4-2015

# Project planning with alternative technologies in uncertain environments

Stefan CREEMERS

Bert DE REYCK
*Singapore Management University*, bdreyck@smu.edu.sg

Roel LEUS

European Journal of Operational Research 242 (2015) 465–476

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Stochastics and Statistics

# Project planning with alternative technologies in uncertain environments

Stefan Creemers [a], Bert De Reyck [b], Roel Leus [c,*]

[a] Management Department, IESEG School of Management, Lille, France
[b] Department of Management Science & Innovation, University College London, United Kingdom
[c] Research Group ORSTAT, Faculty of Economics and Business, KU Leuven, Belgium

ARTICLE INFO

*Article history:*
Received 26 August 2013
Accepted 5 November 2014
Available online 21 November 2014

*Keywords:*
Project scheduling
Uncertainty
Net present value
Alternative technologies
Stochastic activity durations

ABSTRACT

We investigate project scheduling with stochastic activity durations to maximize the expected net present value. Individual activities also carry a risk of failure, which can cause the overall project to fail. In the project planning literature, such technological uncertainty is typically ignored and project plans are developed only for scenarios in which the project succeeds. To mitigate the risk that an activity's failure jeopardizes the entire project, more than one alternative may exist for reaching the project's objectives. We propose a model that incorporates both the risk of activity failure and the possible pursuit of alternative technologies. We find optimal solutions to the scheduling problem by means of stochastic dynamic programming. Our algorithms prescribe which alternatives need to be explored, and how they should be scheduled. We also examine the impact of the variability of the activity durations on the project's value.



## 1. Introduction

Projects in many industries are subject to considerable uncertainty, due to many possible causes. Factors influencing the completion date of a project include activities that are required but that were not identified beforehand, activities taking longer than expected, activities that need to be redone, resources being unavailable when required, late deliveries, etc. In research and development (R&D) projects, there is also the risk that activities may fail altogether, requiring the project to be halted completely. This risk is often referred to as *technical risk*. In this text, we focus on two main sources of uncertainty in R&D projects, namely uncertain activity durations and the possibility of activity failure: we incorporate the concept of *activity success or failure* into the analysis of projects with stochastic activity durations, where the successful completion of an activity can correspond to a technological discovery or scientific breakthrough. We examine the impact of these two factors on optimal planning strategies that maximize the project's value, and on its value itself.

This work is a continuation of De Reyck and Leus (2008), where an algorithm is developed for project scheduling with uncertain activity outcomes and where project success is achieved only if all individual activities succeed. Reference De Reyck and Leus (2008) constituted the first description of an optimal approach for handling activity failures in project scheduling, but neither stochastic activity durations nor the possibility of pursuing multiple alternatives for the same result, and the inherent possibility of activity selection, were accounted for. Earlier work studied optimal procedures for special cases; see Chun (1994), for instance. Other references relevant to this text stem from the discipline of chemical engineering, mainly the work by Grossmann and his colleagues (e.g., Jain & Grossmann, 1999; Schmidt & Grossmann, 1996), who studied the scheduling of failure-prone new-product development (NPD) testing tasks when non-sequential testing is admitted. They point out that in industries such as chemicals and pharmaceuticals, the failure of a single required environmental or safety test may prevent a potential product from reaching the marketplace, which has inspired our modeling of possible activity and project failure. Therefore, our models are also of particular interest to drug-development projects, in which stringent scientific procedures have to be followed in distinct stages to ensure patient safety, before a medicine can be approved for production. Such projects may need to be terminated in any of these stages, either because the product is revealed not to have the desired properties or because of harmful side effects. Illustrations of modeling pharmaceutical projects, with a focus on resource allocation, can be found in Gittins and Yu (2007) and Yu and Gittins (2008).

Due to the risk of activity failure resulting in overall project failure, it has been suggested that R&D projects should explore multiple alternative ways for developing new products (Sommer & Loch, 2004). To mitigate the risk that an individual activity's failure jeopardizes the

* Corresponding author. Tel.: +32 16 32 69 67; fax: +32 16 32 66 24.
*E-mail addresses:* s.creemers@ieseg.fr (S. Creemers), bdereyck@ucl.ac.uk (B. De Reyck), Roel.Leus@kuleuven.be (R. Leus).

http://dx.doi.org/10.1016/j.ejor.2014.11.014
0377-2217/

entire project, we model projects in which the same (intermediate or final) outcome can be pursued in several different ways, where one success allows the project to continue. The different attempts can be multiple trials of the same procedure or the pursuit of different alternative ways to achieve the same outcome, e.g., the exploration of alternative technologies. Following Baldwin and Clark (2000), a unit of alternative interdependent tasks with a distinguished deliverable will be called a *module*.

Project profitability is often measured by the project's net present value (NPV), the discounted value of the project's cash flows. This NPV is affected by the project schedule and therefore, the timing of expenditures and cash inflows has a major impact on the project's financial performance, especially in capital-intensive industries. The goal of this article is to find optimal scheduling strategies that maximize the expected NPV (eNPV) of the project while taking into account the activity costs, the cash flows generated by a successful project, the variability in the activity durations, the precedence constraints, the likelihood of activity failure and the option to pursue multiple trials or technologies. Thus, this article extends the work of Buss and Rosenblatt (1997), Benati (2006), Sobel, Szmerekovsky, and Tilson (2009) and Creemers, Leus, and Lambrecht (2010), who focus on duration risk only, and of Schmidt and Grossmann (1996), Jain and Grossmann (1999) and De Reyck and Leus (2008), who look into technical risk only (although Schmidt and Grossmann (1996) also explore the possibility of introducing multiple discrete duration scenarios).

Our contributions are fourfold: (1) we introduce and formulate a generic model for optimal scheduling of R&D activities with stochastic durations, non-zero failure probabilities and modular completion subject to precedence constraints; to the best of our knowledge, such a model has never been studied before; (2) we develop a dynamic-programming recursion to determine an optimal policy for executing the project while maximizing the project's eNPV, extending the algorithm of Creemers et al. (2010) with activity failures, multiple trials and phase-type (PH) distributed activity durations instead of exponentials; (3) we conduct numerical experiments to demonstrate the computational capabilities of the algorithm; and (4) we examine the impact of activity duration risk on the optimal scheduling policy and project values. Interestingly, our findings indicate that higher operational variability does not always lead to lower project values, meaning that (sometimes costly) variance reduction strategies are not always advisable. To the best of our knowledge, this is the first article to numerically support such a recommendation.

The remainder of this text is organized as follows. In Section 2, we provide the necessary definitions and a detailed problem statement. We produce solutions by means of a backward dynamic-programming recursion for a Markov decision process, which is discussed in Section 3. Section 4 reports on our computational performance on a representative set of test instances. In Section 5, a computational experiment is described in which we examine the effect of activity duration variability on the eNPV of a project and Section 6 evaluates two different choices for the policy class to be considered. Section 7 contains a brief summary of the text.

## 2. Definitions and problem statement

### 2.1. Stochastic project scheduling

A project consists of a set of activities $N = \{0, \ldots, n\}$. The execution of a project with stochastic components (in our case, stochastic activity outcomes and durations) is a dynamic decision process. A solution, therefore, cannot be represented by a schedule but takes the form of a *policy*: a set of decision rules defining *actions* at *decision times*, which may depend on the prior outcomes. Decision times are typically the start of the project and the completion times of activities; a tentative next decision time can also be specified by the decision maker. An action entails the start of a precedence-feasible set of activities (see Section 2.2 for a statement of the precedence constraints). In this way, a schedule is constructed gradually as time progresses. Next to the information available at the start of the project, a decision at time $t$ can only use information on duration realizations and activity outcomes that has become available before or at time $t$; this is the so-called *non-anticipativity constraint*. Activities should be executed without interruption.

Each activity $i \in N\backslash\{n\}$ has a probability of technical success $p_i$; we assume that $p_0 = 1$. We do not consider (renewable or other) resource constraints and assume the outcomes of the different tasks to be independent. We define a *success (state) vector* as an $n$-component binary vector $\mathbf{x} = (x_0, x_1, \ldots, x_{n-1})$, with one component associated with each activity in $N \setminus \{n\}$. We let $X_i$ represent the Bernoulli random variable with parameter $p_i$ as success probability for each activity $i$, and we write $\mathbf{X} = (X_0, X_1, \ldots, X_{n-1})$. Information on an activity's success (the realization of $X_i$) becomes available only at the end of that activity. We say that $\mathbf{x}$ is a *realization* or *scenario* of $\mathbf{X}$. The duration $D_j \geq 0$ of each activity $j$ is also a stochastic variable; the vector $(D_0, D_1, \ldots, , D_n)$ is denoted by $\mathbf{D}$. We use lowercase vector $\mathbf{d} = (d_0, \ldots, , d_n)$ to represent one particular realization of $\mathbf{D}$, and we assume $Pr[D_0 = 0] = Pr[D_n = 0] = 1$.

We assume that all activity cash flows during the development phase are non-positive, which is typical for R&D projects: the (known) cash flow associated with the execution of activity $i \in N\backslash\{n\}$ is represented by the integer value $c_i \leq 0$ and is incurred at the start of the activity. We choose $c_0 = 0$. If the project is successful (see Section 2.2 for the specific conditions under which this is true) then the final activity $n$ can be executed. This corresponds with obtaining an end-of-project payoff $C \geq 0$, which is received at the start of activity $n$ (which is also its completion time). The value $s_i \geq 0$ represents the starting time of activity $i$; we call the $(n+1)$-vector $\mathbf{s} = (s_0, s_1, \ldots, s_n)$ a schedule, with $s_i \geq 0$ for all $i \in N$. We assume $s_0 = 0$ in what follows: the project starts at time zero. The value $s_i = +\infty$ means that activity $i$ will not be executed at all.

