Imperial College London
Department of Computing

Optimisation of Temporal Networks under Uncertainty

Wolfram Wiesemann

Submitted in part fulfilment of the requirements for the degree of
Doctor of Philosophy in Computing of Imperial College and
the Diploma of Imperial College, June 2010

Abstract

A wide variety of decision problems in operations research are defined on temporal networks, that is, workflows of time-consuming tasks whose processing order is constrained by precedence relations. For example, temporal networks are used to formalise the management of projects, the execution of computer applications, the design of digital circuits and the scheduling of production processes. Optimisation problems arise in temporal networks when a decision maker wishes to determine a temporal arrangement of the tasks and/or a resource assignment that optimises some network characteristic such as the network's makespan (*i.e.*, the time required to complete all tasks) or its net present value.

Optimisation problems in temporal networks have been investigated intensively for more than fifty years. To date, the majority of contributions focus on deterministic formulations where all problem parameters are known. This is surprising since parameters such as the task durations, the network structure, the availability of resources and the cash flows are typically unknown at the time the decision problem arises. The tacit understanding in the literature is that the decision maker replaces these uncertain parameters with their most likely or expected values to obtain a deterministic optimisation problem. It is well-documented in theory and practise that this approach can lead to severely suboptimal decisions.

The objective of this thesis is to investigate solution techniques for optimisation problems in temporal networks that explicitly account for parameter uncertainty. Apart from theoretical and computational challenges, a key difficulty is that the decision maker may not be aware of the precise nature of the uncertainty. We therefore study several formulations, each of which requires different information about the probability distribution of the uncertain problem parameters. We discuss models that maximise the network's net present value and problems that minimise the network's makespan. Throughout the thesis, emphasis is placed on tractable techniques that scale to industrial-size problems.

Acknowledgements

I owe my deepest gratitude to my thesis advisors, Professor Berç Rustem and Dr. Daniel Kuhn from the Department of Computing at Imperial College London. I have benefited greatly from their invaluable suggestions, enlightening advise and constant encouragement, both personally and scientifically. I would also like to express my sincere gratitude to Professor Wolfgang Domschke and Professor Robert Klein who whet my interest in optimisation and decision-making under uncertainty while I was a student at Darmstadt University of Technology.

My very keen appreciation goes to my fellow PhD students and friends at Imperial College London: Nikos Baltas, Dimitra Bampou, Kian Chan, Themis Charalambous, Raquel Fonseca, Michael Hadjiyiannis, Adil Hussain, Will Jones, Iakovos Kakouris, Eva Kalyvianaki, Michalis Kapsos, Xenia Kleniati, Evelina Klerides, Ryan Duy Luong, Paul-Amaury Matt, Nikos Papadakos, George Tzallas-Regas, Angelos Tsoukalas, Phebe Vayanos, Kai Ye and all the others that I should have mentioned. My heartiest thanks go to Angelos Georghiou for countless inspiring discussions at Imperial College and the pubs close by. One day I will proudly tell my children that I shared a room with Steve Zymler, who will undoubtedly shoot to fame for his contributions to the financial markets and the local transportation industry. Also, my academic and social life at Imperial College London greatly benefited from my regular interactions with Dr. Ronald Hochreiter, Dr. Christian Lang and Dr. Peter Pietzuch.

I am deeply indebted to Sharon for all her patience and love, and for reminding me that life holds more than proofs and formulae. I look forward to our next six years and all the other years that will follow. Finally, I cannot overstate my gratitude and appreciation to my family for their nurture, upbringing and unconditional love. Their encouragement and support made it all possible.

Meinen Eltern

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

We define a *temporal network* as a directed, acyclic graph $G = (V, E)$ whose nodes $V = \{1, \ldots, n\}$ represent the network tasks and whose arcs $E \subseteq V \times V$ describe the temporal precedences between the tasks. This convention is known as *activity-on-node* notation; an alternative *activity-on-arc* notation is discussed in [DH02]. In our notation, an arc $(i, j) \in E$ signalises that task j must not be started before task i has been completed. For ease of exposition, we assume that $1 \in V$ represents the unique source and $n \in V$ the unique sink of the network. This can always be achieved by introducing dummy nodes and/or arcs. We assume that the processing of each task requires a non-negative amount of time. Depending on the problem under consideration, the tasks may also give rise to cash flows. Positive cash flows denote cash inflows (*e.g.*, received payments), whereas negative cash flows represent cash outflows (*e.g.*, accrued costs). Figure 1.1 illustrates a temporal network with cash flows.

Optimisation problems arise in temporal networks when the decision maker is able to influence the processing of the network tasks. Most frequently, it is assumed that this is possible in one or two complementary ways. On one hand, the decision maker may be able to decide on the temporal orchestration of the tasks, that is, on the times at which the tasks are processed. On the other hand, the decision maker may be able to change the task durations through the

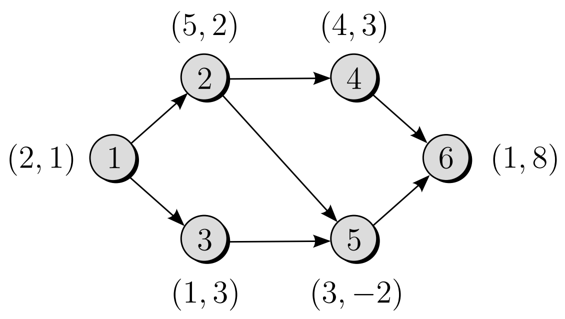


Figure 1.1: Example temporal network. Attached to each node is the duration (first value) and the cash flow (second value) associated with the task.

assignment of resources. A rational decision maker influences the processing of the network tasks in order to optimise an objective function. In this thesis we focus on two prominent objectives, namely the minimisation of the network's makespan (*i.e.*, the time required to process all tasks) and the maximisation of the network's net present value. Other objectives (*e.g.*, cost minimisation or a level resource consumption) are discussed in [DH02].

Temporal networks and their associated optimisation problems are ubiquitous in operations research. In the following, we provide some illustrative examples.

1. **Project Scheduling.** Much of the research on temporal networks originates from the area of project scheduling, see [BDM$^+$99, DH02, NSZ03, Sch05]. In project scheduling, the network tasks represent the various activities in a project (*e.g.*, 'conduct market research' or 'develop prototype'), and the precedence relations describe temporal constraints between the activities (*e.g.*, 'the prototype cannot be developed before the market research has been completed'). The minimisation of a project's makespan and the maximisation of a project's net present value are among the most wide-spread objective functions in project scheduling. We will consider a project scheduling problem in Chapter 4.

2. **Execution of Computer Applications.** Computer applications can be described through flowgraphs whose nodes represent the application commands and whose arcs describe the execution flow. Although a flowgraph typically accommodates sophisticated flow constructs such as loops and conditional branches, it can be converted into a set of alternative execution flows, each of which constitutes a temporal network [vdAtHKB03, WHK08]. The execution of computer applications poses several challenging problems

such as the scheduling of multiple applications on one or more processors and the assignment of resources (*e.g.*, processor time, memory space and I/O access) to application commands, see [BEP$^+$96]. The minimisation of an application's runtime can be cast as a makespan minimisation problem [WHK08].

3. **Design of Digital Circuits.** Modern VLSI (Very-Large-Scale Integration) circuits can contain millions of interconnected logical gates. A key problem in VLSI design relates to the selection of suitable gate sizes [BKPH05]. The gate sizes crucially affect the three primary design objectives 'operating speed', 'total circuit size' and 'power consumption'. A circuit can be expressed as a temporal network whose tasks represent the gates and whose precedences denote the interconnections between the gates. Since the gate delay (*i.e.*, the 'task duration') is a function of the gate size, the maximisation of the circuit speed, subject to constraints on the power consumption and the overall circuit size, can be cast as a makespan minimisation problem in a temporal network. We will investigate circuit sizing problems in Chapter 5.

4. **Process Scheduling.** A typical problem in process scheduling is to manufacture a set of products through a sequence of processing steps. Each processing step can be executed by a number of machines. At any time, a machine can process at most one product, and a product can be processed by at most one machine. Additionally, the processing times can depend on the assignment of resources (*e.g.*, fuel, catalysts and additional manpower). A common objective is to find a resource allocation and processing sequences that optimise the makespan or net present value of the production plan. Process scheduling problems are reviewed in [Bru07, Pin08].

In the remainder of this section, we highlight some of the difficulties that arise when the problem parameters of a temporal network are uncertain. To this end, let us first assume that all parameters are deterministic and that the resource assignment is fixed. We want to determine a vector of start times for the network tasks that optimises the network's makespan or its net present value. The makespan is minimised by the following model.

$$\min_{y \in Y} \; y_n + d_n, \tag{1.1a}$$

where

$$Y = \left\{ y \in \mathbb{R}^n_+ \; : \; y_j \geq y_i + d_i \;\; \forall \, (i,j) \in E \right\}. \tag{1.1b}$$

In this problem, y_i and d_i denote the start time (a variable) and the duration (a parameter) of the ith task, respectively. The set Y contains the admissible start time vectors for the network tasks, that is, all start time vectors that satisfy the precedence constraints. Since n is the unique sink of the network, $y_n + d_n$ represents the network's makespan. Note that (1.1) constitutes a linear program that can be solved efficiently. Indeed, the minimal makespan can be determined much more efficiently if we exploit the following observation. Every admissible start time schedule $y \in Y$ has to satisfy $y_1 \geq 0$ and

$$y_j \geq \max_{i \in V} \left\{ y_i + d_i \; : \; (i,j) \in E \right\} \quad \text{for all } j \in V \setminus \{1\}.$$

Since the makespan is a non-decreasing function of y, the *early start schedule* $y^* \in Y$ with

$$y^*_j = \begin{cases} 0 & \text{if } j = 1, \\ \max_{i \in V} \left\{ y^*_i + d_i \; : \; (i,j) \in E \right\} & \text{otherwise} \end{cases} \tag{1.2}$$

is optimal. Note that the recursion is well-defined because G is acyclic. Hence, we can determine the minimal makespan through a topological sort. In Figure 1.1, the minimal makespan of 12 time units is attained by the start time vector $y^* = (0, 2, 2, 7, 7, 11)^\top$.

The optimality of the early start schedule distinguishes the makespan from other objective functions in temporal networks. To illustrate this point, consider the following net present value maximisation problem.

$$\max_{y \in Y} \; \sum_{i \in V} \zeta_i \beta^{y_i} \tag{1.3}$$

Here, ζ_i denotes the cash flow arising at the start time y_i of task i, $\beta \in (0,1)$ represents

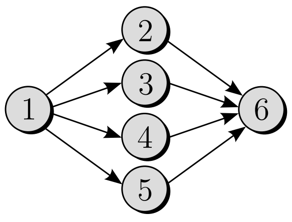


Figure 1.2: Nominal models underestimate the makespan. In the temporal network, tasks 1 and n have duration zero, while the durations of the other tasks follow independent uniform distributions with support $[0, 1]$.

the discount factor, and the set Y of admissible start time schedules is defined in (1.1b). Although the objective function of (1.3) is nonconvex, the problem can be converted into an equivalent linear program by substituting the expressions β^{y_i} with new variables z_i, $i \in V$. We will elaborate on this substitution in Chapter 3. Note that (1.3) is no longer guaranteed to be optimised by the early start schedule y^* if negative cash flows are present. Indeed, for sufficiently large β, the net present value of the network in Figure 1.1 is maximised by the start time vector $y = (0, 2, 2, 7, 8, 11)^\top$. As we will see throughout this thesis, the optimality of the early start schedule can dramatically simplify decision-making in temporal networks when uncertainty is present.

From the previous discussion we conclude that the makespan and the net present value of a deterministic temporal network can be optimised efficiently if the resource assignment is fixed. Let us now assume that the task durations are uncertain. A common suggestion is to solve a *nominal problem* where the uncertain task durations are replaced with their expected values. To see why this approach can be problematic, consider the temporal network $G = (V, E)$ with $V = \{1, \ldots, n\}$ and $E = \{(1, i) : 1 < i < n\} \cup \{(i, n) : 1 < i < n\}$, $n \geq 3$ [Elm05, Mö01]. We illustrate the temporal network for $n = 6$ in Figure 1.2. Assume that the tasks 1 and n have zero duration, while the durations d_i of the tasks $i \in \{2, \ldots, n-1\}$ follow independent uniform distributions with support $[0, 1]$. In this case, the expected duration of tasks 1 and n is zero, while all other tasks have an expected duration of 1/2. From our previous discussion we know that the early start schedule $y^* = (0, \ldots, 0, 1/2)^\top$ minimises the nominal makespan minimisation problem, and hence the obtained estimate for the network's makespan is 1/2. However, the probability that the makespan of the early start schedule does not exceed $t \in [0, 1]$

is given by the expression

$$\mathbb{P}(\max\{d_2, \ldots, d_{n-1}\} \leq t) \;=\; \mathbb{P}(d_2 \leq t, \; \ldots, \; d_{n-1} \leq t) \;=\; \prod_{1<i<n} \mathbb{P}(d_i \leq t) \;=\; t^{n-2}.$$

Thus, the probability to complete all tasks before time $t < 1$ goes to zero as n tends to infinity, and the approximation obtained from solving the nominal problem becomes increasingly weak.

More generally, one can show that the nominal problem always underestimates the expected makespan of the early start schedule. To see this, assume that the task durations d_i, $i \in V$, are random and that each task is started according to the early start policy y^*. Note that y^* constitutes a random vector now because it depends on the random task durations through (1.2). As before, the makespan is $y^*_n + d_n$. The right-hand side of (1.2) is convex and non-decreasing in y^* and d. Hence, we can reformulate the makespan as a convex function of the random task durations d_i by recursively replacing each component of y^* with its definition in (1.2). Jensen's inequality tells us that for a measurable convex function φ and a random vector d, $\varphi(\mathbb{E}(d)) \leq \mathbb{E}(\varphi(d))$. When we solve the nominal problem, we evaluate the left-hand side of this equation (deterministic makespan using expected durations) to approximate the right-hand side (expected makespan using random durations).

These rather pessimistic results on the approximation quality of nominal problems suggest that we should explicitly account for the stochastic nature of temporal networks. Unfortunately, the existence of precedence constraints severely complicates this goal. To see this, consider again the temporal network in Figure 1.1 and assume that d_i, the duration of task $i \in V$, is described by its probability density function f_i and its cumulative distribution function F_i. For ease of exposition, we assume that the task durations are independently distributed. We want to determine the cumulative distribution function of the makespan if each task is started according to the early start schedule y^*. If we denote the cumulative distribution function of y^*_i by G_i, we obtain for the first three tasks

$$G_1(t) = \begin{cases} 1 & \text{if } t \geq 0, \\ 0 & \text{otherwise;} \end{cases} \qquad \text{and} \qquad G_2(t) = G_3(t) = \mathbb{P}(d_1 \leq t) = F_1(t). \tag{1.4}$$

The distribution of y_4 is obtained as follows.

$$G_4(t) = \mathbb{P}(d_1 + d_2 \leq t) = (F_1 * f_2)\,(t), \tag{1.5}$$

where $(\chi_1 * \chi_2)(t) := \int_{\tau\in\mathbb{R}} \chi_1(\tau)\,\chi_2(t-\tau)\,\mathrm{d}\tau$ denotes the convolution of two functions χ_1 and χ_2. The start time of task 5 depends on the maximum of two independent random variables:

$$\begin{aligned} G_5(t) &= \mathbb{P}(\max\{d_1+d_2,\ d_1+d_3\} \leq t) = \mathbb{P}(d_1 + \max\{d_2, d_3\} \leq t) \\ &= (f_1 * \widetilde{G})(t) = (f_1 * [F_2 \cdot F_3])\,(t), \end{aligned}$$

where

$$\begin{aligned} \widetilde{G}(t) &:= \mathbb{P}(\max\{d_2, d_3\} \leq t) = \mathbb{P}(d_2 \leq t,\ d_3 \leq t) \\ &= \mathbb{P}(d_2 \leq t)\,\mathbb{P}(d_3 \leq t) = F_2(t)\,F_3(t). \end{aligned}$$

Here, we used the notation $(\chi_1 \cdot \chi_2)(t) := \chi_1(t)\,\chi_2(t)$. Calculating G_6 is more involved as it depends on the maximum of *dependent* random variables:

$$\begin{aligned} G_6(t) &= \mathbb{P}(\max\{d_1+d_2+d_4,\ d_1+d_2+d_5,\ d_1+d_3+d_5\} \leq t) \\ &= \mathbb{P}(d_1 + \max\{d_2+d_4,\ d_2+d_5,\ d_3+d_5\} \leq t) \\ &= (f_1 * \widehat{G})(t), \end{aligned}$$

where

$$\begin{aligned} \widehat{G}(t) &:= \mathbb{P}(\max\{d_2+d_4,\ d_2+d_5,\ d_3+d_5\} \leq t) \\ &= \int_{\delta_2,\delta_5\geq 0} \mathbb{P}(\delta_2+d_4 \leq t)\,\mathbb{P}(\delta_2+\delta_5 \leq t)\,\mathbb{P}(d_3+\delta_5 \leq t) f_2(\delta_2)\,f_5(\delta_5)\,\mathrm{d}\delta_2\,\mathrm{d}\delta_5 \\ &= \int_{\substack{\delta_2,\delta_5\geq 0,\\ \delta_2+\delta_5\leq t}} \mathbb{P}(d_4 \leq t-\delta_2)\,\mathbb{P}(d_3 \leq t-\delta_5)\,f_2(\delta_2)\,f_5(\delta_5)\,\mathrm{d}\delta_2\,\mathrm{d}\delta_5 \\ &= \int_{\substack{\delta_2,\delta_5\geq 0,\\ \delta_2+\delta_5\leq t}} F_4(t-\delta_2)\,F_3(t-\delta_5)\,f_2(\delta_2)\,f_5(\delta_5)\,\mathrm{d}\delta_2\,\mathrm{d}\delta_5. \end{aligned}$$

The cumulative distribution function of the network's makespan is given by $G_6 * f_6$. Clearly, this approach becomes impractical for large networks. In fact, we cannot expect that there is an algorithm that determines the cumulative distribution function of the makespan efficiently. It has been shown in [Hag88] that even if the task durations are independent random variables with a two-valued support, the calculation of the expected value or any pre-specified quantile of the makespan of y^* is #PSPACE-hard. The situation is complicated by the practical difficulty to estimate the distributions of all task durations.

