. Crainic, M. Kaut, and S. W. Wallace. Single source single-commodity stochastic network design. *Computational Management Science*, 9(1): 139–160, 2012b. doi: 10.1007/s10287-010-0129-0. Special issue on 'Optimal decision making under uncertainty'.

H. Vaagen and B. Aas. A multidisciplinary framework for robust planning and decision-making in dynamically changing engineering construction projects. In B. Grabot, B. Vallespir, S. Gomes, A. Bouras, and D. Kiritsis, editors, *Advances in Production Management Systems. Innovative and Knowledge-Based Production Management in a Global-Local World*, pages 515–522. Springer Berlin Heidelberg, 2014. ISBN http://id.crossref.org/isbn/978-3-662-44739-0. doi: 10.1007/978-3-662-44739-0_63. IFIP WG 5.7 International Conference, APMS 2014, Ajaccio, France.

H. Vaagen, S. W. Wallace, and M. Kaut. Modelling consumer-directed substitution. *International Journal of Production Economics*, 134(2):388–397, 2011. doi: 10.1016/j.ijpe.2009.11.012. special issue on 'Robust Optimization in Supply Chain Management'.

S. Van de Vonder, E. Demeulemeester, R. Leus, and W. Herroelen. Proactive-reactive project scheduling trade-offs and procedures. In J. Józefowska and J. Weglarz, editors, *Perspectives in Modern Project Scheduling*, volume 92 of *International Series in Operations Research & Management Science*, pages 25–51. Springer, 2006. doi: 10.1007/978-0-387-33768-5_2.

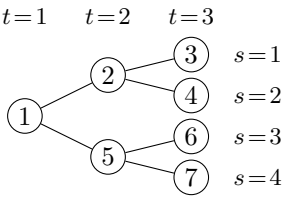


Figure 10: Example scenario tree

S. W. Wallace. Stochastic programming and the option to do it differently. *Annals of Operations Research*, 177(1):3–8, 2010. doi: 10.1007/s10479-009-0600-x.

D. Wynn and P. Clarkson. Design project planning, monitoring and re-planning through process simulation. In *Proceedings of ICED'09, the 17th International Conference on Engineering Design*, volume 1, pages 25–36, Stanford, USA, 2009. Design Society.

A. Model formulation

As usual, we formulate the model on a scenario tree that describes the development of uncertainty, such as the one in Figure 10. There are two different ways of associating the scenario-tree nodes to the time periods. One way is to let the nodes correspond the *points in time* when the decisions are made, so that time flows *between the nodes* (on the arcs of the scenario tree). This is a common practise in, for example, financial models, where the decision variables correspond to instantaneous decisions (buy/sell). There, it is also common to use the last-period nodes purely for accounting, i.e., no investments are done in these nodes. Hence, if we have the tree from Figure 10 with one-day periods, then node 1 corresponds to today morning, 2 and 5 to tomorrow morning, and the rest of the nodes (the leaves of the tree) to the morning after that—so the total time horizon is two days.

In our case, on the other hand, the model's variables represent actions that cover the whole period: start/stop decisions at the start of the period, resource-usage during the period, and variables describing the end-of-period status. In this case, it is more natural to associate the nodes with the periods, i.e., let the time flow *inside the nodes*, not between them. In this setting, there is no need for special nodes at the end of the horizon, since we have the end-of-period status variables. With this interpretation, the tree from Figure 10, with one-day periods, would mean that node 1 represents the whole first day, nodes 2 and 5 the second day, and the leaf nodes the third day—so the total time horizon is three days.

A.1. Notation

First, we have to decide how to address the nodes in the scenario tree. For this, there are two basic approaches: we can use the node number n directly, or we can use the time and scenario number (t, s) instead. The latter case has the advantage that the model looks

just like a standard deterministic one, with an extra scenario index. On the other hand, we create unnecessary copies of the variables that have to be 'bound together' using the so-called *nonanticipativity constraints*. For example, node 2 in Figure 10 belongs to scenarios 1 and 2, so we would need constraints $x_{(2,1)} = x_{(2,2)}$, for all decision variables x in the node.