We follow Igelmund and Radermacher (1983), Möhring (2000) and Stork (2001), who study project scheduling with resource constraints and stochastic activity durations, in interpreting every scheduling policy $\Pi$ as a function $\mathbb{R}_{\geq}^{n-1} \times \mathbb{B}^n \to \mathbb{R}_{\geq}^{n+1}$, with $\mathbb{R}_{\geq}$ the set of non-negative reals and $\mathbb{B} = \{0, 1\}$. The function $\Pi$ maps given samples $(\mathbf{d}, \mathbf{x})$ of activity durations and success vectors to vectors $\mathbf{s}(\mathbf{d}, \mathbf{x}; \Pi)$ of feasible activity starting times (schedules). For a given duration scenario $\mathbf{d}$, success vector $\mathbf{x}$ and policy $\Pi$, $s_n(\mathbf{d}, \mathbf{x}; \Pi)$ denotes the makespan of the schedule, which coincides with project completion. Note that not all activities need to be completed (or even started) by $s_n$, nor that the realization of all $X_i$'s needs to be known.

We compute the NPV for schedule $\mathbf{s}$ as

$$f(\mathbf{s}) = Ce^{-rs_n} + \sum_{\substack{i=1 \\ s_i \neq \infty}}^{n-1} c_i e^{-rs_i}, \tag{1}$$

with $r$ a continuous discount rate chosen to represent the time value of money: the present value of a cash flow $c$ incurred at time $t$ equals $ce^{-rt}$, where $e$ is the base of the natural logarithm. Our goal in this article is to select a policy $\Pi^*$ that maximizes $\mathbb{E}[f(\mathbf{s}(\mathbf{D}, \mathbf{X}; \Pi))]$, with $\mathbb{E}[\cdot]$ the expectation operator with respect to $\mathbf{D}$ and $\mathbf{X}$; we write $\mathbb{E}[f(\Pi)]$, for short. The generality of this problem statement suggests that optimization over the class of all policies is probably computationally impractical. We therefore restrict our optimization to a subclass that has a simple combinatorial representation and where decision points are limited in number: our solution space $\mathcal{P}$ consists of all policies that start activities only at the end of other activities (activity 0 is started at time 0). The solution space also contains policy $\Pi_0$, which corresponds with immediate abandonment of the project (formally, all starting times apart from $s_0$ are set to infinity), which will be preferable when $C$ is not large enough compared to the costs of the

activities: then it is better simply not to undertake the project at all, with objective value 0.

### 2.2. Modular projects

Modularity means splitting the design and production of technologies into independent subparts (Baldwin & Clark, 2000). This has benefits towards inventory management for mass-produced items via techniques such as commonality and postponement (Chopra & Meindl, 2013), but also with respect to the duration and chances of success of a product development project by itself: in this setting, a module is a set of alternative development activities that pursue a similar target, representing repeated trials or technological alternatives. Lenfle (2011) provides a thorough literature review of the management of projects via modules, and he points out that different alternatives can be pursued either in parallel or sequentially, or following a mix of both strategies. Obviously, management can also decide *not* to pursue certain alternatives, for instance because their cost is too high compared to their expected benefits.

Lenfle refers to the Manhattan Project as one prime example where such techniques were applied (for instance, multiple alternative bomb assembly designs were tested simultaneously). Weitzman (1979) brings up the evaluation and selection of alternative suppliers for some commodity as one possible practical application. Nelson (1961) cites a RAND working paper on the development of a new microwave relay system at Bell Telephone Laboratories, where the eventual success of the development was greatly facilitated by running multiple approaches in parallel to solving some of the encountered development problems. Granot and Zuckerman (1991) refer to the development of nylon at DuPont, where numerous polymers were tested one by one before the discovery of a suitable polyamide. Abernathy and Rosenbloom (1969) evaluate the merits of a parallel strategy at a critical point in a million-dollar advanced power-supply development project. In the context of the development of an AIDS vaccine, Ding and Eliashberg (2002) note that 'In many new product development (NPD) situations, the development process is characterized by uncertainty, and no single development approach will necessarily lead to a successful product. To increase the likelihood of having at least one successful product, multiple approaches may be simultaneously funded at the various NPD stages.'

In this text, we will take the modular structure of the project as given, assuming that an appropriate project network design and initial selection of development alternatives have already been set out. Formally, the set of modules is $M = \{0, \ldots, m\}$; each module $i \in M$ contains the activities $N_i \subset N$, and the set of modules constitutes a partition of $N$: $N = \bigcup_{i \in M} N_i$ and $N_i \cap N_j = \varnothing$ if $i \neq j$. $A$ is a (strict) partial order on $M$, i.e., an irreflexive and transitive relation, which represents technological precedence constraints. (Dummy) modules 0 and $m$ represent the start and the end of the project, respectively; they are the (unique) least and greatest element of the partially ordered set $(M, A)$ and are assumed to contain only one (dummy) activity, indexed by 0 and $n$, respectively. On the activities within each module $i$, we also impose a partial order $B_i$, to allow for modeling precedence requirements between these activities. In drug development, for example, when a certain module is needed to show the effectiveness of a drug, two precedence-related activities could represent the repeated measurement of the beneficial effects of the drug: the first test is performed after one week; the effects after two weeks will only be measured if first the effects after one week are inconclusive (Coolen, Wei, Talla Nobibon, & Leus, 2011). Alternatively, trials may be repeated in different doses and with different test subjects (Huysmans, Coolen, Talla Nobibon, & Leus, 2012). Precedence constraints within modules may also represent fallback options for project failure, as 'contingency plans': plans devised for an outcome different from expected. Comparable modeling choices were made in Coolen et al. (2011) and in Huysmans et al. (2012), but without discounting the cash flows, in which case durations become irrelevant and scheduling all activities sequentially is a dominant choice.

For convenience, we associate a completion time $h_i(\mathbf{s}; \mathbf{d}, \mathbf{x})$ with each module $i$, in the following way (here and later, we omit the arguments if no misinterpretation is possible): $h_i = \min_{j \in N_i | x_j = 1} \{s_j + d_j\}$, coinciding with the earliest completion of a successful activity contained in the module; if the min-operator optimizes over the empty set then we define $h_i := +\infty$, meaning that the module is never successfully completed. For a given success vector $\mathbf{x}$ and durations $\mathbf{d}$, we then say that a schedule $\mathbf{s}$ is *feasible* if the following conditions are fulfilled:

$$h_i \leq s_j \quad \forall (i,k) \in A, \forall j \in N_k \tag{2}$$

$$s_i + d_i \leq s_j \quad \forall k \in M, \forall (i,j) \in B_k \tag{3}$$

Eq. (2) are inter-module precedence constraints, which imply that a necessary condition for the start of an activity $j \in N_k$ is success for all the predecessor modules $i$ of the module $k$ to which $j$ belongs, where a module is said to be successful if at least one of its constituent activities succeeds. Eq. (3) are intra-module constraints: an activity $j$ can only be started when all predecessor activities $i$ in the same module have been completed, and its execution would obviously be useful only if all those predecessors failed. An activity's starting time equal to infinity corresponds to not executing the activity and therefore not incurring any related expenses, or in case of activity $n$, not receiving the project payoff. Consequently, the project payoff is achieved $(s_n \neq \infty)$ only if every module is successful.

The classic PERT problem (Adlakha & Kulkarni, 1989; Elmaghraby, 1977; Kulkarni & Adlakha, 1986; Ludwig, Möhring, & Stork, 2001) aims at characterizing the random variable $s_n(\mathbf{D}, \mathbf{1}; \Pi^{ES})$, where policy $\Pi^{ES}$ starts all activities as early as possible, each module contains only one activity, and $\mathbf{1}$ is an $n$-vector with value 1 in all components. Contrary to the makespan, however, NPV is a non-regular measure of performance: starting activities as early as possible is not necessarily optimal, since the $c_i$ are usually negative.

### 2.3. Illustration

Fig. 1 illustrates the foregoing definitions and problem statement. This project consists of seven activities, $N = \{0, 1, 2, 3, 4, 5, 6\}$, where 0 and $n = 6$ are dummies. There are five modules, so $m = 4 : N_0 = \{0\}, N_1 = \{1, 2, 3\}, N_2 = \{4\}, N_3 = \{5\}$ and $N_4 = \{6\}$. In the example, $B_1 = \{(1, 3), (2, 3)\}$. Note that Fig. 1 actually shows the transitive reduction of $A$: the order relation $A$ also contains elements such as $(0, 2)$ and $(1, 4)$, while the arcs $N_0 \to N_2$ and $N_1 \to N_4$ are not included in the figure.

A policy $\Pi_{12}$ for this project is the following: start the project at time 0 $(s_0 = 0)$ and immediately initiate activities 1 and 2 $(s_1 = s_2 = 0)$. If $X_1 = X_2 = 0$ then abandon the project: set $s_3 = s_4 = s_5 = s_6 = +\infty$. Otherwise, module $N_1$ completes successfully. In that case, start both activities 4 and 5 upon the successful completion of activity 1 or 2 (whichever is the earliest), and terminate the project if either 4 or 5 fails. Note that under policy $\Pi_{12}$, activity 3 is never started, and we effectively include activity *selection* as part of the decisions to be made. Represented as a function, $\Pi_{12}$ entails the following

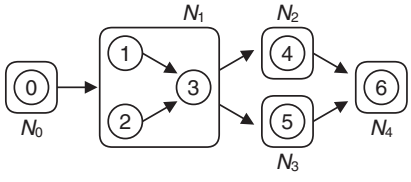



**Fig. 1.** Example module network.

mapping:

$$(d_1, d_2, d_3, d_4, d_5, x_0, x_1, x_2, x_3, x_4, x_5) \mapsto (0, 0, 0, \infty, h_1, h_1, \max\{h_2; h_3\}),$$

with $h_1 = \min_{j=1,2;x_j=1}\{d_j\}$ and $h_1 = \infty$ if $x_1 = x_2 = 0$, $h_2 = h_1 + d_4$ if $x_4 = 1$ and $h_2 = \infty$ if $x_4 = 0$, and $h_3 = h_1 + d_5$ if $x_5 = 1$ and $h_3 = \infty$ if $x_5 = 0$.