1.2 Contributions and Structure of the Thesis

We develop solution techniques for optimisation problems in temporal networks under uncertainty. The problems that we consider vary in the required information about the uncertain problem parameters, the employed risk measure (expected value, quantiles and the worst case) and the objective function (makespan and net present value). We apply our techniques to problems in project scheduling and VLSI design. However, we stress that the proposed techniques apply to other application areas of temporal networks as well.

Apart from a review of the background theory in Chapter 2 and conclusions in Chapter 7, the thesis is divided into four chapters. Each of these chapters investigates one specific class of optimisation problems in temporal networks, which can be summarised as follows.

In Chapter 3 we maximise a network's expected net present value when the task durations and cash flows are described by a discrete set of alternative scenarios with associated occurrence probabilities. In this setting, the choice of scenario-independent task start times frequently leads to infeasible schedules or severe losses in revenues. We determine an optimal target processing time policy for the network tasks instead. Such a policy prescribes a task to be started as early as possible in the realised scenario, but never before its (scenario-independent) target processing time. We formulate the resulting model as a global optimisation problem and present a branch-and-bound algorithm for its solution. The contents of this chapter are published in

1. W. Wiesemann, D. Kuhn and B. Rustem. *Maximizing the Net Present Value of a Project under Uncertainty.* European Journal of Operational Research 202(2):356–367, 2010.

Chapter 4 investigates a resource allocation model that minimises the makespan of a temporal network. The model accommodates multiple resources and decision-dependent task durations inspired by microeconomic theory. First, we elaborate a deterministic problem formulation. In a second stage, we enhance the model to account for uncertain problem parameters. Assuming that the first and second moments of these parameters are known, the stochastic model minimises an approximation of the value-at-risk of the network's makespan. As a salient feature, our approach employs a scenario-free formulation which approximates the durations of the network's task paths via normal distributions. We extend our model to situations in which the moments of the random parameters are ambiguous and describe an iterative solution procedure. The contents of this chapter can be found in

2. W. Wiesemann, D. Kuhn and B. Rustem. *Multi-Resource Allocation in Stochastic Project Scheduling.* Accepted for Publication in Annals of Operations Research, 2009.

In Chapter 5 we study a robust resource allocation problem in temporal networks where the task durations are uncertain, and the goal is to minimise the worst-case makespan. We show that this problem is generically $\mathcal{NP}$-hard. We then develop families of optimisation problems that provide convergent lower and upper bounds on the optimal value of the problem. The upper bounds correspond to feasible allocations whose objective values are bracketed by the bounds. Hence, we obtain a series of feasible allocations that converge to the optimal solution and whose optimality gaps can be quantified. The contents of this chapter are based on

3. W. Wiesemann, D. Kuhn and B. Rustem. *Robust Resource Allocations in Temporal Networks.* Under Revision for Mathematical Programming, 2010.

Chapter 6 investigates Markov decision processes (MDPs), which provide a generic framework that is used to model and solve dynamic net present value maximisation problems in temporal networks. Unfortunately, the solutions of MDPs are often of limited practical use due to their sensitivity to distributional model parameters, which are typically unknown and have to be estimated by the decision maker. To counter the detrimental effects of estimation errors, Chapter 6 considers robust MDPs that offer probabilistic guarantees in view of the unknown parameters. To this end, we assume that an observation history of the MDP is available. Based on this history, we derive a confidence region that contains the unknown parameters with a pre-specified probability $1 - \beta$. Afterwards, we determine a decision that attains the highest worst-case performance over this confidence region. By construction, this decision achieves or exceeds its worst-case performance with a confidence of at least $1 - \beta$. The method involves the solution of tractable conic programs of moderate size. We illustrate how our approach can be applied to temporal networks. The contents of this chapter are based on

4. W. Wiesemann, D. Kuhn and B. Rustem. *Robust Markov Decision processes.* Under Review for Mathematics of Operations Research, 2010.

During my doctoral studies, I was in the fortunate position to collaborate with colleagues on a number of different research projects. Since the resulting publications are not directly related to the topic of this thesis, I shall only list them in the following.

5. D. Kuhn, W. Wiesemann and A. Georghiou. *Primal and Dual Linear Decision Rules in Stochastic and Robust Optimization.* Accepted for Publication in Mathematical Programming, 2009.
6. R. Fonseca, S. Zymler, W. Wiesemann and B. Rustem. *Robust Optimization of Currency Portfolios.* Accepted for Publication in Journal of Computational Finance, 2009.
7. A. Tsoukalas, W. Wiesemann and B. Rustem. *Global Optimisation of Pessimistic Bi-Level Problems.* In: P. M. Pardalos and T. F. Coleman (eds.): Lectures on Global Optimization, Fields Communications Series, American Mathematical Society, 2009.
8. W. Wiesemann, R. Hochreiter and D. Kuhn. *A Stochastic Programming Approach for QoS-Aware Service Composition.* Proceedings of the 8th IEEE International Symposium

on Cluster Computing and the Grid, Lyon, 2008.

9. R. Fonseca, W. Wiesemann and B. Rustem. *International Portfolio Management under Uncertainty.* Under Review for European Journal of Operational Research, 2010.

10. T. Charalambous, E. Klerides and W. Wiesemann. *Transmission Scheduling of Wireless Networks under SINR Constraints.* Under Review for IEEE Transactions on Wireless Communications, 2010.

11. E. Kalyvianaki, W. Wiesemann, Q. H. Vu, D. Kuhn and P. Pietzuch. *Efficient Query Planning with Reuse in Distributed Stream Processing Systems.* Under Review for 36th International Conference on Very Large Data Bases, 2010.

12. S. A. Spacey, W. Wiesemann, D. Kuhn, W. Luk and P. H. J. Kelly. *Robust Software Partitioning.* Under Review for INFORMS Journal on Computing, 2009.

1.3 Notation

By default, all vectors are column vectors. We denote the p-norm of a vector x by $\|x\|_p$. We denote by e_k the kth canonical basis vector, while e denotes the vector whose components are all ones. In both cases, the dimension will usually be clear from the context. We denote the set of real numbers, non-negative real numbers and strictly positive real numbers by $\mathbb{R}$, $\mathbb{R}_+$ and $\mathbb{R}_{++}$, respectively. We denote the set of natural numbers (including zero) by $\mathbb{N}_0$.

We say that a set has a *tractable representation* if set membership can be described by finitely many convex constraints and, potentially, auxiliary variables. Similarly, a function has a tractable representation if its epigraph does. An *explicit* optimisation problem has finitely many variables and constraints.

Some of the chapters in this thesis require additional notation. We defer the introduction of that notation to the relevant chapters.

Chapter 2

Background Theory

We start with a review of deterministic optimisation problems in temporal networks. We then discuss three popular methodologies to model and solve generic optimisation problems under uncertainty. We close with an overview of the issues that arise when these methodologies are applied to temporal networks, and we review the relevant literature. More specific reviews of related work are provided in the Chapters 3–6.

2.1 Temporal Networks

The literature on temporal networks is vast and has been reviewed, amongst others, in [BDM$^+$99, BEP$^+$96, BKPH05, Bru07, DH02, FL04, NSZ03, Pin08, Sch05]. Instead of giving a detailed account of all contributions, we classify some of the most popular research directions according to the three dimensions 'resources', 'network' and 'objective'. More elaborate classification schemes can be found in [BDM$^+$99, Bru07, DH02].

Resource Characteristics. Optimisation problems in temporal networks may assume that a resource allocation has been fixed, or they can involve the assignment of one or multiple resources. In the latter case, we can distinguish between three prevalent types of resources. *Non-renewable resources* are available in pre-specified quantities and are not replenished during the planning horizon. Typical examples of non-renewable resources are cash and man-hours. In contrast, *renewable resources* are replenished every time period, but the decision maker has to meet specified per-period consumption quotas. Examples of renewable resources are processing times on manufacturing machines and processors. In practise, many resources are *doubly-constrained*, that is, they share the restrictions of non-renewable and renewable resources. Other resource characteristics include time windows during which the resources are available, as well as spatial aspects (*e.g.*, immobile resources such as a shipyard).

Network Characteristics. Network characteristics describe the properties of the network tasks and precedences. Tasks are *preemptive* if their processing can be interrupted to execute other tasks. For example, modern operating systems use preemptive multitasking to generate the illusion of executing multiple computer applications in parallel on a single processor. If the execution of network tasks must not be interrupted, then the tasks are called *non-preemptive*. Project scheduling, circuit design and many problems in machine scheduling assume that the network tasks are non-preemptive. In the introduction, we assumed that all precedences in the temporal network are of *finish-start* type, that it, an arc from node i to node j in the temporal network prescribes that task j cannot be started before task i has been completed. Alternatively, one can consider *generalised precedences* that stipulate lower and upper bounds on the time that may pass between the start and completion of any two network tasks. Other network characteristics include time windows during which the tasks must be executed (*e.g.*, ready times and deadlines) and cash flows that arise when certain tasks are processed.

Objective Function. One commonly distinguishes between *regular objective functions*, which are optimised by the early start schedule (1.2), and *nonregular objective functions*, which may not be optimised by the early start schedule. Typical regular objective functions are the makespan and the lateness of the makespan beyond a given deadline. An example of a nonregular objective is the net present value.

The methods developed in this thesis address several combinations of the aforementioned problem characteristics. Chapter 3 assumes that the resource allocation is fixed and maximises the net present value under non-preemptive tasks and generalised precedences. In Chapters 4 and 5

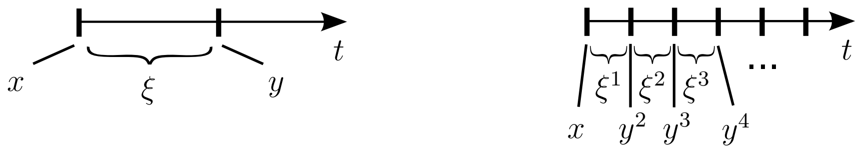


Figure 2.1: Temporal structure of two-stage (left) and multi-stage (right) recourse problems. In the left time line, the wait-and-see decision y may depend on x and ξ. In the right time line, the wait-and-see decision y^t may depend on x and ξ^s, $s < t$.

we determine assignments of non-renewable resources that minimise the makespan under non-preemptive tasks and finish-start precedences. Chapter 6 studies a generic solution technique that is primarily suited for net present value maximisation problems with renewable resources, non-preemptive tasks and finish-start precedences.

2.2 Optimisation under Uncertainty

In practise, most decisions are taken under significant uncertainty about relevant data such as future market developments and resource availabilities. If such decision problems are formulated as optimisation models, the models contain parameters whose values are uncertain. In the following, we review three popular approaches to model and solve optimisation problems with uncertain parameters. In the remainder of the thesis, we will apply these approaches to optimisation problems in temporal networks.

2.2.1 Stochastic Programming

Stochastic programming models the uncertain problem parameters as random variables with known probability distributions. One of the basic models is the *two-stage recourse problem.*

$$\inf_{x \in X} \left\{ f(x) + \mathbb{E}\left[Q(x;\xi) \right] \right\}, \tag{2.1a}$$

where

$$Q(x;\xi) := \inf_{y \in Y(x,\xi)} \left\{ q(y;x,\xi) \right\}. \tag{2.1b}$$

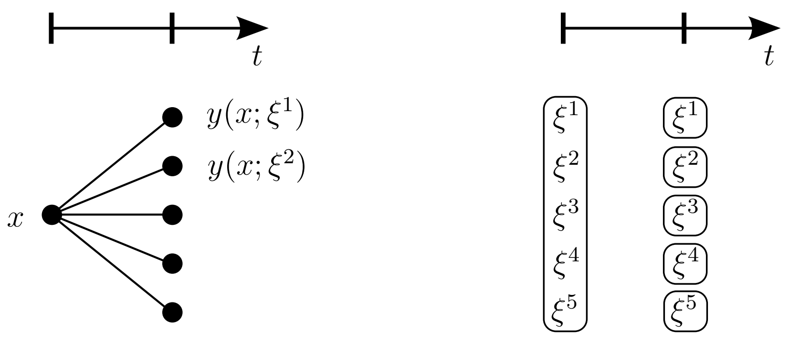


Figure 2.2: Scenario representation of two-stage recourse problems. The left chart shows that for each realisation (scenario) ξ^k of the random vector ξ, a separate recourse decision $y(x;\xi^k)$ can be selected. The right chart visualises the acquisition of information over time. At the beginning of the first time period, the decision maker is unaware of the realised scenario ξ^k. Her information set (*i.e.*, the set of scenarios that may be realised) therefore contains all scenarios. In the second time period, the decision maker knows the realised scenario ξ^k. Her information set has therefore shrunk to one of the singleton sets on the right.

In this problem, the parameter vector ξ is assumed to be uncertain. The decision maker needs to take a *here-and-now decision* $x \in X$ before the value of ξ is known, while the *wait-and-see decision* $y \in Y(x,\xi)$ can be selected under full knowledge of ξ. Conceptually, we can assume that x is chosen at the beginning of time period 1, ξ is revealed during time period 1, and y is selected at the beginning of time period 2 (after ξ is known), see Figure 2.1 (left). The goal is to minimise the sum of first-stage costs $f(x)$ and expected second-stage costs $\mathbb{E}\left[Q(x;\xi)\right]$, where the expectation is taken with respect to ξ. Note that for any value of x and ξ, the second-stage problem $Q(x;\xi)$ is deterministic. If there is a finite set of values $\xi^1, \xi^2, \ldots$ such that $\xi \in \{\xi^1, \xi^2, \ldots\}$ with probability one, then (2.1) can be formulated as an explicit optimisation problem. Otherwise, (2.1) can be approximated by a surrogate model that replaces the probability distribution of ξ with a finite-valued approximation. In either case, the resulting optimisation model has the structure of a *scenario fan* whose branches represent the possible realisations of ξ, see Figure 2.2.

Several variations of problem (2.1) are common. On one hand, the expected value in (2.1a) is often replaced with other risk measures such as the (conditional) value-at-risk or the variance. On the other hand, the two-stage structure (decision – realisation of uncertainty – decision) can be extended to multiple decision stages. In a *multi-stage recourse problem*, the parameter vector ξ can be subdivided into vectors $\xi^1, \ldots, \xi^T$ such that $\xi = (\xi^1, \ldots, \xi^T)$ and ξ^t is revealed

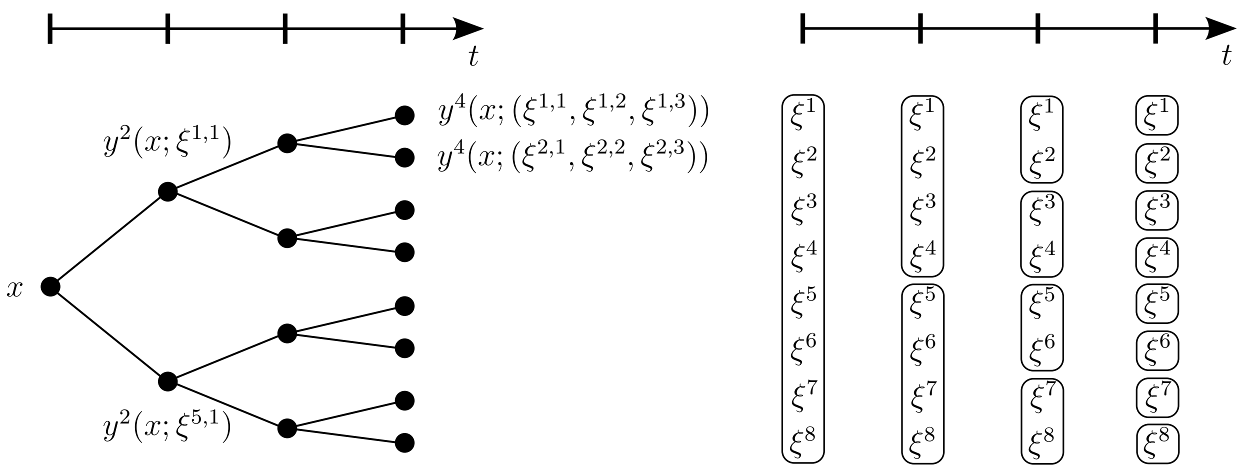

t
t
x
$y^2(x;\xi^{1,1})$
$y^2(x;\xi^{5,1})$
$y^4(x;(\xi^{1,1},\xi^{1,2},\xi^{1,3}))$
$y^4(x;(\xi^{2,1},\xi^{2,2},\xi^{2,3}))$
ξ^1 ξ^2 ξ^3 ξ^4 ξ^5 ξ^6 ξ^7 ξ^8

Figure 2.3: Scenario representation of multi-stage recourse problems. In analogy to Figure 2.2, $\xi^{k,t}$ denotes the tth subvector of the scenario $\xi^k = (\xi^{k,1}, \ldots, \xi^{k,T})$. In the chart on the left, each path from the root node to a leaf node constitutes one scenario. Two scenarios ξ^k and ξ^l are undistinguishable at the beginning of period t if $\xi^{k,s} = \xi^{l,s}$ for all $s < t$. In this case, ξ^k and ξ^l are contained in the same information set at time t, and non-anticipativity stipulates that $y^t(x;(\xi^{k,1}, \ldots, \xi^{k,t-1})) = y^t(x;(\xi^{l,1}, \ldots, \xi^{l,t-1}))$. For example, non-anticipativity requires that $y^2(x;\xi^{k,1}) = y^2(x;\xi^{l,1})$ for $k, l \in \{1, \ldots, 4\}$ and $y^3(x;(\xi^{5,1},\xi^{5,2})) = y^3(x;(\xi^{6,1},\xi^{6,2}))$.

during time period $t = 1, \ldots, T$. The decision maker can take a recourse decision y^t at the beginning of every time period $t = 2, \ldots, T+1$, and y^t may depend on the values of $\xi^1, \ldots, \xi^{t-1}$, see Figure 2.1 (right). Note that y^t may *not* depend on the values of ξ^s, $s \geq t$, since this information is not available at the time the recourse decision is taken. This causality requirement is called *non-anticipativity*. If the probability distribution of ξ has finitely many values, then the optimisation model associated with a multi-stage recourse problem has the structure of a *scenario tree*, see Figure 2.3. While convex two-stage recourse problems can be efficiently approximated, multi-stage problems 'generically are computationally intractable already when medium-accuracy solutions are sought' [SN05]. We will revisit recourse problems in Chapter 3, where we model a net present value maximisation problem as a two-stage recourse problem.