For this reason, we have chosen the node-based (also called *compact*) formulation for our model. In this case, the structure of the scenario tree is described by providing, for each node of the tree, its parent node and probability (either absolute or relative to the parent). In addition, the expected value in the objective function is simply a sum of the nodes' contributions, weighted by their conditional probabilities.

A.1.1. Sets

Name	Description
$\mathcal{A}$	set of all activities
$\mathcal{N}$	set of all scenario-tree nodes
$\mathcal{R}$	set of all resources
$\mathcal{A}_{\mathrm{I}} \subset \mathcal{A}$	indicator activities – no duration
$\mathcal{A}_{\mathrm{R}} \subset \mathcal{A}$	real activities (with duration); $\mathcal{A}_{\mathrm{R}} = \mathcal{A} \setminus \mathcal{A}_{\mathrm{I}}$
$\mathcal{A}_{\mathrm{U}} \subset \mathcal{A}_{\mathrm{R}}$	activities that undo/reverse the results of other
$\mathcal{A}_{\mathrm{C}}(a) \subset \mathcal{A}_{\mathrm{R}}$	activities that $a \in \mathcal{A}_{\mathrm{R}}$ conflicts with (must be undone for a to start)
$\mathcal{D}_a^{\cap}$	set of activities that $a \in \mathcal{A}$ depends on; all must be finished
$\mathcal{D}_a^{\cup}$	set of activities that $a \in \mathcal{A}$ depends on; at least one must be finished
$\mathcal{N}_{\mathrm{L}} \subset \mathcal{A}$	set of leaf nodes, i.e., nodes without children
$\mathcal{N}_l^{\mathrm{P}} \subset \mathcal{A}$	set of nodes on path from the root to leaf node $l \in \mathcal{N}_{\mathrm{L}}$
$\mathcal{L}_r$	a set of intervals for piecewise-linear costs of resource $r \in \mathcal{R}$

A.1.2. Parameters

Name	Description
D_a	duration of activity a in scenario s
$\mathrm{P}(n)$	Probability of node $n \in \mathcal{N}$.
$n \dot{-} \Delta t$	predecessor of node n, Δt periods before node n
$n^{\dot{-}}$	parent node of n; special case of $n \dot{-} \Delta t$ with $\Delta t = 1$
$\mathrm{t}(n)$	period of node n
T_0	the first period
A_{F}	the final activity – finishing this marks the end of the project
$\bar{\mathrm{L}}_{r,l}$	upper bound of resource r in cost level $l \in \mathcal{L}_r$
$\mathrm{R}_{a,r}$	amount of resource $r \in \mathcal{R}$ used by activity $a \in \mathcal{A}_{\mathrm{R}}$ in each per.
$\mathrm{A}_{\mathrm{U}}(a) \in \mathcal{A}$	for $a \in \mathcal{A}_{\mathrm{U}}$, this is the activity a undoes/reverts

	periods			
var.	1	2	3	4
$x_{a,n}$	0	1	0	0
$y_{a,n}$	0	1	0	0
$z_{a,n}$	0	0	0	0
$u_{a,n}$	0	0	1	1
$v_{a,n}$	0	0	0	0

(a) Indicator act.

	time periods					
var.	1	2	3	4	5	6
$x_{a,n}$	0	1	0	0	0	0
$y_{a,n}$	0	0	0	1	0	0
$z_{a,n}$	0	1	1	1	0	0
$u_{a,n}$	0	0	0	0	1	1
$v_{a,n}$	0	0	0	0	0	0

(b) Real activity

	time periods					
var.	1	2	3	4	5	6
$x_{a,n}$	0	1	0	0	0	0
$y_{a,n}$	0	0	0	0	0	0
$z_{a,n}$	0	1	1	0	0	0
$u_{a,n}$	0	0	0	0	0	0
$v_{a,n}$	0	0	0	1	0	0

(c) Stopped activity

Figure 11: Illustration of the main binary variables. Fig. 11a illustrates an indicator activity triggered at the end of period 2, Fig. 11b a real activity of duration 3, started at period 2, and Fig. 11c the same activity, this time forced to stop after two periods.