## 3. Markov decision process

### 3.1. Policy class

In the literature, the input parameters of the PERT problem are often referred to as a PERT network, and a PERT network with independent and exponentially distributed activity durations is also called a *Markovian PERT network*. For Markovian PERT networks, Kulkarni and Adlakha (1986) describe an exact method for deriving the distribution and moments of the earliest project completion time using continuous-time Markov chains (CTMCs), where it is assumed that each activity is started as soon as its predecessors are completed (an early-start schedule).

Buss and Rosenblatt (1997), Sobel et al. (2009) and Creemers et al. (2010) investigate an eNPV objective and use the CTMC described by Kulkarni and Adklakha as a starting point for their algorithms. A similar problem is studied by Benati (2006), who proposes a heuristic scheduling rule. Next to stochastic durations, Buss and Rosenblatt (1997) also consider activity delays. These studies, however, all assume success in all activities and an exponential distribution for all durations and they also imply the requirement that all activities be executed.

De Reyck and Leus (2008) study project scheduling with known activity durations but with uncertain activity outcomes. In that article, if an activity A ends no later than the start of another activity B then knowledge of the outcome (success or failure) of A can sometimes be used to avoid incurring the cost for B, since a failure in A would allow abandoning the project, but payment for B cannot be avoided when B has already started before the outcome of A is discovered. For a given selection of such 'information flows' between activities (under the form of additional precedence constraints), a late-start schedule is then optimal when the activity durations are known. Unfortunately, late-start scheduling is difficult to implement in case of stochastic durations, and Sobel et al. (2009) implicitly restrict their attention to scheduling policies that start activities only at the end of other activities. Buss and Rosenblatt (1997) partially relax this restriction by starting an activity only after a fixed time interval (delay), but they do not decide which sets of activities to start at what time (all eligible activities are started as soon as possible after their delay). Creemers et al. (2010) study the same problem as Sobel et al. (2009) and achieve significant computational performance improvements.

In this article, we also propose to restrict the attention to policies that start activities at the completion time of other activities. This can be seen to be a dominant set of policies for those cases in which the project payoff is sufficiently large relative to the costs associated with the intermediate activities, because the benefit of delaying the payment of an activity would then be more than offset by the disadvantage of the higher possibility of delay in obtaining the payoff; this reasoning holds for any discount rate $r > 0$. The generalization in which activity starting times are delayed by a fixed offset beyond their earliest starting time poses significant computational challenges (cf. Buss & Rosenblatt, 1997). The models and algorithms in this article can be extended so that activities can also be started when other activities are 'underway,' and in Section 6, we describe our findings for an experiment where we consider the possible start of new activities after each phase in the PH distribution of each ongoing activity (a setting that gives rise to a larger policy class, hence a larger search space). The experiment indicates that the average improvement in the objective function is minor (up to 0.3 percent of the payoff at most, depending on the settings). We recognize that the practical relevance of this larger policy class can obviously be questioned, and the experiment should merely be seen as an approximation of the setting where activities can be started whenever the decision maker chooses. We conclude that only starting activities at the completion time of other activities is not a very restrictive decision, under the settings tested.

Below, we extend the work of Creemers et al. (2010) to accommodate PH-distributed activity durations, possible activity failures and a modular project network, allowing also for activity selection. We first study the special case of exponential activity durations (Section 3.2), followed by an illustration (Section 3.3) and by a treatment of more general distributions (Section 3.4).

### 3.2. The exponential case

For the moment, we assume each duration $D_i$ to be exponentially distributed with rate parameter $\lambda_i = 1/\mathbb{E}[D_i]$ $(i = 1, \ldots, n-1)$; we consider more general distributions in Section 3.4. At any time instant $t$, an activity's status is either *idle* (not yet started), *active* (being executed), or *past* (successfully finished, failed, or considered redundant because its module is completed). Let $I(t)$, $Y(t)$ and $P(t)$ represent the activities in $N$ that are idle, active and past, respectively; these three sets are mutually exclusive and $I(t) \cup Y(t) \cup P(t) = N$. The state of the system is defined by the status of the individual activities and is represented by a triplet $(I, Y, P)$. State transitions take place each time an activity becomes past and are determined by the policy at hand. The project's starting conditions are $Y(0) = \{0\}$ and $I(0) = N \setminus \{0\}$, while the condition for *successful* completion of the project is $P(t^*) = N$, where $t^*$ represents the project completion time.

The problem of finding an optimal scheduling policy corresponds to optimizing a discounted criterion in a continuous-time Markov decision chain (CTMDC) on the state space $Q$, with $Q$ containing all the states of the system that can be visited by the transitions (which are called *feasible* states); the decision set is described below. We apply a backward stochastic dynamic-programming (SDP) recursion to determine optimal decisions based on the CTMC described in Kulkarni and Adlakha (1986). The key instrument of the SDP recursion is the *value function* $F(\cdot)$, which determines the expected NPV of each feasible state at the time of entry of the state, conditional on the hypothesis that optimal decisions are made in all subsequent states and assuming that all 'past' modules (with all activities past) were successful. In the definition of the value function $F(I, Y)$, we supply sets $I$ and $Y$ of idle and active activities as parameters (which uniquely determines the past activities). When an activity finishes, three different state transitions can occur: (1) activity $j \in N_i$ completes successfully; (2) activity $j \in N_i$ fails and another activity $k \in N_i$ is still idle or active; (3) activity $j \in N_i$ fails and all other activities $k \in N_i$ have already failed (or it is the only activity in the module).

We define the order $B^*$ on set $N$ to relate activities that do not necessarily belong to the same module, as follows:

$$(i, j) \in B^* \Leftrightarrow (\exists B_m : (i, j) \in B_m) \vee (\exists (l, m) \in A : i \in N_l \wedge j \in N_m).$$

We call an activity $j$ *eligible* at time $t$ if $j \in I(t)$ and $\forall (k, j) \in B^* : k \in P(t)$. Let $E(I, Y) \subset N$ be the set of eligible activities for given sets $I$ and $Y$ of idle and active activities. Upon entry of a state $(I, Y, P) \in Q$, a decision needs to be made whether or not to start eligible activities in $E(I, Y)$ and if so, which. If no activities are started, a transition towards another state occurs at the first completion of an element of $Y$. Not starting any activities while there are no active activities left, corresponds to abandoning the project. Let $\hat{\lambda} = \sum_{k \in Y} \lambda_k$. The probability that activity $j \in Y$ completes first among the active activities equals $\lambda_j / \hat{\lambda}$ (competing exponentials; see our working paper Creemers, De Reyck, and Leus (2013) for more details). The expected time to the

first completion is $\hat{\lambda}^{-1}$ time units (the length of this timespan is also exponentially distributed) and the appropriate discount factor to be applied for this timespan is $\hat{\lambda}/\left(r+\hat{\lambda}\right)$ (see working paper). In state $(I, Y, P) \in Q$, the expected NPV to be obtained from the next state on condition that no new activities are started equals

$$F_0(I,Y) = \frac{\hat{\lambda}}{r+\hat{\lambda}} \sum_{j\in Y} \frac{p_j\lambda_j}{\hat{\lambda}} F(I \setminus N_i, Y \setminus N_i) + \frac{\hat{\lambda}}{r+\hat{\lambda}} \sum_{j\in Y: N_i\setminus\{j\}\not\subset P} \frac{(1-p_j)\lambda_j}{\hat{\lambda}} F(I, Y\setminus\{j\}), \quad (4)$$

with $j \in N_i$ in the summations. Our side conditions are $F(I, \varnothing) = 0$ for all $I$.

The second alternative is to start a non-empty set of eligible activities $S \subseteq E(I, Y)$ when a state $(I, Y, P) \in Q$ is entered. This leads to incurring a cost $\sum_{j\in S} c_j$ and an immediate transition to another state, with no discounting required. The corresponding eNPV, conditional on set $S \neq \varnothing$ being started, is

$$F_S(I,Y) = F(I \setminus S, Y \cup S) + \sum_{j\in S} c_j. \quad (5)$$

The total number of decisions $S$ that can be made is $2^{|E(I,Y)|}$. The decision corresponding to the highest value in (4) and (5) determines $F(\cdot)$:

$$F(I,Y) = \max\left\{F_0(I,Y)\,;\, \max_{S\neq\varnothing}\{F_S(I,Y)\}\right\}, \quad (6)$$

for feasible state $(I, Y, N \setminus (I \cup Y))$.

The computation of the backward SDP recursion (6) starts in state $(\varnothing, \{n\}, N \setminus \{n\})$. Subsequently, the value function is evaluated stepwise for all other states. The optimal objective value $\max_{\Pi\in\mathcal{P}} \mathbb{E}[f(\Pi)]$ is obtained as $F(N \setminus \{0\}, \{0\})$. We should note that the policies from which one with the best objective function is chosen, do not consider the option of starting activities at the end of activities that are redundant (past) because another activity already made their module succeed.