Apart from recourse problems, stochastic programming studies problems with *chance constraints*. The basic two-stage chance constrained problem can be formulated as follows.

$$\inf_{x \in X} \left\{ f(x) \,:\, \mathbb{P}\left(Q(x;\xi) \leq 0\right) \geq 1 - \epsilon \right\}, \tag{2.2}$$

where Q is defined in (2.1b). The temporal structure of problem (2.2) is the same as for two-stage recourse problems, see Figure 2.1 (left). The goal is to find a here-and-now decision x such that with a probability of at least $1 - \epsilon$, there is a wait-and-see decision $y(x;\xi) \in Y(x,\xi)$ that satisfies $q(y(x;\xi);x,\xi) \leq 0$. Chance constrained problems are notoriously difficult to solve. Indeed, even if the second-stage problem Q is a linear program, the feasible region of (2.2) is typically nonconvex and disconnected. Moreover, calculating the left-hand side of the constraint in (2.2) requires the evaluation of a multi-dimensional integral, which itself constitutes a difficult problem. As a result, most solution approaches for (2.2) settle for approximate solutions. Similar to recourse problems, chance constrained problems can be extended to multiple decision stages. In Chapter 4 we will model a makespan minimisation problem as a two-stage chance constrained problem.

For an in-depth treatment of stochastic programming, see [KW94, Pré95, RS03].

2.2.2 Robust Optimisation

In its basic form, robust optimisation studies semi-infinite problems of the following type.

$$\inf_{x \in X} \left\{ f(x) \,:\, g_i(x;\xi) \leq 0 \;\; \forall \xi \in \Xi,\; i = 1, \ldots, I \right\} \tag{2.3}$$

We interpret x as a here-and-now decision and ξ as an uncertain parameter vector with support Ξ. The goal is to minimise the deterministic costs $f(x)$ while satisfying the constraints for all possible realisations of ξ. Note that (2.3) is a single-stage problem since it does not contain any recourse decisions. If Ξ constitutes a finite set of scenarios $\xi^1, \xi^2, \ldots$, then (2.3) can be formulated as an explicit optimisation problem. If Ξ is of infinite cardinality, then (2.3) can be solved with iterative solution procedures from semi-infinite optimisation [HK93]. One of the key contributions of robust optimisation has been to show that for sets Ξ of infinite cardinality but specific structure, one can apply duality theory to transform problem (2.3) into an explicit optimisation problem. We illustrate this approach with an example.

Example 2.2.1 *Assume that $I = 1$, $X \subseteq \mathbb{R}^n$, $\Xi = \left\{ \xi \in \mathbb{R}^k_+ : W\xi \leq h \right\}$ for $W \in \mathbb{R}^{m \times k}$ and*

$h \in \mathbb{R}^m$ and $g_1(x;\xi) = \xi^\top A\,x$ for $A \in \mathbb{R}^{k\times n}$. Also assume that Ξ is non-empty and bounded. We can then reformulate the constraint in (2.3) as follows.

$$\begin{aligned}
g_1(x;\xi) \leq 0 \;\; \forall \xi \in \Xi \quad &\Leftrightarrow \quad \sup_{\xi \in \Xi} \{g_1(x;\xi)\} \leq 0 \\
&\Leftrightarrow \quad \max_{\xi \in \mathbb{R}^k_+} \left\{\xi^\top A\,x \,:\, W\xi \leq h\right\} \leq 0 \\
&\Leftrightarrow \quad \min_{\lambda \in \mathbb{R}^m_+} \left\{h^\top \lambda \,:\, W^\top \lambda \geq Ax\right\} \leq 0 \\
&\Leftrightarrow \quad h^\top \lambda \leq 0,\; W^\top \lambda \geq Ax \qquad \textit{for some } \lambda \in \mathbb{R}^m_+
\end{aligned}$$

Here, the third equivalence follows from linear programming duality. We have thus transformed the semi-infinite constraint in (2.3) into a finite number of constraints that involve x and some auxiliary variables λ.

Much of the early work on robust optimisation focuses on generalisations of the reformulation scheme illustrated in Example 2.2.1. Unfortunately, single-stage models such as (2.3) are too restrictive for decision problems in temporal networks. Indeed, the task start times can typically be chosen as a wait-and-see decision, and optimisation problems that account for this flexibility provide significantly better solutions. We discuss this issue in more detail in the next section and in Chapters 3–6. We are therefore interested in *two-stage robust optimisation problems* such as the following one.

$$\inf_{x \in X} \; \sup_{\xi \in \Xi} \; \inf_{y \in Y(x,\xi)} \; \{f(x) + q(y;x,\xi)\} \tag{2.4}$$

Here, q is the objective function of the second-stage problem Q defined in (2.1b), and $Y(x,\xi) \subseteq \mathbb{R}^l_+$. In this problem, the here-and-now decision x is accompanied by a wait-and-see decision $y \in Y(x,\xi)$ that can be selected under full knowledge of ξ. The temporal structure of this problem is analogous to the two-stage recourse problem (2.1), see Figure 2.1 (left). The goal is to minimise the sum of first-stage costs $f(x)$ and worst-case second-stage costs $\sup_{\xi \in \Xi} Q(x;\xi)$, see (2.1b), where the worst case is taken with respect to ξ. Two-stage robust optimisation

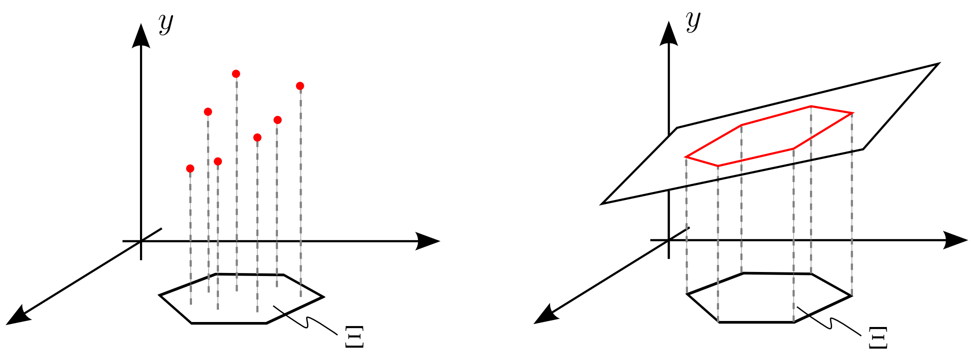


Figure 2.4: Approximations employed by two-stage recourse problems (left) and two-stage robust optimisation problems (right) for a random vector ξ with a continuous probability distribution. In the left chart, the support Ξ of ξ is replaced with a discrete-valued probability distribution. For each possible realisation (scenario) ξ^k, an individual second-stage decision $y(x;\xi^k)$ may be chosen. In the right chart, the support Ξ remains unchanged, but the second-stage decision $y(x;\xi)$ is restricted to be an affine function of ξ.

problems are generically intractable, see [BTGGN04]. A tractable approximation can be derived from the following identity.

$$\inf_{x\in X}\ \sup_{\xi\in\Xi}\ \inf_{y\in Y(x,\xi)}\ \{f(x)+q(y;x,\xi)\} \quad = \quad \inf_{\substack{x\in X,\\ y\in\mathcal{Y}(x)}}\ \sup_{\xi\in\Xi}\{f(x)+q(y(\xi);x,\xi)\}\,, \tag{2.5a}$$

where for $x \in X$,

$$\mathcal{Y}(x) = \left\{(y:\Xi\mapsto\mathbb{R}^l_+)\ :\ y(\xi)\in Y(x,\xi)\ \ \forall\,\xi\in\Xi\right\}. \tag{2.5b}$$

The identity (2.5a) allows us to reduce the min-max-min problem (2.4) to the min-max problem on the right-hand side of (2.5a) at the cost of augmenting the set of first-stage decisions. For a given here-and-now decision $x \in X$, $\mathcal{Y}(x)$ denotes the space of all functions on Ξ that map parameter realisations to feasible wait-and-see decisions. A function y is called a *decision rule* because it specifies the second-stage decision in (2.4) as a function of the uncertain parameters ξ. Note that the choice of an appropriate decision rule on the right-hand side of (2.5a) is part of the first-stage decision. The identity (2.5a) holds regardless of the properties of X and Ξ because $\mathcal{Y}(x)$ does not impose any structure on the decision rules (such as measurability).

Since $\mathcal{Y}(x)$ constitutes a function space, further assumptions are required to ensure that the problem on the right-hand side of (2.5a) can be solved. A popular approach is to restrict $\mathcal{Y}(x)$ to the space of affine or piecewise affine functions of ξ, see [BTGN09, CSSZ08]. As we will show in Chapter 5, this restriction allows us to reformulate the model on the right-hand side of (2.5a) as an explicit optimisation problem. Figure 2.4 compares the scenario approximation from the previous section with the decision rule approximation.

In Chapter 5 we will solve a makespan minimisation problem as a two-stage robust optimisation problem. Instead of approximating the optimal second-stage decision via decision rules, however, this chapter will develop convergent lower and upper bounds on the optimal value of the problem. The upper bounds correspond to feasible solutions whose objective values are bracketed by the bounds. We will compare our method with two popular classes of decision rules. Moreover, in Chapter 6 we will apply robust optimisation to immunise stochastic dynamic programs against estimation errors. In that chapter, we will employ decision rules to approximate several $\mathcal{NP}$-hard optimisation problems.

For an introduction to robust optimisation, see [BS04, BTGN09]. Two-stage robust optimisation problems are discussed in [BTGN09, CSSZ08, JLF07, LJF04, LLMS09, Sti09].

2.2.3 Stochastic Dynamic Programming

Stochastic dynamic programming studies the modelling and solution of optimisation problems via *Markov decision processes* (MDPs). MDPs allow to model dynamic decision problems in which the outcomes are partly random and partly under the control of the decision maker. At each time period, the MDP is in some state s, and the decision maker takes an action a. The state s' in the successive time period is random and depends on both the current state s and the selected action a. However, the new state does *not* depend on any other past states or actions: this is the *Markov property.* For each transition of the MDP, the decision maker receives a reward that depends on the old state, the new state and the action that triggered the transition.

For the purposes of this thesis, it will suffice to consider discrete-time MDPs with finite state and action spaces. We therefore assume that an MDP is defined through its state space $\mathcal{S} = \{1, \ldots, S\}$, its action space $\mathcal{A} = \{1, \ldots, A\}$ and a discrete planning horizon $\mathcal{T} = \{0, 1, 2, \ldots\}$

that can be finite or infinite. The initial state is a random variable with known probability distribution p_0. If action $a \in \mathcal{A}$ is chosen in state $s \in \mathcal{S}$, then the subsequent state is $s' \in \mathcal{S}$ with probability $p(s'|s,a)$. We assume that the probabilities $p(s'|s,a)$, $s' \in \mathcal{S}$, sum up to one for each state-action pair $(s,a) \in \mathcal{S} \times \mathcal{A}$. The decision maker receives an expected reward of $r(s,a,s') \in \mathbb{R}$ if action $a \in \mathcal{A}$ is chosen in state $s \in \mathcal{S}$ and the subsequent state is $s' \in \mathcal{S}$. Without loss of generality, we can assume that every action is admissible in every state. Indeed, if action $a \in \mathcal{A}$ is not allowed in state $s \in \mathcal{S}$, we can 'forbid' this action by setting all rewards $r(s,a,s')$, $s' \in \mathcal{S}$, to a large negative value. For the objective functions that we consider, we can furthermore assume that all rewards $r(s,a,s')$ are non-negative. This can always be achieved by adding a sufficiently large positive constant to each reward $r(s,a,s')$.

The MDP is controlled through a policy $\pi = (\pi_t)_{t \in \mathcal{T}}$, where $\pi_t(a|s_0, a_0, \ldots, s_{t-1}, a_{t-1}; s_t)$ represents the probability to choose action $a \in \mathcal{A}$ if the current state is s_t and the state-action history is given by $(s_0, a_0, \ldots, s_{t-1}, a_{t-1})$. Note that contrary to the state transitions of the MDP, the policy π need not be Markovian. If the planning horizon $\mathcal{T}$ is infinite, then we evaluate a policy π in view of its *expected total reward* under the discount factor $\lambda \in (0,1)$:

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \lambda^t r(s_t, a_t, s_{t+1}) \,\middle|\, s_0 \sim p_0\right] \tag{2.6}$$

Here, $\mathbb{E}$ denotes the expectation with respect to the random process defined by the transition probabilities p and the policy π. The notation $s_0 \sim p_0$ indicates that the initial state s_0 is a random variable with probability distribution p_0. We will define an analogous objective function for finite horizon MDPs in Chapter 6. For a fixed policy π, the *policy evaluation problem* asks for the value of expression (2.6). The *policy improvement problem*, on the other hand, asks for a policy π that maximises (2.6). For the objective (2.6), the policy evaluation and improvement problems can be solved efficiently via policy and value iteration.

Example 2.2.2 (Inventory Management) *Consider the following infinite horizon inventory problem. At the beginning of each time period, the decision maker can order $a \in \mathbb{N}_0$ units of a product at unit costs c. The ordered products arrive at the beginning of the next time period. During each time period, an independent and identically distributed random demand δ arises* *for the product. This demand is served at a unit price p from the current inventory, and there is no backlogging (i.e., demand that cannot be satisfied within the period is lost). The inventory can hold at most I units of the product. The goal is to find an inventory control policy that maximises the expected total reward under some discount factor* λ.

We can formulate this problem as an infinite horizon MDP as follows. The state set $\mathcal{S} = \{0, \ldots, I\}$ *describes the inventory level at the beginning of each time period. In state* $s \in \mathcal{S}$*, the admissible actions* $\{0, \ldots, I - s\}$ *determine the order quantity. Note that the actions are state-dependent in this example. The transition probabilities are*

$$p(s'|s,a) = \begin{cases} \mathbb{P}(\delta = s + a - s') & \text{if } s' \neq 0, \\ \sum_{i=s+a}^{\infty} \mathbb{P}(\delta = i) & \text{otherwise,} \end{cases}$$

and the rewards are given by $r(s, a, s') = p(s + a - s') - ca$*. Here we assume that the random demand* δ *is non-negative with probability one. A policy* π *could order* $\omega \in \mathbb{N}$ *units whenever the current inventory falls below some threshold* $\Omega \in \mathbb{N}_0$*. This policy is defined through* $\pi_t(a|s_0, a_0, \ldots, s_{t-1}, a_{t-1}; s_t) = 1$ *if* $s_t < \Omega$ *and* $a = \omega$ *and* $\pi_t(a|s_0, a_0, \ldots, s_{t-1}, a_{t-1}; s_t) = 0$ *otherwise. Note that this policy* π *is Markovian.*

Most of the literature on MDPs assumes that the expected rewards r and the transition kernel P are known, with a tacit understanding that they have to be estimated in practise. However, it is well-known that the expected total reward (2.6) can be very sensitive to small changes in r and P, see [MSST07]. Thus, a decision maker is confronted with two different sources of uncertainty. On one hand, she faces *internal variation* due to the stochastic nature of MDPs. On the other hand, she needs to cope with *external variation* because the estimates for r and P may deviate from their true values. In Chapter 6 we will apply robust optimisation to counter the detrimental effects of estimation errors. We will furthermore show how MDPs can be used to solve multi-stage net present value maximisation problems in temporal networks.

There are numerous variations of the Markov decision process defined in this section. For an overview of the major models and solution approaches, see [Ber07, Put94].

2.3 Optimisation of Temporal Networks under Uncertainty

Decisions in temporal networks are often taken under significant uncertainty about the network structure (*i.e.*, the tasks and precedences of the network), the task durations, the ready times and deadlines of the tasks, the cash flows and the availability of resources. In this thesis, we focus on problems in which the task durations (Chapters 3–6), the cash flows (Chapters 3 and 6) and the tasks' ready times and deadlines (Chapter 3) are uncertain. Problems with uncertain network structure are studied in the literature on GERT networks, see [Neu79, Neu99, Pri66]. A problem that accounts for uncertain resource availabilities is considered in [Yan05].

An optimisation problem under uncertainty needs to specify *when* information about the uncertain parameters becomes available, and *what* information is revealed about them. Both issues are straightforward in the optimisation problems reviewed in Section 2.2. In a multi-stage recourse problem, for example, we observe the subvector ξ^t of the uncertain parameters ξ at the beginning of time period $t + 1$, see Figure 2.1 (right). Likewise, in a stochastic dynamic program, we observe the current state of the MDP at the beginning of each time period.

The situation is different for temporal networks, and it is this difference that complicates the modelling and solution of decision problems in temporal networks. It is customary to assume that the duration and cash flow of a task is observed when the task is completed. However, the completion time of a task depends on the task's start time, which is chosen by the decision maker. Hence, in contrast to the problems studied in Section 2.2, the times at which we learn about the random parameters depend on the chosen decision. Recourse problems with decision-dependent uncertainty are studied in [GG06, JWW98], and a robust optimisation problem with decision-dependent uncertainty is formulated in [CGS07]. However, the resulting optimisation problems are computationally demanding, and they typically have to undergo drastic simplifications before they can be solved.