U_a	Multiplier for duration of undo-activities
$\mathrm{C}^{\mathrm{R}}_{r,l}$	cost of using resource $r \in \mathcal{R}$ with cost level $l \in \mathcal{L}_r$
$\mathrm{C}^{\mathrm{E}}_{t}$	cost of finishing the whole project at the end of t

A.1.3. Variables

Name	Description	Range
$x_{a,n}$	has activity a started at the start of the period of node n?	$\{0,1\}$
$y_{a,n}$	has activity a finished at the end of the period of node n?	$\{0,1\}$
$z_{a,n}$	is activity a running during the period of node n?	$\{0,1\}$
$u_{a,n}$	has activity a been finished by the start of the period of node n?	$\{0,1\}$
$v_{a,n}$	has activity a been stopped at start of the period of node n?	$\{0,1\}$
$w^{l}_{r,n}$	amount of resource r at cost-level l used in node n	≥ 0

The function of these variables is illustrated in Figure 11. Note that indicator activities have zero durations and therefore $z_{a,n} = 0$ for all nodes. In addition, $y_{a,n} = 1$ only in the node where the project has successfully finished, while $u_{a,n} = 1$ also in all the node's successors. Finally, since the decision to stop is done at the start of the period, $v_{a,n} = 1$ implies $z_{a,n} = 0$. In addition, a stopped activity does not get marked as finished, i.e., $u_{a,n} = 0$ in all nodes.

From a conceptual point of view, it is important to realize that the only 'real' decision variables are $x_{a,n}$ and $v_{a,n}$; the rest are auxiliary variables whose values are determined by the 'real' ones.

A.2. The model

In this section we provide the complete formulation of the optimization model. In the model, we use the following notation: $\lceil x \rceil$ denotes the value of x rounded up to the nearest integer, $|\mathcal{S}|$ denotes the cardinality (size) of set $\mathcal{S}$ and $\text{if}\big(A \,|\, x \,|\, y\big)$ means 'if A then x else y'.

Furthermore, we use the convention that invalid subscripts evaluate to zero; for example $x_{a,n\dot{-}\Delta t} = 0$ for nodes n with $\mathrm{t}(n) - \Delta t < \mathrm{T}_0$. We also omit '$n \in \mathcal{N}$' from the constraint descriptions, since all constraints are valid for all nodes. This is to increase readability.

A.2.1. Objective function

The objective is to minimize the expected costs, consisting of the resource-usage costs and extra penalty term depending on the finishing time of the whole project, i.e., end time of the final activity $\mathrm{A_F}$.

$$\text{minimize} \sum_{n \in \mathcal{N}} \mathrm{P}(n) \Big[\sum_{\substack{r \in \mathcal{R} \\ l \in \mathcal{L}_r}} \mathrm{C}^{\mathrm{R}}_{r,l}\, w^{l}_{r,n} + \mathrm{C}^{\mathrm{E}}_{\mathrm{t}(n)}\, y_{\mathrm{A_F},n} \Big] \tag{1}$$

A.2.2. Activity-tracking constraints

$$y_{a,n} = x_{a,n} \qquad a \in \mathcal{A}_{\mathrm{I}} \tag{2}$$

$$y_{a,n} \le x_{a,n\dot{-}(\mathrm{D}_a - 1)} \qquad a \in \mathcal{A}_{\mathrm{R}} \setminus \mathcal{A}_{\mathrm{U}} \tag{3}$$

$$y_{a,n} \le 1 - \sum_{\Delta t=0}^{\mathrm{D}_a - 1} v_{a,n\dot{-}\Delta t} \qquad a \in \mathcal{A}_{\mathrm{R}} \setminus \mathcal{A}_{\mathrm{U}} \tag{4}$$