### 3.3. Illustration

In this section, we illustrate the functioning of the SDP algorithm by analyzing the example project with seven activities ($n = 6$) introduced in Section 2.3, for which the module order $A$ is described by Fig. 1. Further input data are provided in Table 1; the project's payoff value $C$ is 300 and the discount rate is 10 percent per time unit ($r = 0.1$).

For exponentially distributed activity durations, the SDP recursion described in Section 3.2 can be applied to find an optimal policy. At the onset of the project (in state $(N \setminus \{0\}, \varnothing, \{0\})$) we can decide to start either the first activity, the second activity, or both, from module 1. The SDP recursion evaluates the expected outcome of each of these decisions and selects one that yields the highest expected NPV (assuming that optimal decisions are made at all future decision times). In our example, it is optimal to start only the first activity (corresponding to an objective function of 3.27) and we subsequently end up in state $(\{2, 3, 4, 5\}, \{1\}, \{0\})$, in which two possibilities arise. If activity 1 succeeds, module 1 succeeds as well and a transition occurs to state $(\{4, 5\}, \varnothing, \{0, 1, 2, 3\})$; otherwise (if activity 1 fails), we end up in state $(\{2, 3, 4, 5\}, \varnothing, \{0, 1\})$ and have to make a decision: either we start activity 2, corresponding to a transition to state $(\{3, 4, 5\}, \{2\}, \{0, 1\})$ and an eNPV at that time for the remainder of the project of −1.06, or we abandon the project altogether obtaining a current value of 0. The optimal decision in this case is obviously not to continue the project.

**Table 1**
Project data for the example project.

| Task $i$ | Cash flow $c_i$ | Mean duration $\mathbb{E}[D_i]$ | $p_i$ (percent) |
|---|---|---|---|
| 0 | 0 | 0 | 100 |
| 1 | −20 | 10 | 40 |
| 2 | −35 | 2 | 35 |
| 3 | −70 | 8 | 75 |
| 4 | −10 | 2 | 100 |
| 5 | −10 | 2 | 60 |
| 6 | | 0 | 100 |

After a successful completion of module 1, two new activities become eligible. The optimal decision is to start both activities 4 and 5, leading to state $(\varnothing, \{4, 5\}, \{0, 1, 2, 3\})$. Two possibilities then arise: either activity 4 or activity 5 finishes first. Irrespective of which activity completes first, if either activity 4 or 5 fails then the entire project fails. If activity 4 (resp. 5) finishes first and succeeds, activity 5 (resp. 4) is still in progress and needs to finish successfully for the project payoff to be earned. We refer to this optimal policy for exponential durations by the name $\Pi_1$.

The relevant part of the corresponding decision tree is represented in Fig. 2, in which the project evolves from left to right. A *decision node*, represented by a square, indicates that a decision needs to be made at that point in the process; a *chance node*, denoted by a circle, indicates that a random event takes place. Underneath each decision node, we indicate the eNPV conditional on an optimal decision being made in the node, which applies only to the part of the project that remains to be performed. For each decision node, a double dash // is added to each branch that does not correspond to an optimal choice in the SDP recursion.

### 3.4. Generalization towards PH distributions

We now assume that the durations $D_j$ of the activities $j \in N \setminus \{0, n\}$ are mutually independent PH-distributed stochastic variables. PH distributions were first introduced by Neuts (1981) as a means to

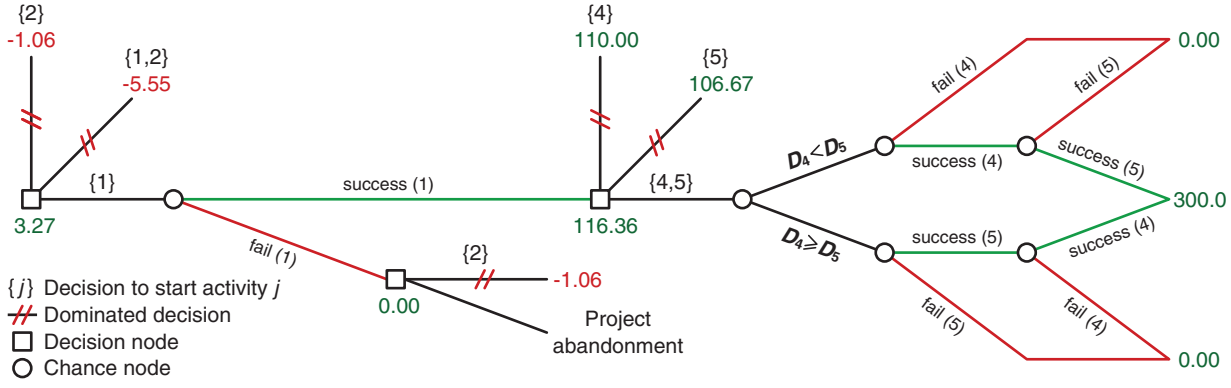



**Fig. 2.** Optimal paths in the decision tree for the example project.

approximate general distributions using a combination of exponentials. We will adopt so-called acyclic PH distributions for the activity durations in order to assess the impact of activity duration variability on the eNPV of a project. In this section, we informally describe PH distributions and show how to determine the optimal eNPV of a project when activity durations are PH distributed. More details, including a moment-matching approach, are described in Creemers et al. (2013).

Due to the properties of the acyclic PH distribution, each activity $j \neq 0, n$ can be seen as a sequence of $z_j$ phases where:

- each phase $\theta_{ju}$ has an exponential duration with rate $\lambda_{ju}$,
- each phase $\theta_{ju}$ has a probability $\tau_{ju}$ to be the initial phase when starting activity $j$,
- each phase $\theta_{ju}$ is visited with a given probability $\pi_{jvu}$ when departing from another phase $\theta_{jv}$.

Acyclicity of the distribution implies that a state is never visited more than once. Since the execution of a task is non-preemptive, the execution of the sequence of phases as well as the execution of a phase itself should be uninterrupted. Therefore, upon completion of a phase $\theta_{ju}$:

- activity $j$ completes with probability $\pi_{ju0}$ (absorption is reached in the underlying Markov chain),
- phase $v$ is started with probability $\pi_{juv}$.

The exponential distribution for activity $j \in N \setminus \{0, n\}$ is then a PH distribution with $z_j = 1$, $\tau_{j1} = 1$ and $\lambda_{j1} \equiv \lambda_j$.

Maintaining the definition of $Y(t)$ given in Section 3.2, define $Y^\circ(t)$ as the set of phases of the activities in $Y(t)$ that are being executed at time instant $t$. Clearly, $Y$ can be obtained from $Y^\circ$. The state of the system is again fully determined by the status of the individual activities and is now represented by a triplet $(I, Y^\circ, P)$. The SDP recursion described in the previous subsection for computing function $F$ is easily extended to accommodate PH distributions; the most important modification is in Eq. (4), which becomes

$$\begin{aligned}
&\frac{\hat{\lambda}^\circ}{r+\hat{\lambda}^\circ} \sum_{\theta_{ju} \in Y^\circ} \pi_{ju0} \frac{p_j \lambda_{ju}}{\hat{\lambda}^\circ} F(I \setminus N_i, Y^\circ \setminus N_i^\circ) \\
&+ \frac{\hat{\lambda}^\circ}{r+\hat{\lambda}^\circ} \sum_{\theta_{ju} \in Y^\circ : N_i \setminus \{j\} \not\subset P} \pi_{ju0} \frac{(1-p_j)\lambda_{ju}}{\hat{\lambda}^\circ} F(I, Y^\circ \setminus \{\theta_{ju}\}) \\
&+ \frac{\hat{\lambda}^\circ}{r+\hat{\lambda}^\circ} \sum_{\theta_{ju} \in Y^\circ} \frac{\lambda_{ju}}{\hat{\lambda}^\circ} \sum_{\substack{v=1 \\ v \neq u}}^{z_j} \pi_{juv} F(I, Y^\circ \cup \{\theta_{jv}\} \setminus \{\theta_{ju}\}),
\end{aligned} \tag{7}$$

with $j \in N_i$, $\hat{\lambda}^\circ = \sum_{\theta_{kv} \in Y^\circ} \lambda_{kv}$ and $N_i^\circ = \{\theta_{ku} : k \in N_i\}$. We use the result that the probability that phase $\theta_{ju} \in Y^\circ$ completes first among the active phases equals $\lambda_{ju}/\hat{\lambda}^\circ$ and that the expected time to the first completion is $\hat{\lambda}^{\circ\,-1}$ time units.

## 4. Computational performance

In this section, we will briefly evaluate the computational performance of the SDP algorithm. Our experiments are performed on an AMD Phenom II with 3.21 gigahertz CPU speed and 2 gigabytes of RAM. To investigate the impact of variability, we use PH distributions to model the activity durations, which will allow us to increase or decrease the variability and examine its impact on the project's eNPV by changing the Squared Coefficient of Variation ($SCV$) of the activity durations (for simplicity, we assume all activity durations to have equal $SCV$). Setting $SCV = 1$ corresponds to exponentially distributed activity durations, $SCV = 0$ coincides with deterministic durations.

We borrow the datasets that were generated by Coolen et al. (2011): these consist of 10 instances for each of various values of the number of activities $n$ and for $OS = 0.4$, 0.6 and 0.8, with 'order strength' $OS$ the number of comparable activity pairs according to the induced order $B^*$, divided by the maximum possible number of such pairs (this value is only approximate). Average activity durations are not used by Coolen et al. (2011) and are additionally generated for each activity, for each instance separately; each such average duration is a uniform integer random variate between 1 and 15. In the generated instances, all activities apart from the final one have negative cash flows and the final activity has a positive payoff (which is also significantly larger in absolute value); we refer to the appendix of Coolen et al. (2011) for more details.