Apart from the time points at which information becomes available, optimisation problems in temporal networks differ from other problems in the type of the revealed information. In many cases, the task durations and cash flows in a temporal network do not correspond to individual parameters, but they are functions of multiple parameters (as is the case in factor models). In such problems, we do not observe the uncertain parameters themselves, but we accumulate knowledge about them with the completion of each task. We can use this information to exclude parameter realisations that are not compatible with the observed durations and cash flows. In contrast, the multi-stage recourse problem reviewed in Section 2.2.1 assumes that the decision maker can directly observe the uncertain parameter vector ξ.

Similar to the problems in Section 2.2, optimisation problems in temporal networks can contain here-and-now as well as wait-and-see decisions. Here-and-now decisions are taken before any of the network tasks are started, whereas a wait-and-see decision associated with task $i \in V$ (*e.g.*, its start time or resource assignment) may depend on all information that is available at the time task i is started. Since the early start schedule optimises regular objective functions (see Section 2.1), it is relatively straightforward to model the task start times as a wait-and-see decision in makespan minimisation problems. We will consider problems with a here-and-now resource allocation and wait-and-see task start times in Chapters 4 and 5. The situation is fundamentally different in net present value maximisation problems where the early start schedule is no longer guaranteed to be optimal. In Chapters 3 and 6 we consider net present value problems in which the resource allocation is fixed, while the task start times can be chosen as a wait-and-see decision.

We close this section with an overview of the literature on temporal networks under uncertainty. Detailed reviews of specific topics will be provided in later chapters.

Although temporal networks have been analysed for more than fifty years [Ful61, Kel61, MRCF59], the literature on temporal networks under uncertainty is surprisingly sparse. Until recently, most research on temporal networks under uncertainty assumed a fixed resource allocation and focused on the makespan of the early start schedule. Following the classification in [MÖ1], we can categorise the literature into methods that identify 'critical' tasks or task paths [Elm00], simulation techniques to approximate the makespan distribution [AK89], approaches that bound the expected makespan [BM95, BNT02, MN79], and methods that bound the cumulative distribution function of the makespan [LMS01, MÖ1].

Optimisation problems that maximise a network's net present value under uncertainty generally model the task start times as a wait-and-see decision, while the resource allocation is assumed to be fixed. The problem has been approximated by a two-stage recourse model in [Ben06], where an optimal delay policy is sought that prescribes how long each task should be delayed beyond its earliest start time. Under the assumption that the task durations are independent and exponentially distributed, the net present value maximisation problem is formulated as a continuous-time Markov decision process in [BR97, TSS06]. Finally, approximate solutions for net present value maximisation problems have been obtained with a number of heuristics, see [Bus95, OD97, Ö98, TFC98, WWS00]. For an overview of net present value maximisation problems in temporal networks, see [HDD97].

Makespan minimisation problems under uncertainty typically assume that a resource assignment is selected here-and-now, while the task start times are modelled as a wait-and-see decision. For non-renewable resources, the makespan minimisation problem has been formulated as a two-stage recourse model in [Wol85] and as a robust optimisation problem in [CGS07, CSS07, JLF07, LJF04]. Except for [CGS07], all of these contributions model the resource assignment as a here-and-now decision. A makespan minimisation problem with renewable resources is studied in [MS00].

For an in-depth review of optimisation problems in temporal networks under uncertainty, see [AK89, BKPH05, Elm05, HL04, HL05, JW00, LI08, MÖ1, Pin08, Sah04].

Chapter 3

Maximisation of the Net Present Value

3.1 Introduction

This chapter studies a temporal network whose tasks give rise to cash flows. Positive cash flows denote cash inflows (*e.g.*, received payments), whereas negative cash flows represent cash outflows (*e.g.*, accrued costs). We maximise the network's net present value (NPV), which is the discounted sum of all arising cash flows. NPV maximisation problems arise in project management, process scheduling and several other application areas. For example, in capital-intensive IT and construction projects, large amounts of money are invested over long periods of time, and the wise coordination of cash in- and outflows crucially affects the profitability of such projects. The NPV can be regarded as the 'cash equivalent' of undertaking a project.

We consider temporal networks whose task durations and cash flows are described by a discrete set of alternative scenarios with associated occurrence probabilities. Since the cash flows can be positive or negative, the early start policy (1.2) does not yield an optimal solution, see Section 1.1. Similarly, the choice of scenario-independent task start times frequently leads to infeasible schedules or severe losses in revenues. In Chapter 6 we will determine truly adaptive schedules for NPV maximisation problems that react to the uncertainties revealed over time. However, such schedules become computationally demanding for large networks. To overcome this difficulty, this chapter determines an optimal (scenario-independent) target processing time

(TPT) policy for the network tasks. In case task $i \in V$ could be started earlier than its TPT in the realised scenario, it will be postponed to its TPT. If, on the other hand, task i cannot be started at its TPT (because preceding tasks finish late), then it will be started as soon as possible thereafter. Following the terminology from Section 2.2.1, we solve a two-stage recourse problem in which the TPT policy is chosen here-and-now, whereas the factual task start times are modelled as a wait-and-see decision. The class of TPT policies is a strict subset of the class of non-anticipative scheduling decisions. By restricting ourselves to this class, we can solve the NPV maximisation problem for networks of non-trivial size. Our model accommodates generalised precedence relations [EK90, EK92] but disregards resource restrictions. We discuss these assumptions in Section 3.3.

The remainder of this chapter is organised as follows. In the next section we summarise related literature. Section 3.3 introduces our problem formulation, while Section 3.4 describes the components of a branch-and-bound solution procedure. Section 3.5 presents and interprets the results of an extensive numerical study. We conclude in Section 3.6.

3.2 Literature Review

Maximising the NPV of a temporal network was first suggested in [Rus70].[1] The paper considers problem instances (G, ζ, d, β), where $G = (V, E)$ represents the structure of the temporal network, ζ_i the cash flow arising at the start time of task $i \in V$, d_i the duration of task i and $\beta = 1/(1 + \alpha)$ the discount factor with internal rate of return $\alpha > 0$. An arc $(i, j) \in E$ prescribes that task j must not be started before task i has been completed. The assumption that the cash flows are realised at the beginning of the tasks is not restrictive; we come back to this point in Section 3.3. All parameters are assumed to be deterministic, and there are no

[1]We will use a consistent notation for all models reviewed in this section. Therefore, we may slightly modify some of the original formulations without changing their meaning.

resource restrictions. The objective is to

$$
\begin{aligned}
\underset{y}{\text{maximise}} \quad & \sum_{i \in V} \zeta_i \beta^{y_i} && \text{(3.1a)} \\
\text{subject to} \quad & y \in \mathbb{R}^n \\
& y_j \geq y_i + d_i \qquad \forall\, (i,j) \in E, && \text{(3.1b)} \\
& y_1 = 0. && \text{(3.1c)}
\end{aligned}
$$

In this formulation, the components of the decision vector y represent the task start times. The objective function maximises the sum of discounted cash flows. The constraints ensure satisfaction of the precedence relations and non-negativity of the schedule. Without loss of generality, it is assumed that the first task is started at time zero. A deadline Δ can be imposed by adding the constraint $y_n \leq \Delta$. In [Rus70], (3.1) is solved through a sequence of linear programs whose objective functions are obtained by linearisation around the current candidate solution. The duals of these approximations can be formulated as network flow problems, and the author proves local convergence of the overall procedure.

It is shown in [Gri72] that the variable substitution $z_i := \beta^{y_i}$ converts (3.1) into an equivalent linear program:

$$
\begin{aligned}
\underset{z}{\text{maximise}} \quad & \sum_{i \in V} \zeta_i z_i \\
\text{subject to} \quad & z \in \mathbb{R}^n \\
& z_j \leq \beta^{d_i} z_i \qquad \forall\, (i,j) \in E, \\
& z_1 = 1, \\
& z_n \geq 0.
\end{aligned}
$$

A deadline Δ can be enforced by replacing the last constraint with $z_n \geq \beta^{\Delta}$. In [Gri72] this problem is solved with a network simplex variant.

In [NZ00], the network simplex algorithm from [Gri72] is extended to temporal networks $\Gamma = (G, \zeta, d, \beta)$ with generalised precedences. Here, $G = (V, E)$ represents the network structure,

ζ_i the cash flow arising at the start time of task $i \in V$, d_{ij} the minimum time lag between the start times of tasks i and j ($d_{ij} < 0$ is allowed; hence, the precedences are called 'generalised') and β the discount factor.[2] The authors solve problems of the following type.

$$\begin{aligned}
&\underset{y}{\text{maximise}} && g(y) := \sum_{i \in V} \zeta_i \beta^{y_i} && \text{(3.2a)}\\
&\text{subject to} && y \in \mathbb{R}^n && \\
& && y_j \geq y_i + d_{ij} \qquad \forall\,(i,j) \in E, && \text{(3.2b)}\\
& && y_1 = 0. && \text{(3.2c)}
\end{aligned}$$

It is shown in [DH02, SZ01] that the algorithm presented in [NZ00] performs favourably in practise. In Section 3.4.1 we will use this algorithm to solve subproblems that arise in our branch-and-bound procedure.

In [EH90], an approximate solution procedure to the NPV maximisation problem is proposed, while an exact method based on the steepest ascent principle is developed in [SZ01]. Comparisons of the various approaches can be found in [DH02, SZ01].

Over the last two decades, numerous publications have addressed extensions of the deterministic NPV maximisation problem, the majority of which allow for different types of resource constraints. We do not provide more details on those efforts and refer the interested reader to the extensive surveys [DH02, HDD97]. Interestingly, the incorporation of uncertainty has attracted significantly less attention, just as temporal networks under uncertainty have in general been neglected for a long time.

Assuming independent and exponentially distributed task durations, a stochastic version of (3.1) is considered in [BR97, TSS06]. Both contributions employ continuous-time Markov chains whose states assign labels 'not yet started', 'in progress' and 'finished' to all tasks. Continuous-time Markov chains were first applied to temporal networks in [KA86]. The restriction to exponentially distributed durations can be relaxed at the cost of augmenting the state space. Unfortunately, the number of states in the Markov chain grows exponentially with the network

[2]Generalised precedences are explained further in Section 3.3.

size, even when assuming exponentially distributed task durations. As a result, these methods are primarily applicable to small networks.

Several heuristics have been suggested for general task duration distributions. A suboptimal task delay policy is determined via simulation-based optimisation in [Bus95]. The author points out that the problem is very challenging since the objective function is highly variable but flat near the (suspected) optimum. A simulated annealing heuristic for discretised task duration distributions is presented in [WWS00]. A general solution approach for stochastic temporal networks based on floating factor policies can be found in [TFC98]. The authors define the total float of task $i \in V$ as the difference between its latest (κ_i) and earliest (λ_i) start times given some deadline and average task durations. For a fixed float factor $\alpha \in [0, 1]$, task i should be started as early as possible but not before time $\lambda_i + \alpha(\kappa_i - \lambda_i)$. It is suggested to evaluate the impact of α via Monte Carlo simulation and to choose the value of α that minimises a composite risk measure (e.g., the probability that the makespan or the overall costs exceed specified tolerances).

The method that comes closest to ours is developed in [Ben06]. Again, the network structure is assumed to be given as $G = (V, E)$, where an arc $(i, j) \in E$ stipulates that task j cannot be started before task i has been finished. The author assumes that the cash flows are deterministic, whereas finitely many scenarios $s \in S$ with associated occurrence probabilities p_s specify the uncertain task durations d_i^s, $i \in V$. The goal is to maximise the expected net present value over all scenarios, which is done heuristically by determining a delay policy with the following two-stage procedure. In the first stage, the optimal 'average' task start times are approximated as follows.

$$\begin{aligned}
\underset{y}{\text{maximise}} \quad & \sum_{i \in V} \zeta_i \beta^{y_i} \\
\text{subject to} \quad & y \in \mathbb{R}^n \\
& y_j \geq \sum_{s \in S} p_s \max_{i \in V} \left\{ y_i + d_i^s \; : \; (i, j) \in E \right\} \qquad \forall j \in V \setminus \{1\}, \\
& y_1 = 0.
\end{aligned}$$

In this model, cash flow ζ_i is assumed to arise at the start time y_i of task $i \in V$; we refer to Section 3.3 for a further discussion. The precedence constraints are imposed to hold in expectation. This model is not convex, and the author uses a local search procedure to obtain locally optimal start times. In the second stage, a fixed-delay policy r is determined by setting $r_1 := y_1^*$ and

$$r_j := y_j^* - \sum_{s \in S} p_s \max_{i \in V} \{y_i^* + d_i^s \,:\, (i,j) \in E\} \qquad \forall\, j \in V \setminus \{1\}\,,$$

where y^* denotes an optimal solution to the first-stage problem. The fixed-delay policy r prescribes to delay the start time of task $i \in V$ by r_i time units (compared to its earliest possible start time, which itself depends on the realised scenario). This approach is very attractive from a computational point of view, but it does not give any guarantees with respect to optimality. We will revisit this method in Section 3.5 when we compare its solutions with schedules obtained from TPT policies.

Finally, we mention the contributions [OD97, Ö98], which maximise a network's NPV subject to capital constraints and multiple task execution modes. Capital is treated as a randomly replenished resource which can be temporarily acquired at given costs. The authors present an online scheduling heuristic to solve this problem.

3.3 Problem Formulation

We study temporal networks in activity-on-node notation (see Section 1.1) with generalised precedence relations. This means that we allow for both minimum and maximum time lags between the start and completion times of the network tasks. A minimum time lag of length $\delta \geq 0$ between the start times of tasks i and j is modelled as a precedence relation $(i,j) \in E$ with positive value $d_{ij} = \delta$, whereas a similar maximum time lag of length $\delta \geq 0$ corresponds to a precedence relation $(j,i) \in E$ with negative value $d_{ji} = -\delta$. This allows us to represent both minimum and maximum time lags by inequalities of type $y_q \geq y_p + d_{pq}$, $(p,q) \in E$, where y_p and y_q represent the start times of tasks p and q, respectively. Since the completion time of a task equals its start time plus its duration, this approach immediately extends to time lags specified in terms of both start and completion times. A finish-start precedence between tasks i and j (*i.e.*, 'j cannot start before i has been completed') reduces to a minimum time lag of value 0 between the completion time of i and the start time of j. We refer to [EK90, EK92] for a detailed discussion of generalised precedence relations, together with convenient ways of specifying temporal networks in this format.

We consider problem instances $\Upsilon = (G, S, p, \zeta, d, \Delta, \beta)$, where $G = (V, E)$ represents the network structure and $S = \{1, \ldots, m\}$ the index set of discrete scenarios with occurrence probabilities p_s for $s \in S$. ζ_i^s denotes the cash flow arising at the start time of task $i \in V$ in scenario $s \in S$. The assumption that cash flows arise at the task start times is not restrictive: imagine, for example, that a cash flow z_i^s in scenario $s \in S$ arises when task $i \in V$ is *completed*. Assuming that the discount factor is β and that the duration of task i amounts to δ_i^s in scenario s, the end-of-task cash flow z_i^s is equivalent to a cash flow $\zeta_i^s = \beta^{\delta_i^s} z_i^s$ at the start time of task i in scenario s. The value of precedence relation $(i, j) \in E$ in scenario $s \in S$ is denoted by d_{ij}^s. Without loss of generality, we assume that for a given precedence $(i, j) \in E$, d_{ij}^s is of equal sign for all $s \in S$. We can then define the subset of positive-valued and negative-valued precedence relations by $E^+ = \big\{(i, j) \in E : d_{ij}^s \geq 0 \; \forall \, s \in S\big\}$ and $E^- = E \setminus E^+$, respectively. In Section 1.1 we stipulated that task 1 (n) constitutes the unique source (sink) of the network. In the light of generalised precedence relations, we now impose the same requirements for the subgraph $G = (V, E^+)$. In order to avoid unbounded problem instances, that is, instances in which it is beneficial to delay some network tasks indefinitely, we assume that there is a scenario-independent deadline Δ. Since we can choose Δ as large as we wish, this assumption does not restrict the generality of our model. As before, β denotes the discount factor.

With the notation introduced above, our model can be formulated as follows.

$$\begin{aligned}
&\underset{r,y}{\text{maximise}} && f(r,y) := \sum_{s\in S} p_s \sum_{i\in V} \zeta_i^s \beta^{y_i^s} && \text{(3.3a)}\\
&\text{subject to} && r\in\mathbb{R}^n, \quad y\in\mathbb{R}^{nm} && \\
& && y_j^s = \max\left\{\sup_{i\in V}\left\{y_i^s + d_{ij}^s \,:\, (i,j)\in E^+\right\}, r_j\right\} \qquad \forall\, j\in V,\, s\in S, && \text{(3.3b)}\\
& && y_j^s \geq y_i^s + d_{ij}^s \qquad \forall\,(i,j)\in E^-,\, s\in S, && \text{(3.3c)}\\
& && y_n^s \leq \Delta \qquad \forall\, s\in S, && \text{(3.3d)}\\
& && r_j \geq 0 \qquad \forall\, j\in V. && \text{(3.3e)}
\end{aligned}$$

For future use, we also define $\mathcal{Y}_\Upsilon := \{(r,y)\in\mathbb{R}^n\times\mathbb{R}^{nm} : (r,y) \text{ satisfies (3.3b)–(3.3e)}\}$. In model (3.3), r represents the desired TPT policy. This policy is chosen here-and-now, that is, before the realised scenario is known. The wait-and-see decision y_i^s denotes the factual start time of task $i\in V$ if scenario $s\in S$ is realised. Since the cash flows are realised at the task start times, the objective function represents the expected NPV over all scenarios. (3.3b) uniquely specifies the task start times y as a function of preceding start times and the TPT policy r. In particular, the start time of task j in scenario s only depends on the start times of preceding tasks i, $(i,j)\in E^+$, in this scenario, the respective minimum time lags and r_j. Note that the value of vector y is uniquely determined by r and d. Hence, y is not a decision vector in the ordinary sense, but it rather constitutes an ancillary variable vector that is required to evaluate the expected NPV over all scenarios. (3.3c) ensures satisfaction of the negative-valued precedence relations, while (3.3d) enforces the deadline in all scenarios. Note that (3.3d) cannot be replaced with maximum time lags between events 1 and n since y_1^s is allowed to be strictly positive (by choosing $r_1 > 0$). (3.3e) ensures non-negativity of the solution. We remark that r can be chosen freely as long as the corresponding vector y satisfies all precedence relations in every scenario. Fixing r to the zero vector, for example, entails that all tasks are started as early as possible in every scenario. The relation enforced between r and y as described by (3.3b) is in accordance with our definition of TPT policies (see Section 3.1), and it constitutes a sufficient condition for non-anticipativity of the solution. However, it is not a necessary condition for non-anticipativity, and there might be feasible scheduling policies that result in better solutions.