$$y_{a,n} \le \sum_{\Delta t=1}^{\lceil \mathrm{U}_b \mathrm{D}_b \rceil} x_{a,n\dot{-}\Delta t} \qquad a \in \mathcal{A}_{\mathrm{U}}, b = \mathrm{A_U}(a) \tag{5}$$

$$\mathrm{U}_b \mathrm{D}_b\, y_{a,n} \le \sum_{\Delta t=0}^{\lceil \mathrm{U}_b \mathrm{D}_b \rceil - 1} z_{a,n\dot{-}\Delta t} - \mathrm{U}_b \sum_{\Delta t=1}^{\mathrm{t}(n) - \mathrm{T}_0} z_{b,n\dot{-}\Delta t} + \mathrm{U}_b \mathrm{D}_b \qquad a \in \mathcal{A}_{\mathrm{U}}, b = \mathrm{A_U}(a) \tag{6}$$

$$x_{a,n} \le u_{b,n} + \sum_{\Delta t=0}^{\mathrm{t}(n) - \mathrm{T}_0 - 1} v_{b,n\dot{-}\Delta t} \qquad a \in \mathcal{A}_{\mathrm{U}}, b = \mathrm{A_U}(a) \tag{7}$$

$$u_{a,n} = u_{a,n\dot{-}} + y_{a,n\dot{-}} \qquad a \in \mathcal{A} \tag{8}$$

$$z_{a,n} = x_{a,n} - v_{a,n} + z_{a,n\dot{-}} - y_{a,n\dot{-}} \qquad a \in \mathcal{A}_{\mathrm{R}} \tag{9}$$

$$v_{a,n} \le z_{a,n\dot{-}} \qquad a \in \mathcal{A}_{\mathrm{R}} \setminus \mathcal{A}_{\mathrm{U}} \tag{10}$$

Constraints (2)–(5) determine the activity end: (2) for indicator activities that have zero duration and (3)–(4) for real activities, except for the undo-activities modelled by constraints (5)–(6). Constraints (3) and (5) link the activity end to its start time, while (4) ensures that a stopped activity does not get marked as finished and (6) models the

dynamic duration of the undo-activities. In addition, (7) ensures that an undo activity can start only if the activity it reverts has finished or been stopped. Note that if an activity cannot be stopped, we can drop (4) and restate (3) as an equality.

Constraint (8) controls the indicator $u_{a,n}$, saying that an activity is finished in a node if it was finished in the node's parent, or if it ended there.

Similarly, constraint (9) states that an activity runs in a node if it either started there, or was running in the parent node and neither finished nor stopped. Finally, (10) says that an activity can be stopped only if it runs.

A.2.3. Dependencies and conflicts between activities

$$x_{a,n} \leq u_{b,n} + \mathrm{if}\big(a \in \mathcal{A}_\mathrm{I} \,|\, y_{b,n} \,|\, 0\big) + \mathrm{if}\Big(b \in \mathcal{A}_\mathrm{U} \,|\, 1 - \sum_{\Delta t=1}^{\mathrm{t}(n)-\mathrm{T}_0} x_{\mathrm{A_U}(b),n\dot{-}\Delta t} \,|\, 0\Big) \qquad a \in \mathcal{A} \setminus \mathcal{A}_\mathrm{U}, b \in \mathcal{D}_a^\cap \tag{11}$$

$$x_{a,n} \leq \sum_{b \in \mathcal{D}_a^\cup} \Big[u_{b,n} + \mathrm{if}\big(a \in \mathcal{A}_\mathrm{I} \,|\, y_{b,n} \,|\, 0\big) + \mathrm{if}\Big(b \in \mathcal{A}_\mathrm{U} \,|\, 1 - \sum_{\Delta t=1}^{\mathrm{t}(n)-\mathrm{T}_0} x_{\mathrm{A_U}(b),n\dot{-}\Delta t} \,|\, 0\Big)\Big] \qquad a \in \mathcal{A} \setminus \mathcal{A}_\mathrm{U} : |\mathcal{D}_a^\cup| > 0 \tag{12}$$