**Table 2**
Number of successfully analyzed networks out of 10.

| $n$ | $OS = 0.8$ | $OS = 0.6$ | $OS = 0.4$ |
|---|---|---|---|
| 11 | 10 | 10 | 10 |
| 21 | 10 | 10 | 10 |
| 31 | 10 | 10 | 10 |
| 41 | 10 | 10 | 7 |
| 51 | 10 | 10 | 5 |
| 61 | 10 | 6 | 3 |
| 71 | 9 | 5 | 3 |
| 81 | 10 | 4 | 1 |
| 91 | 9 | 4 | 0 |
| 101 | 10 | 1 | 0 |
| 111 | 9 | 1 | 0 |
| 121 | 8 | 0 | 0 |

**Table 3**
Average size of the state space ($|Q|$) for analyzed networks.

| $n$ | $OS = 0.8$ | $OS = 0.6$ | $OS = 0.4$ |
|---|---|---|---|
| 11 | 74 | 248 | 628 |
| 21 | 396 | 4,303 | 29,793 |
| 31 | 2,174 | 192,984 | 911,558 |
| 41 | 15,871 | 1,619,351 | 25,051,988 |
| 51 | 98,559 | 1,940,598 | 90,057,422 |
| 61 | 177,916 | 29,540,126 | 278,145,443 |
| 71 | 2,260,271 | 85,611,285 | 82,971,948 |
| 81 | 2,070,967 | 34,261,271 | 176,976,352 |
| 91 | 23,128,416 | 145,911,293 | |
| 101 | 24,804,064 | 165,306,852 | |
| 111 | 67,477,195 | 56,193,712 | |
| 121 | 69,245,416 | | |

For exponential durations, an upper bound on $|Q|$ is $3^n$. Enumerating all these $3^n$ states is not recommended, as the majority of the states typically do not satisfy the precedence constraints. For PH durations, the bound becomes $\prod_{j \in N} 3^{z_j}$. In order to minimize storage and computational requirements, we adopt the techniques proposed by Creemers et al. (2010): as the algorithm progresses, the information on the earlier generated states will no longer be required for further computation and therefore the memory occupied can be freed. This procedure is based on a partition of $Q$, allowing for the necessary subsets to be generated and deleted when appropriate.

In our implementation, the storage requirement for 600,000 states amounts to a maximum of 4.58 megabytes; we only generate feasible states. On our computer, a maximum state space of 268,435,456 states can be stored entirely in memory. Our results with exponential durations are presented in Tables 2–4, gathered per combination of values for $OS$ and $n$ (all runtimes are reported in seconds). The discount rate $r$ equals 10 percent. The tables show that networks of up to 40 activities are analyzed with relative ease. When $n = 51$, however, the optimal solution of most networks with low order strength ($OS = 0.4$) is beyond reach when the system memory is restricted to 2 gigabytes. When $OS = 0.6$, the performance is limited to networks with $n = 71$ or less. We observe that the density of the induced order $B^*$ is a major determinant for the computational effort: order strengths and computation times clearly display an inverse

**Table 4**
Average CPU time (in seconds) required to find an optimal policy.

| $n$ | $OS = 0.8$ | $OS = 0.6$ | $OS = 0.4$ |
|---|---|---|---|
| 11 | 0 | 0 | 0 |
| 21 | 0 | 0 | 0.03 |
| 31 | 0 | 0.3 | 1.77 |
| 41 | 0.02 | 3.54 | 70.93 |
| 51 | 0.15 | 5.12 | 298.41 |
| 61 | 0.32 | 128.31 | 2, 397.93 |
| 71 | 17.53 | 469.34 | 27, 065.53 |
| 81 | 5.7 | 1817.54 | 15, 605.91 |
| 91 | 107.61 | 1, 322.77 | |
| 101 | 105.66 | 894.61 | |
| 111 | 283.57 | 10, 540.86 | |
| 121 | 528.81 | | |

relation. Additionally, the real bottleneck for the algorithm turns out to be memory space rather than CPU time: projects with $n = 81$ and $OS = 0.4$ require less than 5 hours runtime on average (the highest runtime over all the tested settings), which is still practical for industrial-type projects, but larger instances with $OS = 0.4$ cannot be analyzed anymore due to memory limits. From Tables 3 and 4, it may appear that sometimes the instances become easier as the number of jobs increases. This, however, is merely a result of the fact that for larger $n$ not all instances can be solved and therefore the reported averages are essentially truncated, with the largest values not being included.

As a side note, we observe that given the number of states generated, approximation techniques might be useful, either by restricting to 'classic' scheduling heuristics such as list policies, or by resorting to more mainstream approximation techniques for Markovian decision processes (see for instance Powell, 2009; Puterman, 1994). This option is not further pursued in this article.

## 5. Impact of activity duration variability

In this section, we examine the impact of different degrees of variability of the activity durations on a project's value. We do this for the example project instance in Section 5.1, and we generalize by testing with a larger-scale experimental setup in Section 5.2.

### 5.1. Impact of duration variability in the example instance

The policy $\Pi_1$ described in Section 3.3 is optimal for exponential durations; its objective value is 3.27 for the example. The quality of the policy changes when the variability level is different, however. Fig. 3(a) illustrates the functioning of policy $\Pi_1$ with deterministic durations: the policy first executes only activity 1, and then starts both activity 4 and 5 if 1 succeeds, otherwise the project is abandoned.

| | deterministic | exponential |
|---|---|---|
| $\Pi_1$ | −1.26 | 3.27 |
| $\Pi_2$ | 1.50 | −1.06 |

*variability increases* →

**Fig. 4.** eNPV for policies $\Pi_1$ and $\Pi_2$.

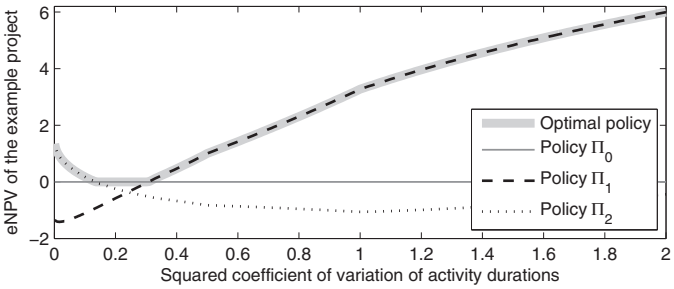



**Fig. 5.** The effect of activity duration variability on the optimal eNPV for the example project.

The objective function value for $\Pi_1$ with deterministic durations is

$$\mathbb{E}[f(\Pi_1)] = c_1 + p_1 e^{-r\mathbb{E}[D_1]}\left(c_4 + c_5 + p_4 p_5 C e^{-r\mathbb{E}[D_5]}\right) = -1.26.$$

An optimal policy $\Pi_2$ for this setting is described by Fig. 3(b), with eNPV

$$\mathbb{E}[f(\Pi_2)] = c_2 + p_2 e^{-r\mathbb{E}[D_2]}\left(c_4 + c_5 + p_4 p_5 C e^{-r\mathbb{E}[D_5]}\right) = 1.50.$$

Here, activity 2 is started at the project's initiation, and activity 1 is never selected (i.e., upon failure of activity 2 the project is abandoned). With exponential durations, on the other hand, $\Pi_2$ has an objective value of −1.06, significantly lower than the optimal value of 3.27 achieved by $\Pi_1$. Interestingly, the inferior policy in the case of exponential durations becomes optimal when activity durations are deterministic. Also, the effect of variability on the eNPV associated with a policy is not monotonic; the eNPV of policy 1 increases, whereas the eNPV of policy 2 decreases. Of particular interest is the fact that the eNPV can actually increase when variability is introduced, which is quite counterintuitive. Note also that for each of the two variability settings, the sign of the objective of two policies is different (one policy achieves a negative NPV while the other one has positive NPV); we summarize these values in Fig. 4. This is a strong case for incorporating all variability information into the computations and not only 'plugging in' the expectations into a deterministic model, since a good project might be cut from the portfolio based only on expected values, whereas it would be able to add value with a carefully selected scheduling strategy.