Model (3.3) employs the expected value as decision criterion, which is in line with the majority of contributions for NPV maximisation under uncertainty. Sometimes, however, cautious decision makers might take a more conservative stance and wish to avoid excessive losses in any particular scenario. Our model can account for individual risk preferences if the task cash flows ζ are replaced with associated utilities. In this case, model (3.3) maximises the expected (discounted) overall utility [Fis70]. Note also that the described model allows both the cash flows and the task durations to depend on the realised scenario. This is desirable as longer task durations typically imply higher task costs, which themselves have a direct impact on the associated cash flows. In accordance with the existing body of literature, we disregard resource constraints and assume a constant discount factor β. Absence of resource restrictions constitutes a compromise that facilitates tractability of the resulting model. Apart from computational considerations, one could justify the absence of resource restrictions by the fact that NPV maximisation models are typically employed in the early stages of the planning process to evaluate the profitability of an investment opportunity. At this stage, resource constraints may be of minor concern and can sometimes be dealt with by managerial intervention (*e.g.*, in the context of project management by acquiring additional resources, shifting holidays or relying on overtime).

A possible variation of our model is to find optimal task delays in the spirit of [Ben06], see Section 3.2. Instead of target processing times, we would then seek for a scenario-independent task delay policy that specifies how much to defer task j beyond the expiry of minimum time lags $(i, j) \in E^+$. The resulting model is neither a special case nor a generalisation of our formulation: for a given problem instance, either model can lead to a superior expected NPV. In view of exact solution procedures, however, such a task delay formulation seems significantly more involved than model (3.3).[3] In Section 3.5.1, we compare TPT policies with task delay policies obtained from the two-stage procedure developed in [Ben06]. Finally, if the durations in model (3.3) do not depend on the realised scenario, then the optimal TPT policy can be

[3]The reason for this becomes clear when we discuss the nodal bounds of our branch-and-bound scheme. While our bounds are determined through linear programs, the bounding problems that arise in the task delay formulation constitute nonconvex problems.

determined by solving a deterministic NPV maximisation problem with averaged cash flows $\zeta_i' := \sum_{s \in S} p_s \zeta_i^s$. This follows from the linearity of the expectation operator.

Let us examine the convexity properties of model (3.3). Leaving the objective function aside for the moment (we could potentially linearise it by using the variable substitution from [Gri72]), only constraint set (3.3b) requires investigation. Its right-hand sides are convex but generically not affine as they constitute maxima of affine functions. Thus, (3.3b) leads to a nonconvex set of feasible solutions. We cannot replace the equalities by greater or equal constraints, however, as otherwise non-anticipativity can be violated in the presence of network tasks with cash outflows. To illustrate this, let us assume that $\zeta_j^s < 0$ for $(j, s) \in V \times S$. Ceteris paribus, it would be beneficial to start task j in scenario s at the latest possible time consistent with all precedence relations, that is, at time $\inf_{k \in V} \left\{ y_k^s - d_{jk}^s : (j, k) \in E \right\}$. As this time can exceed both $\max_{i \in V} \left\{ y_i^s + d_{ij}^s \ : \ (i, j) \in E^+ \right\}$ and r_j, such a decision would anticipate the realised scenario and as such violate causality.

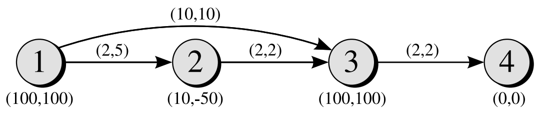


Figure 3.1: Stochastic NPV maximisation problem with two scenarios. The numbers attached to the arcs denote the values of the precedences, while the numbers attached to the nodes represent cash flows. In both cases, the first (second) number refers to the value in scenario 1 (2). Node 4 represents a dummy task that signalises the completion of all network tasks.

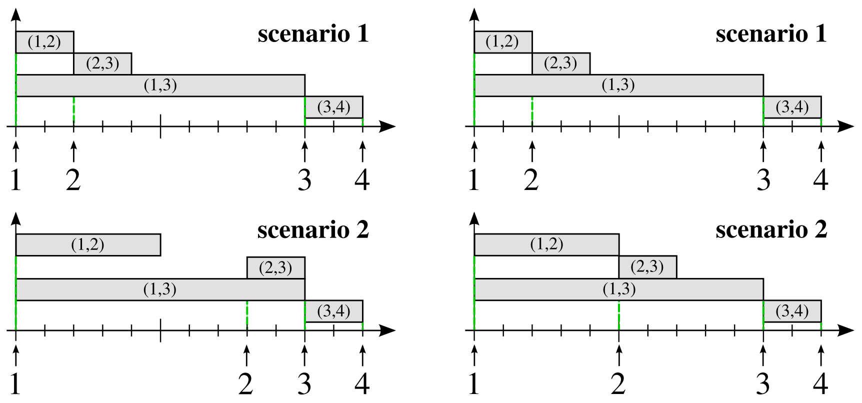


Figure 3.2: Gantt charts for the scenario-wise optimal schedules (left) and the TPT schedule $r = (0, 0, 0, 0)$ (right). The horizontal axis displays the elapsed time, while the vertical axis lists the precedence relations. Arrows indicate the task start times.

The nonconvexity of problem Υ can also be illustrated by the temporal network in Figure 3.1.

For two scenarios, $p = (0.5, 0.5)$, $\Delta = 20$ and any $\beta > 0$, the scenario-wise optimal (*i.e.*, anticipative) solutions y^1 and y^2 are visualised on the left side of Figure 3.2. We see that task 2 is started as early as possible in scenario 1 as it leads to a cash inflow. The same task starts as late as possible in scenario 2, however, since it leads to a cash outflow there. The right part of Figure 3.2 shows the (non-anticipative) schedules stipulated by TPT vector $r = (0, 0, 0, 0)$. Set $\varphi(\lambda) := f((0, \lambda, 0, 0), y_\lambda)$, $\lambda \in [0, 16]$, where y_λ denotes the unique task start time vector that satisfies $((0, \lambda, 0, 0), y_\lambda) \in \mathcal{Y}_\Upsilon$ for a given λ. For a sufficiently large β, φ has zero slope for $\lambda \in [0, 2)$ (changing r_2 has no impact), a negative slope for $\lambda \in (2, 5)$ (the start time of task 2 is postponed in scenario 1), a positive slope for $\lambda \in (5, 8)$ (task 2 is postponed in both scenarios), and finally a negative slope for $\lambda \in (8, 16]$ (tasks 2, 3 and 4 are postponed in both scenarios). Thus, the network's NPV is neither convex nor concave in r.

In view of the solution approach to be proposed, we require the scenario set S to be of small cardinality, that is, it should not contain more than 20–30 elements. While this may be seen as a limitation of our method, we remark that stochastic NPV maximisation problems are known to be challenging [HDD97, HL05]. As an alternative to our approach, one could try to employ a scenario-free uncertainty model. Popular scenario-free approaches to optimisation problems in temporal networks are based on exponentially distributed task durations [BR97, TSS06] or employ a min-max objective [CGS07, CSS07]. We will discuss min-max resource allocation problems in temporal networks in Chapter 5, and Chapter 6 studies temporal networks with Markovian task durations. However, both approaches lead to challenging optimisation problems themselves, the former one due to the curse of dimensionality in dynamic programming and the latter one due to the nonconvexity and two-stage nature of model (3.3). Despite its shortcomings, we thus believe that the proposed approach constitutes a viable tool for the maximisation of a network's NPV. In the future, heuristic solution procedures may help to tackle models with larger scenario sets. Our (exact) solution procedure can be used to assess the performance of such heuristics.

Suitable task duration and cash flow scenarios can be obtained from task-wise estimates or via scenario planning techniques. In the former case, alternative outcomes for the duration and the cash flow of a given task can be determined, for example, by employing three point estimates as in the classical PERT model [MRCF59]. The scenario set S then results from the cross product of all individual task outcomes. Clearly, this approach produces huge scenario sets: even if we assume that every task contributes only three different scenarios, we end up with a scenario set of cardinality $3^{|V|}$. However, scenario reduction techniques may be used to determine a small subset of scenarios that describes the aforementioned cross product as well as possible [HKR09, HR03]. Scenario planning techniques, on the other hand, ask the decision maker to identify the key drivers that affect the durations and cash flows of *all* (or many) tasks. In the context of project management, key drivers could be the weather, commodity prices and future exchange rates. One can then construct an initial set of scenarios by attaching probabilities to the various combinations of possible driver outcomes (*e.g.*, via cross-impact analysis [GH68]). The number of scenarios can subsequently be reduced by clustering techniques. Scenario planning techniques have gained popularity in both theory [KY97, Sch01] and practise [Sch95].

We close this section with an example that illustrates our problem formulation (3.3).

Example 3.3.1 *Consider the temporal network in Figure 3.1, and assume that* $p = (0.5, 0.5)$*, the deadline is* $\Delta = 20$*, and the discount factor is* $\beta = 0.95$*. In this case, model (3.3) becomes*

$$
\begin{aligned}
\underset{r,y}{\text{maximise}} \quad & 1/2\left(100 \cdot 0.95^{y_1^1} + 10 \cdot 0.95^{y_2^1} + 100 \cdot 0.95^{y_3^1}\right) + \\
& 1/2\left(100 \cdot 0.95^{y_1^2} - 50 \cdot 0.95^{y_2^2} + 100 \cdot 0.95^{y_3^2}\right) \\[1em]
\text{subject to} \quad & r \in \mathbb{R}^4, \quad y \in \mathbb{R}^8 \\[1em]
& y_1^1 = r_1, \quad y_2^1 = \max\left\{y_1^1 + 2,\ r_2\right\}, \\
& y_3^1 = \max\left\{y_1^1 + 10,\ y_2^1 + 2,\ r_3\right\}, \quad y_4^1 = \max\left\{y_3^1 + 2,\ r_4\right\}, \\[1em]
& y_1^2 = r_1, \quad y_2^2 = \max\left\{y_1^2 + 5,\ r_2\right\} \\
& y_3^2 = \max\left\{y_1^2 + 10,\ y_2^2 + 2,\ r_3\right\}, \quad y_4^2 = \max\left\{y_3^2 + 2,\ r_4\right\}, \\[1em]
& y_4^1 \leq 20, \quad y_4^2 \leq 20, \\[1em]
& r_1, r_2, r_3, r_4 \geq 0.
\end{aligned}
$$

As we will show later on in Example 3.4.1, this model is optimised by $\widehat{r} = (0, 8, 0, 0)$ *and* $(\widehat{y}^s_1, \widehat{y}^s_2, \widehat{y}^s_3, \widehat{y}^s_4) = (0, 8, 10, 12)$, $s \in \{1, 2\}$. *Thus, the optimal TPT policy assigns a target processing time of* 8 *to task* 2, *while all other tasks should be started as early as possible. The factual task start times for this policy are* 0, 8, 10 *and* 12 *for tasks* 1, 2, 3 *and* 4, *respectively, and they do not depend on the realised scenario. The optimal objective value is* $f(\widehat{r}, \widehat{y}) = 116.96$.

Assume now that in either scenario, task 3 *must be started at most* 7 *time units after task* 1 *has been started. We achieve this by adding a precedence* $(3, 1)$ *to* E *with duration* $(d^1_{31}, d^2_{31}) = (-7, -7)$. *In this case, we would add the constraints*

$$y^1_1 \geq y^1_3 - 7, \quad y^2_1 \geq y^2_3 - 7$$

to the model. The TPT policy $\widehat{r}$ *is no longer feasible under this additional constraint.*

3.4 Solution Procedure

In the following, we develop a branch-and-bound procedure for the solution of model (3.3). Branch-and-bound algorithms solve optimisation problems by implicitly enumerating the set of feasible solutions in a branch-and-bound tree $\mathcal{T}$. Every node of $\mathcal{T}$ represents a subset of the feasible solutions. The tree construction starts at the root node, which represents the entire set of feasible solutions. Branch-and-bound algorithms iteratively select tree nodes $\tau \in \mathcal{T}$ for branching. When a node τ is branched, its set of feasible solutions, $\mathcal{Y}_\tau$, is split into several subsets whose union coincides with $\mathcal{Y}_\tau$. Every subset thus generated represents a 'child' node of τ in $\mathcal{T}$. In principle, nodes of $\mathcal{T}$ can be split until their associated solution sets reduce (or converge) to singletons. In order to avoid such a complete enumeration, one calculates bounds on the optimal objective value achievable at each tree node. A node may then be fathomed as soon as it is guaranteed that it does not contain any better solution than the best one currently known. The crucial components of branch-and-bound procedures are the employed bounds, the branching scheme and the node selection rule, that is, a recipe that specifies which node of $\mathcal{T}$ to split next.

In our branch-and-bound algorithm we determine an upper bound on the optimal objective value achievable at tree node τ by maximising f, the objective function of (3.3), over a relaxation of $\mathcal{Y}_\tau$ that neglects non-anticipativity. This is done by replacing the equalities in (3.3b) by greater or equal relations. As long as the optimal solution (r, y) to such a relaxation contains an y_j^s that satisfies the strict inequality

$$y_j^s > \max \left\{ \sup_{i \in V} \left\{ y_i^s + d_{ij}^s \;:\; (i,j) \in E^+ \right\}, r_j \right\},$$

non-anticipativity is violated (see Sections 2.2.1 and 3.3). In this case, our branching scheme fixes y_j^s to either r_j or $y_i^s + d_{ij}^s$ for one $i \in V$ with $(i,j) \in E^+$, and every such fixation leads to a child node of τ.

We now formalise this idea. The relaxed feasible set $\mathcal{Z}_\Upsilon$ is defined through

$$\left.\begin{array}{ll} y_j^s \geq y_i^s + d_{ij}^s & \forall\, (i,j) \in E,\, s \in S \\ y_j^s \geq r_j & \forall\, j \in V,\, s \in S \\ y_n^s \leq \Delta & \forall\, s \in S \\ r_j \geq 0 & \forall\, j \in V \end{array}\right\} \quad \Leftrightarrow \quad (r, y) \in \mathcal{Z}_\Upsilon. \tag{3.4}$$

Note that $\mathcal{Z}_\Upsilon$ constitutes a convex relaxation of $\mathcal{Y}_\Upsilon$ as defined in (3.3). The requirement that y_j^s has to equal $y_i^s + d_{ij}^s$ for an $i \in V$ with $(i,j) \in E^+$ or r_j will be enforced by restricting (r, y) to one of the hyperplanes

$$\mathcal{Z}_{ij}^s := \begin{cases} \left\{(r,y) \;:\; y_j^s = y_i^s + d_{ij}^s\right\} & \text{for } (i,j) \in E^+,\, s \in S, \\ \left\{(r,y) \;:\; y_j^s = r_j\right\} & \text{for } i = j \in V,\, s \in S, \\ \emptyset & \text{otherwise.} \end{cases} \tag{3.5}$$

We identify a tree node $\tau \in \mathcal{T}$ with the hyperplane restrictions it enforces, that is, $\tau \subseteq V^2 \times S$. For a given node τ, we define the set of feasible solutions, $\mathcal{Y}_\tau$, as well as its relaxation, $\mathcal{Z}_\tau$, as

$$\mathcal{Y}_\tau = \mathcal{Y}_\Upsilon \cap \bigcap_{(i,j,s) \in \tau} \mathcal{Z}_{ij}^s \qquad \text{and} \qquad \mathcal{Z}_\tau = \mathcal{Z}_\Upsilon \cap \bigcap_{(i,j,s) \in \tau} \mathcal{Z}_{ij}^s. \tag{3.6}$$

$\mathcal{Y}_\Upsilon \subseteq \mathcal{Z}_\Upsilon$ implies that $\mathcal{Y}_\tau \subseteq \mathcal{Z}_\tau$, and hence we obtain an upper bound on the achievable objective value at node τ by maximising f over $\mathcal{Z}_\tau$ instead of $\mathcal{Y}_\tau$.

Given a TPT policy $r \in \mathbb{R}^n_+$, we define the *induced schedule* $y(r) \in \mathbb{R}^{nm}$ recursively as follows.

$$y^s_j(r) := \max \left\{ \sup_{i \in V} \left\{ y^s_i(r) + d^s_{ij} \, : \, (i,j) \in E^+ \right\}, r_j \right\}. \tag{3.7}$$

Remember that the precedence relations in E^+ are non-negative valued in all scenarios $s \in S$. If there is a cycle in E^+, then all arcs in the cycle must be associated with zero-valued precedences, for otherwise the network structure is inconsistent (a task cannot be completed before it is started). Thus, the induced schedule is well-defined. Due to the relation between (3.7) and (3.3b), $y^s_j(r)$ equals the factual start time of task j if TPT policy r is implemented and scenario s is realised. By construction, $y(r)$ satisfies all minimum time lags in every scenario. If $y(r)$ also satisfies all maximum time lags in every scenario, that is, if $(r, y(r)) \in \mathcal{Y}_\Upsilon$, then we call r a *feasible policy.*

With this notation, our solution approach can be described as follows.