$$x_{a,n} \leq u_{b,n} + \sum_{\Delta t=0}^{\mathrm{t}(n)-\mathrm{T}_0} v_{b,n\dot{-}\Delta t} \qquad a \in \mathcal{A}_\mathrm{U}, b \in \mathrm{A_U}(a) \tag{13}$$

$$x_{a,n} \leq u_{b,n} + 1 - \sum_{\Delta t=0}^{\mathrm{t}(n)-\mathrm{T}_0} x_{c,n\dot{-}\Delta t} \qquad a \in \mathcal{A}, c \in \mathcal{A}_\mathrm{C}(a), b \in \mathcal{A}_\mathrm{U} : \mathrm{A_U}(b) = c \tag{14}$$

Constraints (11)–(12) control the start of dependent activities for the two types of dependencies: the dependent activity can start if the activities it depends on have finished and we have not started undoing/reverting them (if applicable). The first conditional part in each right-hand side is there because an indicator activity has zero duration, so it can start (and end) in the same period where the activity it depends on ended.

Constraints (13)–(14) model conflicts between activities: (13) states that the undo-activity can start only if the activity being undone either finished or was stopped and (14) ensures that activity a cannot start if we had started any conflicting activity c, unless c was undone using the appropriate activity b.

A.2.4. Stochastic dependencies

$$x_{a,n} \leq u_{b,n} + \mathrm{if}\big(a \in \mathcal{A}_\mathrm{I} \,|\, y_{b,n} \,|\, 0\big) \qquad a \in \mathcal{A}, b \in \mathcal{D}^{\cap}_{a,n} \tag{15}$$

$$x_{a,n} \leq \sum_{b \in \mathcal{D}^{\cap}_{a,n}} \big[u_{b,n} + \mathrm{if}\big(a \in \mathcal{A}_\mathrm{I} \,|\, y_{b,n} \,|\, 0\big)\big] \qquad a \in \mathcal{A} : |\mathcal{D}^{\cup}_{a,n}| > 0 \tag{16}$$

$$x_{a,n} = 0 \qquad a \in \mathcal{A}, b \in \mathcal{D}^{\cap}_{a,n} : \big|l \in \mathcal{N}_\mathrm{L} : n \in \mathcal{N}^\mathrm{P}_l \;\&\; b \in \mathcal{D}^{\cap}_{a,l}\big| < \big|l \in \mathcal{N}_\mathrm{L} : n \in \mathcal{N}^\mathrm{P}_l\big| \tag{17}$$

Constraints (15) and (16) are the stochastic counterparts of (11) and (12), simplified with an assumption that the activities a depends on cannot be undone. This is because in our examples, the stochastic dependency is on indicator activities that cannot be undone[3].

Constraint (17) prevents the situation where the stochastically-dependent activity is declared finished before the stochasticity is fully revealed, i.e., if there is a non-zero probability of an incoming design change.

A.2.5. Resources and timing

$$\sum_{a \in \mathcal{A}_\mathrm{R}} \mathrm{R}_{a,r}\, z_{a,n} \leq \sum_{l \in \mathcal{L}_r} w^l_{r,n} \qquad r \in \mathcal{R} \tag{18}$$

$$u_{\mathrm{A_F},l} + y_{\mathrm{A_F},l} = 1 \qquad l \in \mathcal{N}_\mathrm{L} \tag{19}$$

Constraint (18) ensures that all the resources used by activities get counted by the resource-usage variables, and hence appear in the objective function. Finally, (19) requires the final activity $\mathrm{A_F}$, and therefore the whole project, to finish in each scenario—otherwise, the model would not do anything, since we are minimizing the costs.

[3] If one had stochastic dependence on a real activity, one would have to expand the constraints or introduce an extra indicator variable