Define policy $\Pi_0$ as the immediate abandonment of the project, with zero objective value. Fig. 5 depicts the eNPV of the *optimal* policy for each level of duration variability; for any value of *SCV*, either

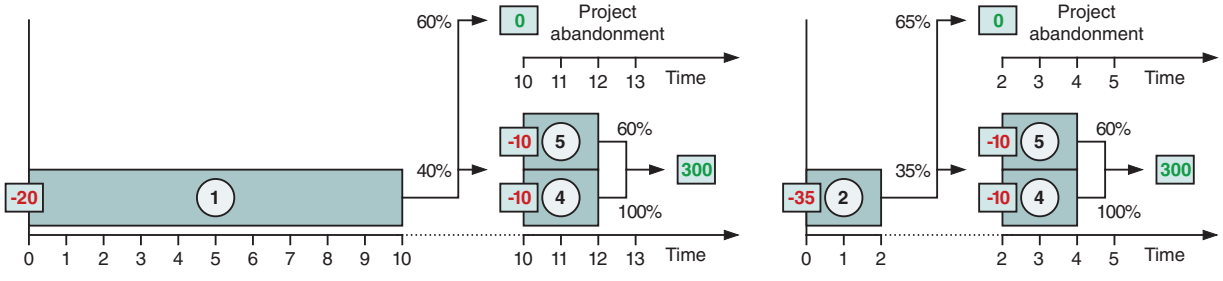
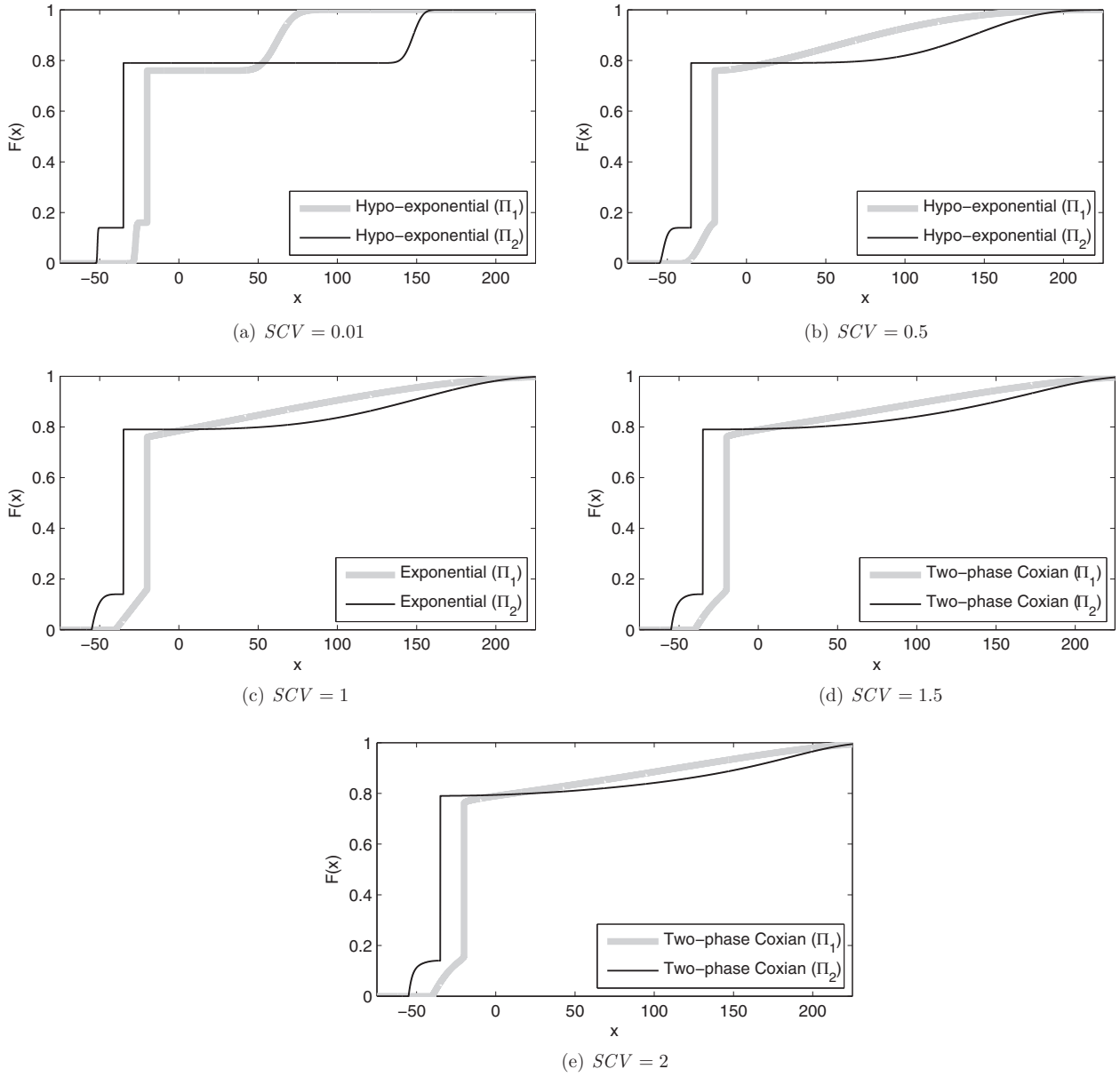



**Fig. 3.** Policies with deterministic durations.




(a) $SCV = 0.01$ (b) $SCV = 0.5$

(c) $SCV = 1$ (d) $SCV = 1.5$

(e) $SCV = 2$

**Fig. 6.** cdf of NPV associated with policies $\Pi_1$ and $\Pi_2$ for various activity duration distributions and various levels of variability.

$\Pi_0$, $\Pi_1$ or $\Pi_2$ is optimal. In particular, for a specific range of *SCV* values, policy $\Pi_0$ (*not* executing the project) is preferable, while different optimal policies appear for other ranges. We observe that eNPV decreases with *SCV* for policy $\Pi_2$. Policy $\Pi_1$, on the other hand, exhibits a U-shaped relationship between *SCV* and project eNPV. In this particular instance, the eNPV of the project is largest when activity durations are highly uncertain (exponentially distributed). This contrasts with the intuition that an increase in uncertainty necessarily entails a decrease of system performance. These findings are further explored in Section 5.2 by means of experiments on a larger set of instances.

Even with exponential durations, it is not a trivial matter to analytically evaluate the entire distribution of a project's NPV; in fact, we are not aware of any studies that have attempted to achieve this directly. More work is available on the analytical evaluation of project makespan in the context of the PERT problem. It turns out that, with discrete independent durations, computing the expected makespan, and computing a single point of the distribution function, are both #P-complete (any #P-complete problem is polynomially equivalent to counting the number of Hamiltonian cycles of a graph and thus in particular NP-complete) (Hagstrom, 1988; Möhring, 2001). Since project NPV is a function of project makespan, this is at least a clear indication that evaluating NPV analytically is probably highly intractable for general duration distributions, and we therefore resort to simulation as a means to approximate the NPV distribution.

For policies $\Pi_1$ and $\Pi_2$ for the example instance, Fig. 6 shows the NPV distribution (cdf) for a number of different values for *SCV*; these plots were obtained via simulation. From module 1, policy $\Pi_1$ only executes activity 1 while $\Pi_2$ only executes activity 2, which is longer but less expensive, and has a slightly higher success probability. We observe that $\Pi_2$ has both a higher upside potential (higher probability of achieving high NPV) as well as a higher downside risk (larger chance of low NPV realizations); the net effect of this comparison is favorable towards policy $\Pi_1$ when *SCV* goes beyond the value of 0.2 (approximately). Apparently, the higher success probability and lower cost of activity 1 become more attractive (compared to activity 2) when the duration variability is higher, such that low duration realizations for $D_1$ can also be achieved, while higher-than-average realizations of $D_1$