Algorithm 3.1 Branch-and-bound scheme for model (3.3).

1. **Initialisation.** Set $\mathcal{L} := \{\tau_0\}$ with $\tau_0 = \emptyset$, $r^* := 0$, $f^* := f(r^*, y(r^*))$ if r^* is feasible and $f^* := -M$ otherwise.[4]

2. **Node Selection.** If $\mathcal{L} = \emptyset$ or

$$\max_{\tau \in \mathcal{L}} \sup_{(r,y) \in \mathcal{Z}_\tau} f(r, y) \leq f^*,$$

then go to Step 5. Otherwise, select a node $\tau \in \mathcal{L}$ with

$$\tau \in \arg\max_{\tau \in \mathcal{L}} \sup_{(r,y) \in \mathcal{Z}_\tau} f(r, y)$$

and set $\mathcal{L} := \mathcal{L} \setminus \{\tau\}$.

[4]Here, M denotes a sufficiently large number, for example $M = \sum_{s \in S} \sum_{i \in V} |\zeta^s_i|$.

3. **Bounding.** For the node τ selected in Step 2, let

$$(\widehat{r}, \widehat{y}) := \arg\max \{ f(r, y) \ : \ (r, y) \in \mathcal{Z}_\tau \}$$

be a solution to the upper bound problem at τ. If $\widehat{r}$ is feasible and

$$f(\widehat{r}, y(\widehat{r})) > f^*,$$

where $y(\widehat{r})$ is defined in (3.7), then set $r^* := \widehat{r}$ and $f^* := f(\widehat{r}, y(\widehat{r}))$, that is, a new incumbent TPT policy has been found.[5]

4. **Branching.** Let

$$V_\tau = \left\{ (j, s) \in V \times S \ : \ \widehat{y}_j^s > \max \left\{ \sup_{i \in V} \left\{ \widehat{y}_i^s + d_{ij}^s \ : \ (i, j) \in E^+ \right\}, \widehat{r}_j \right\} \right\}$$

be the set of task-scenario pairs violating non-anticipativity in $(\widehat{r}, \widehat{y})$. If $V_\tau \neq \emptyset$, then select $(j, s) \in V_\tau$ according to some branching scheme and set

$$\mathcal{L} := \mathcal{L} \cup \Bigg(\bigcup_{\substack{i \in V: \\ (i,j) \in E^+}} \{ \tau \cup \{(i, j, s)\} \} \Bigg) \cup \{ \tau \cup \{(j, j, s)\} \}.$$

Go to Step 2 (next iteration).

5. **Termination.** If $f^* \neq -M$, then r^* represents the optimal TPT policy. Otherwise, the problem is infeasible.

Instead of storing the branch-and-bound tree $\mathcal{T}$ explicitly, the algorithm keeps a list $\mathcal{L}$ of nodes that have been constructed by Steps 1 and 4 but not yet selected by Step 2. The relation between $\mathcal{T}$ and $\mathcal{L}$ is that $\tau \in \mathcal{T}$ if and only if $\tau \in \mathcal{L}$ at some point during the execution of the algorithm.

In Step 1, $\mathcal{L}$ only contains the root node $\tau_0 = \emptyset \subseteq V^2 \times S$. Hence, all non-anticipativity

[5] Note that $\widehat{y} \neq y(\widehat{r})$ in general. We will revisit this point later in the text.

constraints in (3.3b) are relaxed in the beginning. r^* denotes the best TPT policy found so far, and f^* denotes its expected NPV. We assign an expected NPV of $-M$ to infeasible policies.

In Step 2, we first check whether we can terminate. This is the case if no more nodes are available or eligible for further processing. Nodes are *available* for further processing if $\mathcal{L}$ is nonempty. Note that after Step 1, $\mathcal{L}$ contains the root node $\tau_0 = \emptyset \subseteq V^2 \times S$ and thus is *not* empty. A node is *eligible* for further processing if its upper bound exceeds f^*, that is, if it can contain a better TPT policy than r^*. In case $\mathcal{L}$ contains eligible nodes, we select a node that attains the maximal upper bound. In the following, we refer to this node as τ.

In Step 3, we calculate an upper bound on the maximal value of f over $\mathcal{Y}_\tau$. This bound is determined by the maximal value of f over the relaxed constraint set $\mathcal{Z}_\tau$ as defined in (3.6), and the corresponding optimal solution is denoted by $(\widehat{r}, \widehat{y})$. In case $\widehat{r}$ constitutes a feasible TPT policy, we also obtain a lower bound $f(\widehat{r}, y(\widehat{r}))$ on the (globally) best TPT policy. We use this lower bound to improve (r^*, f^*) if possible. Note that $\widehat{y} \neq y(\widehat{r})$ in general: $\widehat{y}$ contains the optimal task start times when neglecting some of the non-anticipativity constraints. As such, $\widehat{y}$ typically violates non-anticipativity, that is, the set V_τ defined in Step 4 is usually nonempty. $y(\widehat{r})$, on the other hand, represents the task start times that result from implementing the TPT policy $\widehat{r}$ (see Section 3.1). Although $y(\widehat{r})$ is non-anticipative by construction, it may violate some negative-valued precedences $(i, j) \in E^-$. In model (3.3), the constraint set (3.3b) ensures that feasible solutions $(r, y) \in \mathcal{Y}_\Upsilon$ satisfy $y = y(r)$. In our branch-and-bound algorithm, coincidence of $\widehat{y}$ and $y(\widehat{r})$ is established gradually by adding hyperplane restrictions $\mathcal{Z}_{ij}^s$.

In case the upper bound solution $(\widehat{r}, \widehat{y})$ violates non-anticipativity, we select an anticipating task-scenario pair $(j, s) \in V \times S$ in Step 4 according to some branching scheme. We analysed several branching schemes, including rules based on task start times, numbers of incoming positive-valued precedences and gaps between task start times and their incoming positive-valued precedences. In our experiments, the following strategy performed best: for every anticipative task-scenario pair $(j, s) \in V \times S$, determine the minimum decrease in objective value caused by shifting j to the expiration time of any of its incoming positive-valued precedences or to r_j in scenario s:

$$\eta(j,s) := p_s \left|\zeta_j^s\right| \min \left\{ \inf_{i \in V} \left\{ \beta^{\widehat{y}_i^s + d_{ij}^s} - \beta^{\widehat{y}_j^s} \,:\, (i,j) \in E^+ \right\}, \beta^{\widehat{r}_j} - \beta^{\widehat{y}_j^s} \right\}.$$

$\eta(j,s)$ approximates the minimum additional expected costs of ensuring non-anticipativity for (j,s). We select the anticipative task-scenario pair (j,s) with maximal $\eta(j,s)$. The hope is that this greedy selection rule leads to a fast decrease of the nodal upper bounds. Having selected a pair $(j,s) \in V \times S$, we create one child node for every possible fixation of y_j^s to one of its predecessors $i \in V$, $(i,j) \in E^+$. We also create a child node that fixes y_j^s to r_j. These new child nodes $\tau \cup \{(i,j,s)\}$ and $\tau \cup \{(j,j,s)\}$ are appended to $\mathcal{L}$, and then we go back to Step 2.

After finitely many iterations, $\mathcal{L}$ does not contain any further available or eligible nodes in Step 2. At this point, the algorithm enters Step 5 and delivers either an optimal TPT policy or establishes the infeasibility of (3.3).

The correctness of our branch-and-bound algorithm is proved in two steps. First, we show that the algorithm always terminates after a finite number of iterations. Afterwards, we show that if the algorithm terminates, then it provides the correct result. The proofs of these assertions require some additional notation. We already described the correspondence between the node list $\mathcal{L}$ and the implicitly generated branch-and-bound tree $\mathcal{T}$. For any node $\tau \in \mathcal{T}$ we denote the set of direct descendants by $D_{\mathcal{T}}(\tau)$. Thus, we have that $\tau' \in D_{\mathcal{T}}(\tau)$ if and only if τ' is added to $\mathcal{L}$ as a result of branching τ in Step 4 of our algorithm. We denote the set of all (transitive) descendants of τ in $\mathcal{T}$ by $D_{\mathcal{T}}^*(\tau)$, that is, $D_{\mathcal{T}}^*(\tau)$ contains all direct descendants of τ, all direct descendant of τ's direct descendants, etc. Similarly, $A_{\mathcal{T}}(\tau)$ denotes the set of direct ancestors of τ (a singleton). We have $\tau \in A_{\mathcal{T}}(\tau')$ if and only if $\tau' \in D_{\mathcal{T}}(\tau)$. Finally, $A_{\mathcal{T}}^*(\tau)$ refers to all (transitive) ancestors of τ in $\mathcal{T}$.

We now prove finite termination and completeness of our algorithm.

Theorem 3.4.1 (Termination) *For any given problem instance, the algorithm terminates after finitely many iterations.*

Proof We show that the generated branch-and-bound tree $\mathcal{T}$ is finite. By the correspondence between $\mathcal{L}$ and $\mathcal{T}$, the claim then follows immediately. For every $\tau \in \mathcal{T}$, $|D_{\mathcal{T}}(\tau)|$ is bounded by $|V|$, because an anticipating task-scenario pair $(j, s) \in V \times S$ can only be fixed to either one of its preceding tasks $i \in V$ with $(i, j) \in E^+$ or to r_j. If $(j, s) \in V_\tau$ is branched upon, then $(j, s) \notin V_{\tau'}$ for any transitive descendant $\tau' \in D^*_{\mathcal{T}}(\tau)$, because either $(i, j, s) \in \tau'$ for $i \in V$ with $(i, j) \in E^+$ or $(j, j, s) \in \tau'$. As a result, no node $\tau \in \mathcal{T}$ can possess more than nm fixations. Hence, both the number of levels in $\mathcal{T}$ and the fan-out within each level are bounded, which proves finiteness of $\mathcal{T}$. ∎

Theorem 3.4.2 (Completeness) *The algorithm returns $f^* = -M$ if the problem is infeasible and a TPT policy r^* with*

$$(r^*, y(r^*)) \in \arg\max \{f(r, y) \, : \, (r, y) \in \mathcal{Y}_\Upsilon\}$$

otherwise.

Proof We first show that the algorithm correctly identifies infeasible instances. The algorithm classifies a problem as infeasible if $f^* = -M$ after termination. Note that f^* can only change in Step 3. For this to happen, however, r^* needs to be feasible, implying that a feasible solution in $\mathcal{Y}_\tau$ (and, a fortiori, in $\mathcal{Y}_\Upsilon$) has been found. Thus, for any infeasible instance, our algorithm returns $f^* = -M$, that is, it correctly recognises the problem's infeasibility.

If instance Υ is feasible, then it has an optimal solution: the existence of a finite deadline, together with the assumption of a unique source in the subgraph (V, E^+), ensures that $\mathcal{Y}_\Upsilon$ is compact. The continuous function (3.3a) thus attains its maximum over $\mathcal{Y}_\Upsilon$ due to the Weierstrass maximum theorem.

We now prove that if the problem is feasible, then the algorithm finds an optimal solution. To show this, let r^{opt} be an optimal TPT policy. We examine the branch-and-bound tree $\mathcal{T}$ generated by our procedure. Let $\mathcal{T}' := \{\tau \in \mathcal{T} \, : \, (r^{\text{opt}}, y(r^{\text{opt}})) \in \mathcal{Y}_\tau\}$, that is, $\mathcal{T}'$ consists of all the tree nodes of $\mathcal{T}$ that contain the optimal solution $(r^{\text{opt}}, y(r^{\text{opt}}))$. Note that $\mathcal{T}' \neq \emptyset$

since it contains at least the root node of $\mathcal{T}$. We now remove from $\mathcal{T}'$ all nodes that have descendants in $\mathcal{T}'$, that is, $\mathcal{T}'' := \mathcal{T}' \setminus \bigcup_{\tau \in \mathcal{T}'} A^*_{\mathcal{T}'}(\tau)$. By construction, $\mathcal{T}'' \neq \emptyset$ holds as well. Let us fix an arbitrary $\tau \in \mathcal{T}''$. During the execution of our algorithm, τ has either been selected in Step 2 or not. If it has never been selected, then the inequality $f(r^{\text{opt}}, y(r^{\text{opt}})) \leq \max\{f(r,y) : (r,y) \in \mathcal{Z}_\tau\} \leq f^*$ must hold at the end of the algorithm, implying that a TPT policy at least as good as r^{opt} has been found. In case τ has been selected in Step 2 at some point, we know by definition of $\mathcal{T}''$ that τ has not been branched. This is only possible if $V_\tau = \emptyset$, that is, if $(\widehat{r}, \widehat{y}) \in \arg\max\{f(r,y) : (r,y) \in \mathcal{Z}_\tau\}$ is non-anticipative, where $(\widehat{r}, \widehat{y})$ denotes the upper bound for τ determined in Step 3. In that case, however, r^* has been updated to $\widehat{r}$ in Step 3 (if necessary) and thus, a TPT policy at least as good as r^{opt} has been identified by our method. ∎

In our algorithm description, we did not consider any dominance rules. In fact, the dominance rules that prevail in the literature (see [BDM$^+$99, DH02]) are based on partial schedules. The tree nodes of $\mathcal{T}$, on the other hand, represent sets of complete schedules which may not yet be feasible due to their anticipativity. As a result, classical dominance rules such as the 'superset-subset' rule [DH02] are not (directly) applicable. Whether other dominance rules can be used beneficially to enhance our algorithm remains an area for further research.

Step 3 of our branch-and-bound procedure requires the efficient solution of

$$\max\{f(r,y) : (r,y) \in \mathcal{Z}_\tau\} \qquad (\Upsilon(\tau))$$

for nodes $\tau \in \mathcal{T}$, where $\mathcal{Z}_\tau$ results from the intersection of $\mathcal{Z}_\Upsilon$ with the hyperplanes indexed by τ, see (3.4)–(3.6). In the following, we refer to this problem as $\Upsilon(\tau)$.

$\Upsilon(\tau)$ is equivalent to the following optimisation problem:

$$\begin{array}{lll}
\underset{r,y}{\text{maximise}} & \displaystyle\sum_{s \in S} p_s \sum_{i \in V} \zeta_i^s \beta^{y_i^s} & \\
\text{subject to} & r \in \mathbb{R}^n, \quad y \in \mathbb{R}^{nm} & \\
& y_j^s \begin{cases} = y_i^s + d_{ij}^s & \text{if } (i,j,s) \in \tau \\ \geq y_i^s + d_{ij}^s & \text{otherwise} \end{cases} & \forall\, (i,j) \in E,\ s \in S, \\
& y_j^s \begin{cases} = r_j & \text{if } (j,j,s) \in \tau \\ \geq r_j & \text{otherwise} \end{cases} & \forall\, j \in V,\ s \in S, \\
& y_n^s \leq \Delta \qquad \forall\, s \in S, & \\
& r_j \geq 0 \qquad \forall\, j \in V. &
\end{array}$$

In analogy to the deterministic model (3.1), we could employ the substitutions $t_j := \beta^{r_j}$, $j \in V$, and $z_j^s := \beta^{y_j^s}$, $j \in V$ and $s \in S$, to transform this problem into an equivalent linear program. As we will show in Section 3.4.1, however, $\Upsilon(\tau)$ can also be reformulated as a deterministic NPV maximisation problem. The latter approach improves the performance of our branch-and-bound procedure since the specialised algorithms reviewed in Section 3.2 outperform linear programming solvers by several orders of magnitude [SZ01]. In Section 3.4.2 we discuss how to exploit information from the father node in the branch-and-bound tree when solving $\Upsilon(\tau)$. This allows us to further speed up the calculation of nodal upper bounds as the algorithms reviewed in Section 3.2 require significantly fewer iterations when warm-started from near-optimal solutions.

We close this section with an illustration of our branch-and-bound procedure.

Example 3.4.1 *Consider again the temporal network in Figure 3.1 with scenario probabilities $p = (0.5, 0.5)$, deadline $\Delta = 20$ and discount factor $\beta = 0.95$. For this problem instance, the branch-and-bound algorithm proceeds as follows.*

We start with Step 1, where we set $\mathcal{L} := \{\tau_0\}$ with $\tau_0 = \emptyset$ and $r^* = 0$. The induced schedule

$y(r^*)$ is the early start schedule visualised in Figure 3.2 (right). Since there are no maximum time lags, this schedule is feasible and leads to an objective value of

$$1/2\left(100 \cdot 0.95^0 + 10 \cdot 0.95^2 + 100 \cdot 0.95^{10}\right) + 1/2\left(100 \cdot 0.95^0 - 50 \cdot 0.95^5 + 100 \cdot 0.95^{10}\right) \approx 115.40.$$

Hence, we set $f^* = 115.40$.

The node $\tau = \tau_0 \in \mathcal{L}$ does not enforce any fixations yet. Hence, the problem $\sup_{(r,y)\in\mathcal{Z}_\tau} f(r, y)$ in Step 2 is maximised by $\widehat{r} = 0$ and the anticipative task start time vector $\widehat{y}$ visualised in Figure 3.2 (left). The objective value is

$$1/2\left(100 \cdot 0.95^0 + 10 \cdot 0.95^2 + 100 \cdot 0.95^{10}\right) + 1/2\left(100 \cdot 0.95^0 - 50 \cdot 0.95^8 + 100 \cdot 0.95^{10}\right) \approx 118.16.$$

Since this value exceeds $f^* = 115.40$, we remove node τ_0 from $\mathcal{L}$ and continue.

In Step 3 we check whether the objective value of the induced schedule $y(\widehat{r})$ exceeds the objective value of the induced schedule $y(r^*)$. Since $\widehat{r}$ and r^* are identical, this is not the case.

In Step 4 we identify the set of anticipative task-scenario pairs for $\tau = \tau_0$ as $V_\tau = \{(2, 2)\}$. We create two new nodes $\tau_1 := \{(1, 2, 2)\}$ ('start task 2 in scenario 2 immediately d^2_{12} time units after task 1 has been started') and $\tau_2 := \{(2, 2, 2)\}$ ('start task 2 in scenario 2 at time r_2') and add them to $\mathcal{L}$.

We are back in Step 2 with $\mathcal{L} = \{\tau_1, \tau_2\}$. For node $\tau = \tau_1$, the problem $\sup_{(r,y)\in\mathcal{Z}_\tau} f(r, y)$ is maximised by $\widehat{r} = 0$ and the early start schedule $\widehat{y}$ shown in Figure 3.2 (right). We have already seen that the associated objective value is $115.40 \leq f^* = 115.40$. For node $\tau = \tau_2$, the problem $\sup_{(r,y)\in\mathcal{Z}_\tau} f(r, y)$ is maximised by $\widehat{r} = (0, 8, 0, 0)$ and the scenario-independent task start times $(\widehat{y}^s_1, \widehat{y}^s_2, \widehat{y}^s_3, \widehat{y}^s_4) = (0, 8, 10, 12)$, $s \in \{1, 2\}$. The objective value of this solution is

$$1/2\left(100 \cdot 0.95^0 + 10 \cdot 0.95^8 + 100 \cdot 0.95^{10}\right) + 1/2\left(100 \cdot 0.95^0 - 50 \cdot 0.95^8 + 100 \cdot 0.95^{10}\right) \approx 116.96.$$

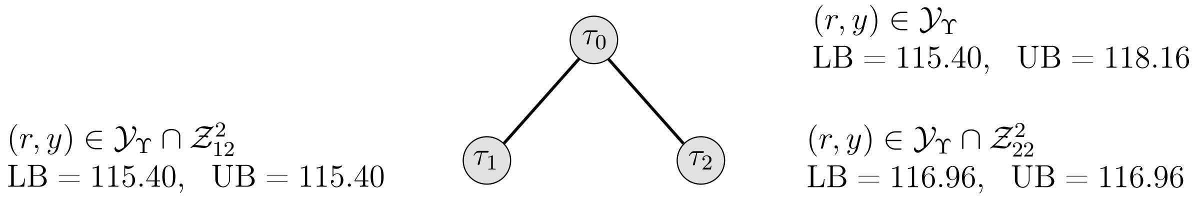
Since $116.96 > f^* = 115.40$, we remove τ_2 from $\mathcal{L}$ and continue.

For node $\tau = \tau_2$*, we have* $y(\widehat{r}) = \widehat{y}$*. Hence,* $f(\widehat{r}, y(\widehat{r})) = 116.96$*, and we update* f^* *and* r^* *in* *Step 3* *to* $f^* = 116.96$ *and* $r^* = (0, 8, 0, 0)$*.*

Since the solution $(\widehat{r}, \widehat{y})$ *is non-anticipative, we have* $V_\tau = \emptyset$ *in* *Step 4*. *We do not branch* τ_2*.*

Back in *Step 2*, *the list* $\mathcal{L}$ *only contains node* τ_1*. Since the objective value of* $\sup_{(r,y)\in\mathcal{Z}_\tau} f(r, y)$ *for* $\tau = \tau_1$ *is* 115.40 *and* $115.40 \leq f^* = 116.96$*, no eligible nodes are left for branching.*

We therefore go to *Step 5* *and return the optimal TPT policy* $r^* = (0, 8, 0, 0)$*. Figure 3.3 visualises the branch-and-bound tree that we generated during the execution of our algorithm.*



$(r, y) \in \mathcal{Y}_\Upsilon$
LB = 115.40, UB = 118.16
τ_0
τ_1
τ_2
$(r, y) \in \mathcal{Y}_\Upsilon \cap \mathcal{Z}^2_{12}$
LB = 115.40, UB = 115.40
$(r, y) \in \mathcal{Y}_\Upsilon \cap \mathcal{Z}^2_{22}$
LB = 116.96, UB = 116.96

Figure 3.3: Branch-and-bound tree generated in Example 3.4.1. We denote the nodal upper and lower bounds by 'UB' and 'LB', respectively. Node τ_1 is fathomed because its upper bound does not exceed the objective value of the incumbent solution, whereas τ_2 is fathomed because its upper bound is attained by a feasible (*i.e.*, non-anticipative) solution.

3.4.1 Efficient Nodal Bounds

Given a stochastic NPV maximisation instance $\Upsilon = (G, S, p, \zeta, d, \Delta, \beta)$ and a tree node $\tau \in \mathcal{T}$, we define an instance $\Gamma(\tau) = (\widetilde{G}, \widetilde{\zeta}, \widetilde{d}, \beta)$ of the deterministic NPV maximisation problem (3.2) as follows. $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ represents a network whose node set $\widetilde{V} = \{0, \ldots, \widetilde{n}\}$ consists of three categories. The first category encompasses the artificial start node 0, which provides a unique source for the network. The second category consists of TPT nodes $i = 1, \ldots, n$, which correspond to the target processing times r_i, $i \in V$. The last category encompasses task-scenario nodes $p = sn + i$ for $i \in V$ and $s \in S$, which represent the task start times in the different scenarios. For a network with two scenarios, for example, nodes $n+1, \ldots, 2n$ describe the start times of tasks $1, \ldots, n$ in scenario 1, respectively, and nodes $2n+1, \ldots, 3n = \widetilde{n}$ describe the corresponding start times in scenario 2. We assign a cash flow of magnitude $p_s\zeta^s_i$ to task-scenario node $p = sn + i$, $i \in V$ and $s \in S$, while the other nodes in $\widetilde{V}$ do not give rise to cash flows. Thus, the node-related data of $\Gamma(\tau)$ can be summarised as follows:

$$\widetilde{V} := \{0, \dots, \widetilde{n}\} \quad \text{with} \quad \widetilde{n} = n(m+1), \tag{3.8}$$

$$\widetilde{\zeta}_p := \begin{cases} p_s \zeta_i^s & \text{if } p = sn + i,\ i \in V \text{ and } s \in S, \\ 0 & \text{otherwise.} \end{cases} \tag{3.9}$$

We now construct the precedences $\widetilde{E}$ of $\Gamma(\tau)$. We establish zero-valued precedences between the artificial start node 0 and all TPT nodes $i = 1, \dots, n$ to ensure non-negativity of the solution. Next, we add zero-valued precedences between the TPT nodes $i = 1, \dots, n$ and the corresponding task-scenario nodes $p = sn + i$, $s \in S$. This guarantees that tasks are not started before their TPTs. For every precedence $(i, j) \in E$, we add to $\Gamma(\tau)$ a precedence of value d_{ij}^s between $p = sn + i$ and $q = sn + j$ for every scenario $s \in S$. This ensures that $\Gamma(\tau)$ obeys the original precedence relations of Υ in all scenarios. Fixations $(i, j, s) \in V^2 \times S$ are modelled as tight maximum time lags between the respective nodes in $\widetilde{V}$. Satisfaction of the deadline Δ, finally, is ensured by adding maximum time lags of duration Δ between 0 and $sn + n$ for all scenarios $s \in S$.

Summing up, $\widetilde{E}$ and $\widetilde{d}$ are defined as follows.

$$\begin{aligned} \widetilde{E} := {} & \{(0, i) \,:\, i = 1, \dots, n\} \cup \{(i, sn + i) \,:\, i \in V,\ s \in S\} \\ & \cup \{(sn + i, i) \,:\, (i, i, s) \in \tau\} \cup \{(sn + i, sn + j) \,:\, (i, j) \in E,\ s \in S\} \\ & \cup \{(sn + j, sn + i) \,:\, (i, j, s) \in \tau,\ i \neq j\} \cup \{(sn + n, 0) \,:\, s \in S\}, \end{aligned} \tag{3.10}$$

$$\widetilde{d}_{pq} := \begin{cases} d_{ij}^s & \text{if } p = sn + i,\ q = sn + j,\ (i, j) \in E \text{ and } (j, i, s) \notin \tau, \\ -d_{ji}^s & \text{if } p = sn + i,\ q = sn + j,\ (j, i) \in E \text{ and } (j, i, s) \in \tau, \\ -\Delta & \text{if } p = sn + n,\ q = 0 \text{ and } s \in S, \\ 0 & \text{otherwise.} \end{cases} \tag{3.11}$$

Note that in case both an ordinary precedence (inherited from Υ) and a fixation exist between two nodes in $\widetilde{V}$, the latter constraint must be more restrictive. Hence, the definition of $\widetilde{d}$ ignores the precedence from Υ in such cases.

Example 3.4.2 *The construction of* $\Gamma(\tau)$ *for a small example network and* $\tau_0 = \emptyset$ *(the root node of* $\mathcal{T}$*) is shown in Figure 3.4, while the fixation process is illustrated in Figure 3.5.*

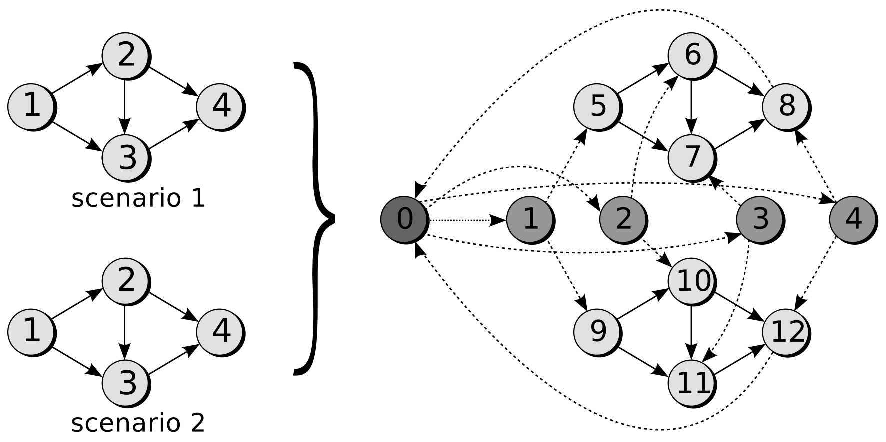


Figure 3.4: For the stochastic NPV maximisation instance in the left chart (only the network structure is shown), the deterministic NPV maximisation problem $\Gamma(\tau_0)$ with $\tau_0 = \emptyset$ is visualised on the right side. 0 represents the artificial source, $1, \ldots, 4$ the TPT nodes and $5, \ldots, 12$ the task-scenario nodes. The precedences $(8, 0)$ and $(12, 0)$ enforce the deadline Δ.

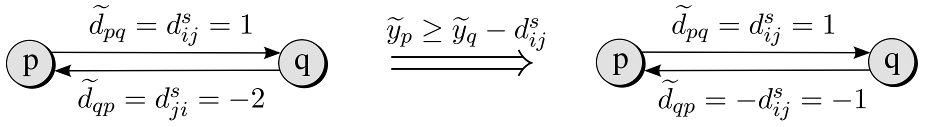

$\widetilde{d}_{pq} = d^s_{ij} = 1$
$\widetilde{d}_{qp} = d^s_{ji} = -2$
$\widetilde{y}_p \geq \widetilde{y}_q - d^s_{ij}$
$\widetilde{d}_{pq} = d^s_{ij} = 1$
$\widetilde{d}_{qp} = -d^s_{ij} = -1$

Figure 3.5: For $i, j \in V$ with a minimum time lag $(i, j) \in E^+$ of duration 1 and a maximum time lag $(j, i) \in E^-$ of duration 2 in scenario $s \in S$, the left chart visualises the corresponding subgraph of $\widetilde{G}$ with $p = sn + i$ and $q = sn + j$. Conducting the fixation (i, j, s) replaces the value of precedence $(q, p) \in \widetilde{E}$ as shown on the right side.

The following theorem establishes the link between $\Upsilon(\tau)$ and $\Gamma(\tau)$.

Theorem 3.4.3 *Consider a problem instance* $\Upsilon = (G, S, p, \zeta, d, \Delta, \beta)$*, a tree node* $\tau \in \mathcal{T}$ *and the deterministic NPV maximisation problem* $\Gamma(\tau) = (\widetilde{G}, \widetilde{\zeta}, \widetilde{d}, \beta)$ *as defined in (3.8)–(3.11). Let* $\widetilde{y}$ *be an optimal solution to* $\Gamma(\tau)$*, where* $\widetilde{y}_p$ *denotes the start time of task* $p \in \widetilde{V}$*. Then*

$$(\widetilde{y}_1, \ldots, \widetilde{y}_{\widetilde{n}}) \in \arg\max \{ f(r, y) \; : \; (r, y) \in \mathcal{Y}_\tau \} .$$

Proof Under the natural identification

$$((\widetilde{y}_1, \ldots, \widetilde{y}_n), (\widetilde{y}_{n+1}, \ldots, \widetilde{y}_{\widetilde{n}})) \quad = \quad ((r_1, \ldots, r_n), (y_1, \ldots, y_{nm})) \quad = \quad (r, y),$$

the feasible sets of $\Gamma(\tau)$ and $\Upsilon(\tau)$ coincide. Furthermore, the objective value of $\widetilde{y}$ in $\Gamma(\tau)$ equals the objective value of (r, y) in $\Upsilon(\tau)$. ∎

Example 3.4.3 *Consider the branch-and-bound node τ_1 that is generated in the solution of the stochastic NPV maximisation problem in Example 3.4.1. Figure 3.6 visualises the deterministic NPV maximisation problem $\Gamma(\tau_1)$ for this node.*

The desired instance $\Gamma(\tau_1)$ of the deterministic NPV maximisation problem (3.2) is

$$\begin{aligned}
\underset{y}{\text{maximise}} \quad & 50 \cdot 0.95^{y_5} + 5 \cdot 0.95^{y_6} + 50 \cdot 0.95^{y_7} + \\
& 50 \cdot 0.95^{y_9} - 25 \cdot 0.95^{y_{10}} + 50 \cdot 0.95^{y_{11}} \\
\text{subject to} \quad & y \in \mathbb{R}^{13} \\
& y_1 \geq y_0, \quad y_2 \geq y_0, \quad y_3 \geq y_0, \quad y_4 \geq y_0, \\
& y_5 \geq y_1, \quad y_9 \geq y_1, \quad y_6 \geq y_2, \quad y_{10} \geq y_2, \\
& y_7 \geq y_3, \quad y_{11} \geq y_3, \quad y_8 \geq y_4, \quad y_{12} \geq y_4, \\
& y_6 \geq y_5 + 2, \quad y_7 \geq y_5 + 10, \quad y_7 \geq y_6 + 2, \quad y_8 \geq y_7 + 2, \\
& y_{10} \geq y_9 + 5, \quad y_{11} \geq y_9 + 10, \quad y_{11} \geq y_{10} + 2, \quad y_{12} \geq y_{11} + 2, \\
& y_9 \geq y_{10} - 5, \\
& y_0 \geq y_8 - 20, \quad y_0 \geq y_{12} - 20, \\
& y_0 = 0.
\end{aligned}$$

The optimal solution to this problem is $(y_0, \ldots, y_{12}) = (0, 0, 0, 0, 0, 0, 2, 10, 12, 0, 5, 10, 12)$ *with objective value* 115.40*, see Example 3.4.1.*

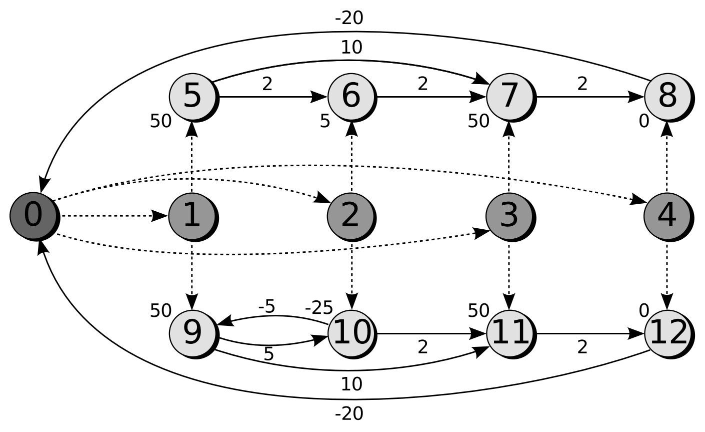


Figure 3.6: Deterministic NPV maximisation problem $\Gamma(\tau_1)$ associated with the branch-and-bound node τ_2 in Example 3.4.1. Node 0 represents the artificial source, $1, \ldots, 4$ the TPT nodes and $5, \ldots, 12$ the task-scenario nodes. A dotted arc (p, q) refers to a precedence with duration $d_{pq} = 0$. Nodes $0, \ldots, 4$ have zero cash flows (not shown).

3.4.2 Warm-Start Technique

A non-root node τ' differs from its ancestor $\tau \in A_{\mathcal{T}}(\tau')$ by exactly one fixation. Hence, an optimal solution to $\Upsilon(\tau)$ is likely to be very similar to an optimal solution to $\Upsilon(\tau')$. This property carries over to the optimal solutions to the deterministic NPV maximisation problems $\Gamma(\tau)$ and $\Gamma(\tau')$. This similarity of nodal solutions is typical for branch-and-bound algorithms and is exploited by warm-start techniques. In our context, this means that at node τ' we should warm-start the algorithm developed in [NZ00] (hereafter referred to as NZ) with an optimal solution to $\Gamma(\tau)$. The hope is that NZ requires significantly fewer iterations than if we apply it to a standard initial solution.