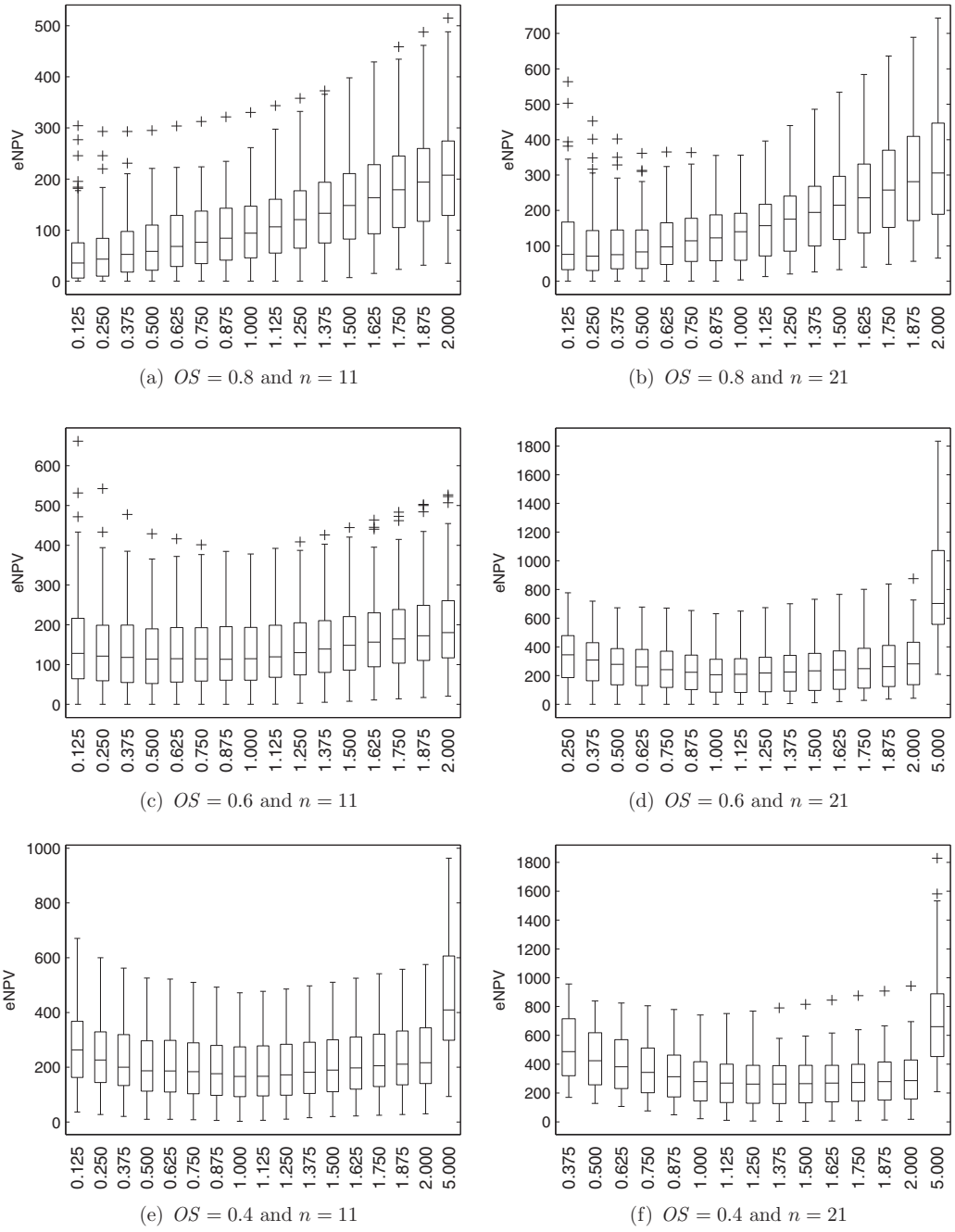


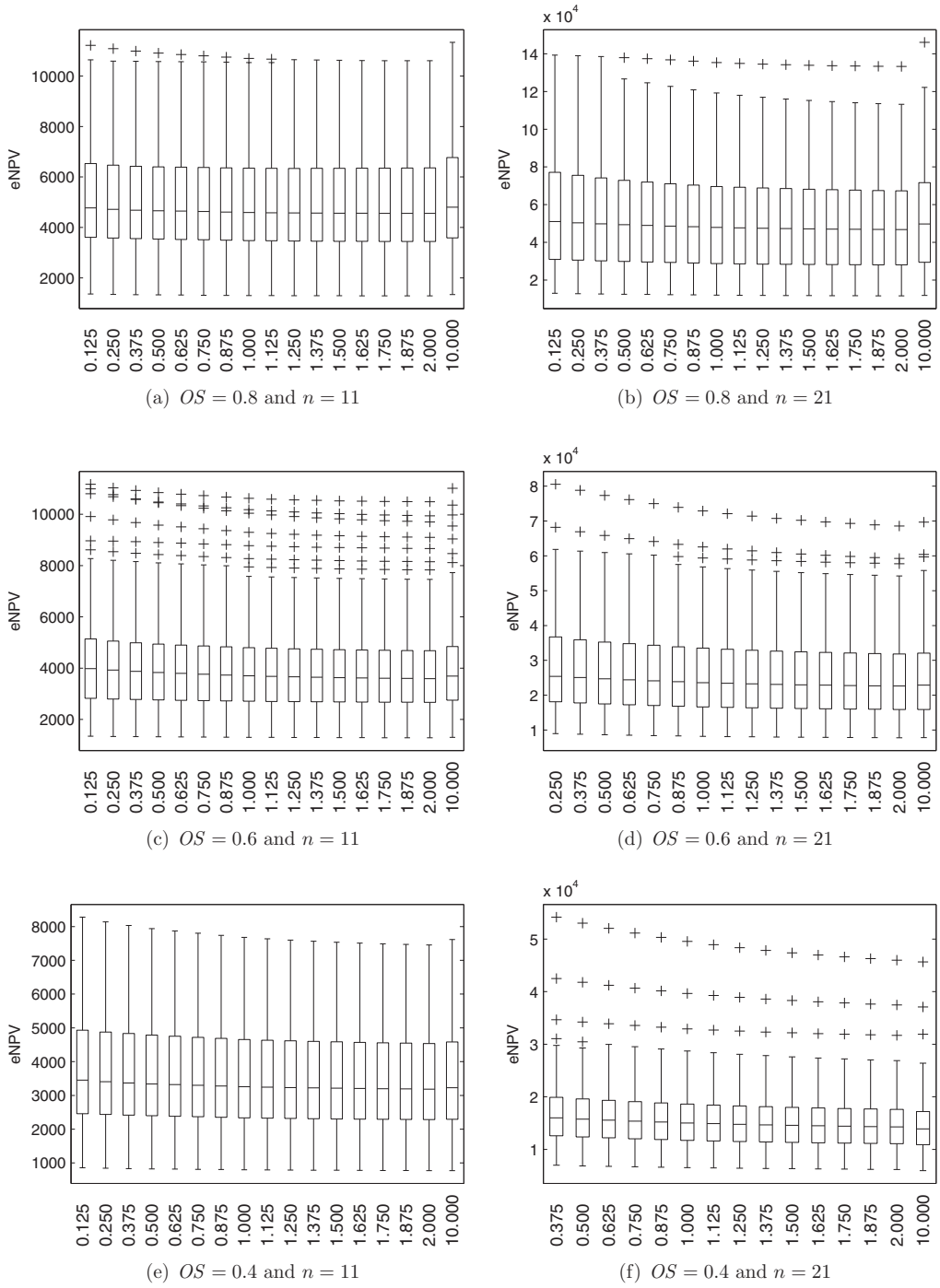

**Fig. 7.** Boxplots of eNPV for different values of *SCV*, *n* and *OS* with $r = 0.1$.

will probably not affect the eNPV with the same magnitude because of the concave and non-increasing dependence of the discount factor with time. In other words, this example indicates that the interplay between activity costs, success probabilities, average durations and the discount factor induces the different dependence of $\Pi_1$ and $\Pi_2$ on *SCV*.

### 5.2. Impact of variability: experiments

Ward and Chapman (2003) argue that all current project risk-management processes induce a restricted focus on the management of project uncertainty. In part, this is because the term 'risk' encourages a 'threat' perspective: we refer the reader to the examples of risk events in the model for variability reduction by Ben-David and Raz (2001) and Gerchak (2000). Ward and Chapman state that a focus on 'uncertainty' rather than risk could enhance project risk management, providing an important difference in perspective, including, but not limited to, an enhanced focus on opportunity management, an 'opportunity' being a 'potential welcome effect on project performance.' Ward and Chapman suggest that management strive for a shift from a threat focus towards greater concern with understanding and managing all sources of uncertainty, with both up-side and down-side consequences, and explore and understand the origins of uncertainty before seeking to manage it. They suggest using the term 'uncertainty management,' encompassing both 'risk management' and 'opportunity management.' See also Loch, DeMeyer, and Pich (2006) for examples of how downside risks can sometimes be turned into upside opportunity (e.g., p. 5 and p. 20).

In order to examine the impact of duration variability on the value of a project in a more structured fashion, we have generated new




(a) $OS = 0.8$ and $n = 11$ (b) $OS = 0.8$ and $n = 21$

(c) $OS = 0.6$ and $n = 11$ (d) $OS = 0.6$ and $n = 21$

(e) $OS = 0.4$ and $n = 11$ (f) $OS = 0.4$ and $n = 21$

**Fig. 8.** Boxplots of eNPV for different values of *SCV*, *n* and *OS* with $r = 0.01$.

instances in line with Coolen et al. (2011), with $n \in \{11, 21\}$ and $OS \in \{0.4, 0.6, 0.8\}$ but now we generate 100 instances per combination of parameter settings, and there is no activity failure nor modular completion of the project (each activity constitutes a separate module). The payoff value $C$ is (uniform) randomly chosen from interval $[0.9C_0; 2C_0]$, where $C_0$ is the payoff value that associates objective value 0 (break-even) with the early-start policy $\Pi^{ES}$ for $SCV = 1$. We consider a wide range of *SCV* values; for more details on the generation of the duration distributions, see Creemers et al. (2013). The results are graphically summarized in Fig. 7 for $r = 10$ percent and in Fig. 8 for $r = 1$ percent. We investigate the effect of different variability levels (different values of *SCV*) on the value of the project. We observe that variability reduction is systematically *not* beneficial for the project's value as measured by eNPV in the cases where the precedence network is rather dense and the discount rate is high; this corresponds with Fig. 7(a)–(c).

These results may be explained by: (1) the likelihood of serial execution, and (2) the concaveness of the discount function $e^{-rt}$. With high *OS*, the precedence network is close to serial, and an increase in duration variability results in an increase in the probability of completing the activity after a short amount of time. Due to the concave shape of the discount function, the gain in the objective associated with low duration realizations can offset the loss associated with higher duration realizations, and this is more pronounced for higher *r*.

**Table 5**
Comparison of policy classes: average difference in eNPV as a proportion of the payoff.

| | | *SCV* | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 0.125 | 0.25 | 0.375 | 0.5 | 0.625 | 0.75 | 0.875 | 1 |
| $n = 11$ | $OS = 0.8$ | 0.001 | 0.001 | 0.001 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| | $OS = 0.6$ | 0.001 | 0.001 | 0.001 | 0.001 | 0.001 | 0.001 | 0.000 | 0.000 |
| | $OS = 0.4$ | 0.003 | 0.002 | 0.002 | 0.001 | 0.002 | 0.001 | 0.001 | 0.000 |
| $n = 21$ | $OS = 0.8$ | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| | $OS = 0.6$ | – | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| | $OS = 0.4$ | – | – | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |

Low *OS*, by contrast, will imply that activities are more often executed in parallel, and then the start of new activities is more frequently defined by the maximum of multiple activity durations, the so-called *merge (bias) effect* (Klingel, 1966). This merge effect is less likely to give rise to short completion times even in the event that some activity durations are low, and thus reduces the benefits associated with the concave discount function. Optimal scheduling policies will indeed tend to execute some of the activities in parallel rather than serially when possible (low *OS*), because this reduces the project makespan and thus leads to earlier project payoff.

Thus, investing in variability reduction becomes more interesting if: (1) $r$ is low, (2) *OS* is low, and (3) variability can almost be eliminated. With a higher number $n$ of activities, ceteris paribus, the project duration will typically also be higher and there will be more chances for merge bias, so we would expect variability reduction to be more beneficial; this is also confirmed by the experimental results. The figures also show that *very* high variability often exhibits increased eNPV, but this phenomenon only occurs for unrealistically high *SCV* values ($SCV = 10$) in some of the settings. Similar patterns arise when activity failures are included and when there may be more than one activity in the same module (which is not the case in the datasets to which the plots pertain). The effects are also not dependent on the PH-type character of the distributions: we have found comparable behavior in simulations with lognormal and gamma distributions. As a final remark, we underline that all the observations made in this section pertain exclusively to *expected* NPV; obviously, lower duration variability is likely to induce lower variability in the NPV realizations as well, which may or may not be of significant importance to management, depending for instance on whether an entire portfolio of projects or rather only one individual project is being managed.

## 6. Policy class: experiments

Following up on the discussion in Section 3.1, we further examine the possible choices for the policy class. Table 5 contains the results for an experiment with which we evaluate whether the consideration of policies that start activities only at the end of other activities, is very restrictive. The experiments were run on the datasets with $n = 11$ and 21 that were used in Section 5.2. We consider $SCV \in \{0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875, 1\}$. For $n = 21$ and $OS = 0.4$, we do not report results for networks with $SCV \in \{0.125, 0.25\}$, and we also do not cover the combination $n = 21$, $OS = 0.6$ and $SCV = 0.125$. The reason for excluding some combinations is that lower *SCV* requires more phases to model the activity durations: $SCV = 0.25$, for instance, requires four phases for each activity, which results in a network of $4n$ phases. With $r = 0.1$ and for each value of *SCV* and *OS*, Table 5 reports the decrease in the objective value by optimizing over the restricted policy class as compared to the more general class that also considers starting new activities after the completion of each phase of each ongoing activity; the decrease is expressed as a proportion of the payoff $C$ and averaged over the 100 instances.

We conclude that the benefits of allowing activity start also at other times than only at the completion of other activities are minor, and nowhere higher than around 0.3 percent of the payoff. The benefits are higher especially when variability is low; this is logical, since there are more phases and hence more decision times with lower *SCV*. The observation is also in line with the fact that for deterministic durations, late-start scheduling is optimal (see Section 3.1). When $SCV = 1$, the two classes coincide. At the same time, there were no significant differences in the computational effort for finding an optimal member in the larger policy class. In other words, from a computational viewpoint, there is no real downside to allowing decisions to be made during the execution of activities, but the benefits are also quite limited. Other values of $r$ have also been tested, with similar findings.

## 7. Summary and outlook on research perspectives

Project planning with traditional tools typically ignores technological and duration uncertainty. In this article, we have explained how to model scheduling decisions in a practical environment with considerable uncertainty, and we have illustrated how decision making based only on expected values can lead to inappropriate decisions. We have developed a generic model for optimally scheduling R&D projects with stochastic activity durations, possible activity failures and modular project completion. We have assessed the effect of different degrees of activity duration variability on the expected NPV of a project. Finally, we have illustrated that higher operational variability does not always lead to lower project values, meaning that (sometimes costly) variance reduction strategies are not always advisable. This contradicts the intuition that an increase in uncertainty necessarily entails a decrease of system performance.

For future research, there are a number of topics that have been brought up in this article and that deserve further exploration. In particular, an analytical study of the different determinants of the effect of varying duration variability on the expected NPV would be highly interesting; in this article, this analysis was mainly computational. This pertains to project characteristics such as network density, which influences the importance of phenomena such as the merge bias effect, but it can also include the impact of the discount factor. Additionally, higher moments of the duration distributions, such as skewness and kurtosis, might also play a role. As a final interesting research avenue, we mention the study of the variability of a project's NPV rather than only the expected value.

## References

Abernathy, W. J., & Rosenbloom, R. S. (1969). Parallel strategies in development projects. *Management Science*, *15*(10), 486–505.
Adlakha, V. G., & Kulkarni, V. G. (1989). A classified bibliography of research on stochastic PERT networks. *INFOR*, *27*(3), 272–296.
Baldwin, C. Y., & Clark, K. B. (2000). *Design rules: The power of modularity*. Cambridge, MA, USA: The MIT Press.
Ben-David, I., & Raz, T. (2001). An integrated approach for risk response development in project planning. *Journal of the Operational Research Society*, *52*(1), 14–25.
Benati, S. (2006). An optimization model for stochastic project networks with cash flows. *Computational Management Science*, *3*(4), 271–284.
Buss, A. H., & Rosenblatt, M. J. (1997). Activity delay in stochastic project networks. *Operations Research*, *45*(1), 126–139.

Chopra, S., & Meindl, P. (2013). *Supply chain management: Strategy, planning, and operation*, New Jersey, USA: Prentice Hall.
Chun, Y. H. (1994). Sequential decisions under uncertainty in the R&D project selection problem. *IEEE Transactions on Engineering Management*, *41*(4), 404–413.
Coolen, K., Wei, W., Talla Nobibon, F., & Leus, R. (2014). Scheduling modular projects on a bottleneck resource. *Journal of Scheduling*, *17*(1), 67–85.
Creemers, S., De Reyck, B., & Leus, R. (2013). *Project planning with alternative technologies in uncertain environments* (Working paper #1314). KU Leuven, Faculty of Business and Economics, Department of Decision Sciences and Information Management.
Creemers, S., Leus, R., & Lambrecht, M. (2010). Scheduling Markovian PERT networks to maximize the net present value. *Operations Research Letters*, *38*(1), 51–56.
De Reyck, B., & Leus, R. (2008). R&D-project scheduling when activities may fail. *IIE Transactions*, *40*(4), 367–384.
Ding, M., & Eliashberg, J. (2002). Structuring the new product development pipeline. *Management Science*, *48*(3), 343–363.
Elmaghraby, S. (1977). *Activity networks: Project planning and control by network models*, New York, NY, USA: John Wiley & Sons Inc.
Gerchak, Y. (2000). On the allocation of uncertainty-reduction effort to minimize total variability. *IIE Transactions*, *32*(5), 403–407.
Gittins, J., & Yu, J. Y. (2007). Software for managing the risks and improving the profitability of pharmaceutical research. *International Journal of Technology Intelligence and Planning*, *3*(4), 305–316.
Granot, D., & Zuckerman, D. (1991). Optimal sequencing and resource allocation in research and development projects. *Management Science*, *37*(2), 140–156.
Hagstrom, J. N. (1988). Computational complexity of PERT problems. *Networks*, *18*, 139–147.
Huysmans, M., Coolen, K., Talla Nobibon, F., & Leus, R. (2012). *A fast greedy heuristic for scheduling modular projects* (Working paper #1227). KU Leuven, Faculty of Business and Economics, Department of Decision Sciences and Information Management.
Igelmund, G., & Radermacher, F. J. (1983). Preselective strategies for the optimization of stochastic project networks under resource constraints. *Networks*, *13*(1), 1–28.
Jain, V., & Grossmann, I. E. (1999). Resource-constrained scheduling of tests in new product development. *Industrial & Engineering Chemistry Research*, *38*(8), 3013–3026.
Klingel, A. R. Jr. (1966). Bias in Pert project completion time calculations for a real network. *Management Science*, *13*(4), B194–B201.
Kulkarni, V., & Adlakha, V. (1986). Markov and Markov-regenerative PERT networks. *Operations Research*, *34*(5), 769–781.
Lenfle, L. (2011). The strategy of parallel approaches in projects with unforeseeable uncertainty: The Manhattan case in retrospect. *International Journal of Project Management*, *29*(4), 359–373.
Loch, C. H., DeMeyer, A., & Pich, M. T. (2006). *Managing the unknown: A new approach to managing high uncertainty and risk in projects*, Hoboken, NJ, USA: Wiley.
Ludwig, A., Möhring, R. M., & Stork, F. (2001). A computational study on bounding the makespan distribution in stochastic project networks. *Annals of Operations Research*, *102*, 49–64.
Möhring, R. H. (2000). Scheduling under uncertainty: Optimizing against a randomizing adversary. *Lecture Notes in Computer Science*, *1913*, 15–26.
Möhring, R. H. (2001). Scheduling under uncertainty: Bounding the makespan distribution. *Lecture Notes in Computer Science*, *2122*, 79–97.
Nelson, R. R. (1961). Uncertainty, learning, and the economics of parallel research and development efforts. *The Review of Economics and Statistics*, *43*(4), 351–364.
Neuts, M. F. (1981). *Matrix-geometric solutions in stochastic models*, Baltimore, MD, USA: Johns Hopkins University Press.
Powell, W. B. (2009). What you should know about approximate dynamic programming. *Naval Research Logistics*, *56*, 239–249.
Puterman, M. (1994). *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons Inc.
Schmidt, C. W., & Grossmann, I. E. (1996). Optimization models for the scheduling of testing tasks in new product development. *Industrial & Engineering Chemistry Research*, *35*(10), 3498–3510.
Sobel, M. J., Szmerekovsky, J. G., & Tilson, V. (2009). Scheduling projects with stochastic activity duration to maximize expected net present value. *European Journal of Operational Research*, *198*(1), 697–705.
Sommer, S. C., & Loch, C. H. (2004). Selectionism and learning in projects with complexity and unforeseeable uncertainty. *Management Science*, *50*(10), 1334–1347.
Stork, F. (2001). *Stochastic resource-constrained project scheduling* (PhD thesis). Technische Universität Berlin.
Ward, S., & Chapman, C. (2003). Transforming project risk management into project uncertainty management. *International Journal of Project Management*, *21*, 97–105.
Weitzman, M. L. (1979). Optimal search for the best alternative. *Econometrica*, *47*(3), 641–654.
Yu, J. Y., & Gittins, J. (2008). Models and software for improving the profitability of pharmaceutical research. *European Journal of Operational Research*, *189*(2), 459–475.