Let us elaborate this idea. Assume that $\tau' \setminus \tau = \{(i, j, s)\}$ for $(i, j) \in E^+$; the case $\tau' \setminus \tau = \{(j, j, s)\}$ for $j \in V$ is analogous. The precedence that relates i and j is more constraining in $\Gamma(\tau')$ than it is in $\Gamma(\tau)$. The modified precedence is not fulfilled by the optimal solution found for $\Gamma(\tau)$, for otherwise $(i, j, s) \notin V_\tau$ in Step 4 of our branch-and-bound algorithm and hence $(i, j, s) \notin \tau' \setminus \tau$. Since NZ is a variant of the network simplex algorithm, it has to be started from a (primal) feasible solution. If we enforced the fixation $(i, j, s) \in \tau' \setminus \tau$ as a hard constraint, then we would need to specify a feasible initial solution to $\Gamma(\tau')$. Instead, we incorporate it implicitly by penalising its violation in the objective function:

$$g_\pi(\widetilde{y}) := g(\widetilde{y}) + \pi(\beta^{\widetilde{y}_{sn+j}} - \beta^{\widetilde{y}_{sn+i}+d^s_{ij}}).$$

Here, g denotes the objective function of problem $\Gamma(\tau)$, while g_π constitutes the penalised function for some penalty factor $\pi > 0$. Note that $\widetilde{y}_{sn+j} \geq \widetilde{y}_{sn+i} + d^s_{ij}$ holds for all feasible solutions to $\Gamma(\tau)$ and $\Gamma(\tau')$ because the respective minimum time lag is enforced in both problems. Hence, $g_\pi(\widetilde{y}) \leq g(\widetilde{y})$ for all feasible solutions $\widetilde{y}$, and $g_\pi(\widetilde{y}) = g(\widetilde{y})$ if and only if $\widetilde{y}_{sn+j} = \widetilde{y}_{sn+i} + d^s_{ij}$, that is, if $\widetilde{y}$ obeys the new fixation $(i, j, s) \in \tau' \setminus \tau$. Note also that the penalised objective function g_π can be obtained from g by merely modifying two cash flows in $\Gamma(\tau)$:

$$\widetilde{\zeta}_{\pi,p} := \begin{cases} \widetilde{\zeta}_p - \pi\beta^{d^s_{ij}} & \text{if } p = sn + i, \\ \widetilde{\zeta}_p + \pi & \text{if } p = sn + j, \\ \widetilde{\zeta}_p & \text{otherwise.} \end{cases} \tag{3.12}$$

Hence, we can solve the penalty formulation by applying NZ to the slightly modified problem instance $\Gamma_\pi(\tau) = (\widetilde{G}, \widetilde{\zeta}_\pi, \widetilde{d}, \beta)$. The following theorem shows how this penalty formulation relates to $\Gamma(\tau')$:

Theorem 3.4.4 *Consider a problem instance $\Upsilon = (G, S, p, \zeta, d, \Delta, \beta)$ and $\tau, \tau' \subseteq V^2 \times S$, where $\mathcal{Z}_\tau \neq \emptyset$ and $\tau' \setminus \tau = \{(i, j, s)\}$, $(i, j) \in E^+$. Moreover, let $\Gamma_\pi(\tau) = (\widetilde{G}, \widetilde{\zeta}_\pi, \widetilde{d}, \beta)$ be the modified deterministic NPV maximisation problem that penalises the violation of $(i, j, s) \in V_\tau$ in the branch-and-bound node τ. There exists a $\pi_0 \geq 0$ such that for all $\pi \geq \pi_0$, the optimal solution $\widetilde{y}$ to $\Gamma_\pi(\tau)$ found by NZ satisfies*

(i) $\widetilde{y}_{sn+j} = \widetilde{y}_{sn+i} + d^s_{ij} \iff \widetilde{y} \in \arg\max \Gamma(\tau')$;

(ii) $\widetilde{y}_{sn+j} \neq \widetilde{y}_{sn+i} + d^s_{ij} \iff \Gamma(\tau')$ *is infeasible.*

Proof We can assume that $\mathcal{Z}_\Upsilon \cap \mathcal{Z}^s_{ij} \neq \emptyset$ since otherwise the assertion trivially holds for any $\pi_0 \geq 0$. Grinold's variable substitution transforms $\Gamma(\emptyset)$ to an equivalent LP. Being a derivate of the network simplex algorithm, NZ always terminates at a vertex of this LP, which in turn corresponds to a vertex of $\mathcal{Z}_\Upsilon$ [NZ00]. Moreover, by the nature of the hyperplane fixations, the application of NZ to $\Gamma(\tau)$ (and hence $\Gamma_\pi(\tau)$) always terminates in a vertex of $\mathcal{Z}_\Upsilon$, too. Let $\mathcal{V}$ be the finite set containing all vertices of $\mathcal{Z}_\Upsilon$ that do *not* lie on the hyperplane $\mathcal{Z}^s_{ij}$. We can assume $\mathcal{V} \neq \emptyset$ since otherwise the assertion is trivially satisfied for any $\pi_0 \geq 0$. For every $v \in \mathcal{V}$, we can determine a finite π_v such that for all $\pi \geq \pi_v$,

$$g_\pi(v) < \min\left\{g_\pi(\widetilde{z}) \,:\, \widetilde{z} \in \mathcal{Z}_\Upsilon \cap \mathcal{Z}^s_{ij}\right\}.$$

Existence of π_v follows from the fact that $\mathcal{Z}_\Upsilon$ is compact and g is bounded.

Set $\pi_0 = \max_{v\in\mathcal{V}} \pi_v$ and choose any $\pi \geq \pi_0$. If the optimal solution $\widetilde{y}$ to $\Gamma_\pi(\tau)$ lies on the hyperplane $\mathcal{Z}^s_{ij}$, then it is optimal among all elements of $\mathcal{Z}_\tau \cap \mathcal{Z}^s_{ij}$. Since g_π coincides with g on $\mathcal{Z}^s_{ij}$, $\widetilde{y}$ must then be optimal for $\Gamma(\tau')$. If, on the other hand, $\widetilde{y} \notin \mathcal{Z}^s_{ij}$, then the choice of π implies that $\mathcal{Z}_\tau \cap \mathcal{Z}^s_{ij} = \emptyset$, which is equivalent to $\mathcal{Z}_{\tau'} = \emptyset$. The reverse implications, finally, hold by definition. ∎

In practise, we do not need to choose π explicitly to solve $\Gamma_\pi(\tau)$. Indeed, if we employ NZ to solve $\Gamma_\pi(\tau)$, then the values of all cash flows and dual variables in the algorithm description (see [NZ00]) are of the form $\pi a + b$ for $a, b \in \mathbb{R}$. Hence, we can employ a variant of NZ that operates on tuples of cash flows and dual variables, where tuple (a, b) corresponds to the value $\pi a + b$ for some undefined but sufficiently large π. The algorithm description from [NZ00] remains valid, the only differences being that (i) operations on cash flows and dual variables are performed entry-wise and (ii) the variable that leaves the dual basis is chosen in lexicographic order. This is reminiscent of the Big-M method in linear programming [Tah97].

Once we have obtained an optimal solution to $\Gamma_\pi(\tau)$, we can either discard tree node τ' (in case infeasibility has been detected) or update the time lag $\widetilde{d}_{pq}$, $p = sn + j$ and $q = sn + i$, indexed by $\tau' \setminus \tau = \{(i, j, s)\}$. This allows us to use the optimal solution to $\Gamma_\pi(\tau)$ not just for the upper bound of node τ', but also as an initial solution to $\tau'' \in D_\mathcal{T}(\tau')$. The imposition of a tight maximum time lag between p and q entails that we do not require the introduced penalty terms anymore but can rather reuse the original cash flows ζ in subsequent iterations of the branch-and-bound procedure.

We close this section with an example of the warm-start procedure.

Example 3.4.4 *Consider the solution to the deterministic NPV maximisation problem* $\Gamma(\tau_1)$ *associated with the branch-and-bound node* τ_1 *in Example 3.4.1. In comparison to the problem* $\Gamma(\tau_0)$ *associated with the root node of the branch-and-bound tree,* $\Gamma(\tau_1)$ *contains an additional precedence* $(10, 9)$ *of value* $\widetilde{d}_{10,9} = -5$*, see Figure 3.6. The optimal solution* $\widetilde{y}$ *to* $\Gamma(\tau_0)$ *satisfies* $\widetilde{y}_9 = 0$ *and* $\widetilde{y}_{10} = 8$ *and therefore violates the new precedence constraint* $\widetilde{y}_9 \geq \widetilde{y}_{10} - 5$ *contained in* $\Gamma(\tau_1)$*. According to Theorem 3.4.4, we can enforce this new constraint by changing the cash flows associated with tasks* 9 *and* 10 *to*

$$\widetilde{\zeta}_{\pi,9} = 50 - 0.95^{-5}\pi \qquad \text{and} \qquad \widetilde{\zeta}_{\pi,10} = -25 + \pi.$$

For $\pi = 1{,}000$*, for example, we obtain* $\widetilde{\zeta}_{\pi,9} = -723.78$ *and* $\widetilde{\zeta}_{\pi,10} = 975$*. This choice of cash flows guarantees that* $\widetilde{y}_{10} = \widetilde{y}_9 + 5$ *in any optimal solution, that is, task* 10 *will be started exactly 5 time units after task 9 has been started. Note that the combined cash flow of tasks* 9 *and* 10 *evaluates to*

$$\widetilde{\zeta}_{\pi,9}\,\beta^{\widetilde{y}_9} + \widetilde{\zeta}_{\pi,10}\,\beta^{\widetilde{y}_{10}} \quad = \quad -723.78 \cdot 0.95^{\widetilde{y}_9} + 975 \cdot 0.95^{\widetilde{y}_9+5} \quad = \quad 30.66 \cdot 0.95^{\widetilde{y}_9}.$$

This cash flow is identical to the original combined cash flow of tasks 9 *and* 10 *if they are started in immediate succession:*

$$\widetilde{\zeta}_9\,\beta^{\widetilde{y}_9} + \widetilde{\zeta}_{10}\,\beta^{\widetilde{y}_{10}} \quad = \quad 50 \cdot 0.95^{\widetilde{y}_9} - 25 \cdot 0.95^{\widetilde{y}_9+5} \quad = \quad 30.66 \cdot 0.95^{\widetilde{y}_9}$$

Hence, the changes in $\widetilde{\zeta}_\pi$ *do not influence the task start times beyond the desired fixation.*

3.5 Numerical Results

In the first part of this section, we compare TPT policies with alternative policy classes for the stochastic NPV maximisation problem. In the second part, we report on the scalability of our solution procedure and assess its performance as compared to CPLEX, a general purpose optimisation package.

Apart from the illustrative example at the beginning of Section 3.5.1, all considered test instances are randomly constructed with an adapted version of the network generator ProGen/max [NSZ03], which is known to generate difficult network instances. For the construction of the network structure, we adopt the parameter values used in the UBO instances of the PSP/max benchmark library[6] (scaled to the respective problem size). For every scenario, the task cash flows are sampled from a uniform distribution on $[-100, 100]$, while the durations of the minimum time lags are selected from a uniform distribution with support $[1, 10]$. As for the maximum time lags, let δ_{ij}^s denote the start time difference between tasks i and j in scenario $s \in S$ of the early start schedule. If the network structure (as obtained from ProGen/max) prescribes a maximum time lag between i and j, then we set its duration in scenario s to $\theta_{ij}\delta_{ij}^s$, where θ_{ij} is chosen from a uniform distribution with support $\left[\underline{\theta}, \overline{\theta}\right]$. The parameters $\underline{\theta}$ and $\overline{\theta}$ describe the tightness of maximum time lags; their values will be specified later. Similarly, we choose a value of $\theta \max_{s \in S} \Delta^s$ for the deadline, where Δ^s denotes the minimum makespan for scenario $s \in S$ and θ is sampled from a uniform distribution on $\left[\underline{\theta}, \overline{\theta}\right]$. The described generation procedure ensures that feasible TPT policies exist for all instances. Throughout this section, we employ a discount factor of 0.9675.

3.5.1 TPT Policies and Alternative Problem Formulations

Consider the example network encoded through the data in Table 3.1 and Figure 3.7. In order to obtain the corresponding problem instance Υ (see Section 3.3), we apply the following transformations: (i) we discount the cash flows to the task start times; (ii) we convert the

[6]See `http://www.wior.uni-karlsruhe.de/LS_Neumann/Forschung/ProGenMax`.

maximum time lags to minimum time lags between the task start times; and (iii) we introduce an artificial sink node. The resulting network is illustrated in Figure 3.8.

	scenario 1			scenario 2			scenario 3		
	δ_i^1	z_i^1	ζ_i^1	δ_i^2	z_i^2	ζ_i^2	δ_i^3	z_i^3	ζ_i^3
task 1	9	-99.6	-74.0	3	61.0	55.2	3	43.9	39.8
task 2	5	80.7	68.4	6	145.7	119.5	4	-126.3	-110.7
task 3	5	-136.4	-115.6	6	4.5	3.7	3	-78.1	-70.7
task 4	7	-28.6	-22.7	8	74.6	57.3	10	172.3	123.8
task 5	6	-32.7	-26.8	3	-92.6	-83.9	8	-37.4	-28.7

Table 3.1: Example temporal network with 3 scenarios and occurrence probabilities $p = (0.3, 0.2, 0.5)$. Specified are the durations δ_i^s of tasks $i \in V$ in scenarios $s \in S$, the corresponding cash flows z_i^s which are realised at the task *completion* times and their discounted equivalents ζ_i^s at the task *start* times.

For a deadline of $\Delta = 30$, the optimal TPT policy is $r^* = (0, 12, 18, 3, 22)$. Here and in the remainder of this section, we suppress the artificial sink node in the results. Policy r^* has an expected NPV of 6.25, which results from NPVs of -121.2, 151.6 and 24.6 in scenarios 1, 2 and 3, respectively. The corresponding schedules are presented in Figure 3.9. We can identify the tendency to schedule tasks 1 and 4 early, whereas tasks 3 and 5 are delayed. This is in line with the expected cash flows of the tasks. Note that task 2 has a negative expected cash flow and should as such be scheduled late. We cannot assign a TPT larger than 12 to it, however, since otherwise the maximum time lag between tasks 1 and 3 would be violated in scenario 2.

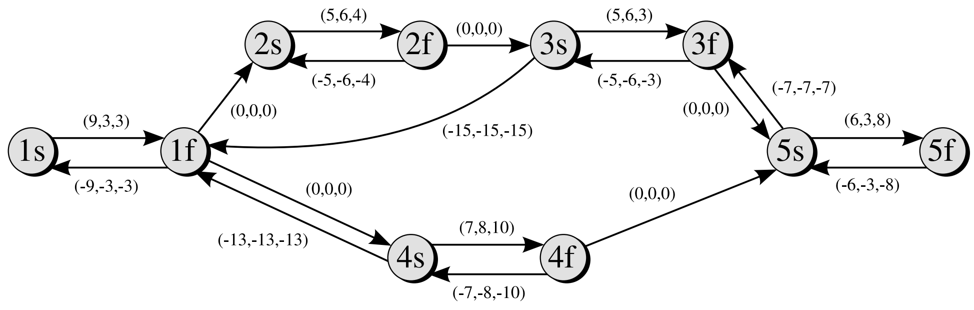


Figure 3.7: Structure of the example network in the notation of [EK92]. For $i \in V$, node is (if) represents the start (completion) event of task i. The triple of numbers attached to an arc from node i to node j describes the minimum amount of time that event i must be realised before event j in the three scenarios: the arc (5s, 3f), for example, stipulates that task 5 must start at most 7 time units after task 3 has been completed.

Our formulation properly takes into account uncertainty but results in a difficult optimisation

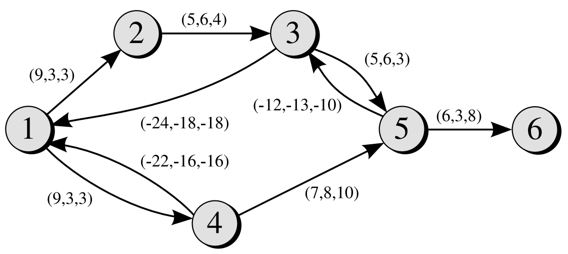


Figure 3.8: The (standardised) problem instance Υ for the network described in Table 3.1 and Figure 3.7. The triples of numbers attached to the arcs represent the values of the corresponding precedences in the three scenarios. The cash flow vector ζ is given in Table 3.1.

problem. Hence, it is tempting to relax the computational burden by solving a simplified model to obtain a feasible schedule with an acceptable expected NPV. In the following, we compare TPT policies with three alternative approaches, namely rigid policies, nominal TPT policies and task delay policies obtained from the two-stage approach in [Ben06], see Section 3.2 (hereafter referred to as TD policies). Rigid policies stipulate scenario-independent task start times that satisfy the minimum and maximum time lags in all scenarios. Contrary to TPT policies, rigid policies never require tasks to be delayed beyond their specified start times. Optimal rigid policies can be determined by solving a deterministic NPV maximisation problem which contains the time lags of all scenarios. Nominal TPT policies are obtained from a deterministic NPV maximisation problem with expected values for both the uncertain time lags and the cash flows. The solution to this deterministic problem can be interpreted as a TPT policy: every task is started as early as possible, but never before its start time in the nominal solution. Even if the optimisation problem which determines an optimal nominal policy is feasible, the resulting TPT policy may be infeasible due to the use of expected time lags. Note that by construction, both rigid and nominal policies form subsets of the class of TPT policies, and as such they can never lead to better schedules than the optimal TPT policy as determined by model (3.3). TD policies are discussed in Section 3.2.

For our example, the optimal rigid policy corresponds to the task start vector $(0, 10, 16, 9, 22)$ and an expected NPV of -9.5. The optimal nominal policy is $r^* = (0, 14.7, 19.4, 4.8, 23.6)$; this policy is infeasible, however, because the deadline is violated in scenarios 1 and 3, and the maximum time lag between tasks 1 and 3 is exceeded in scenarios 2 and 3. Hence, the nominal policy leads to infeasible schedules in all scenarios. The TD policy, finally, results in

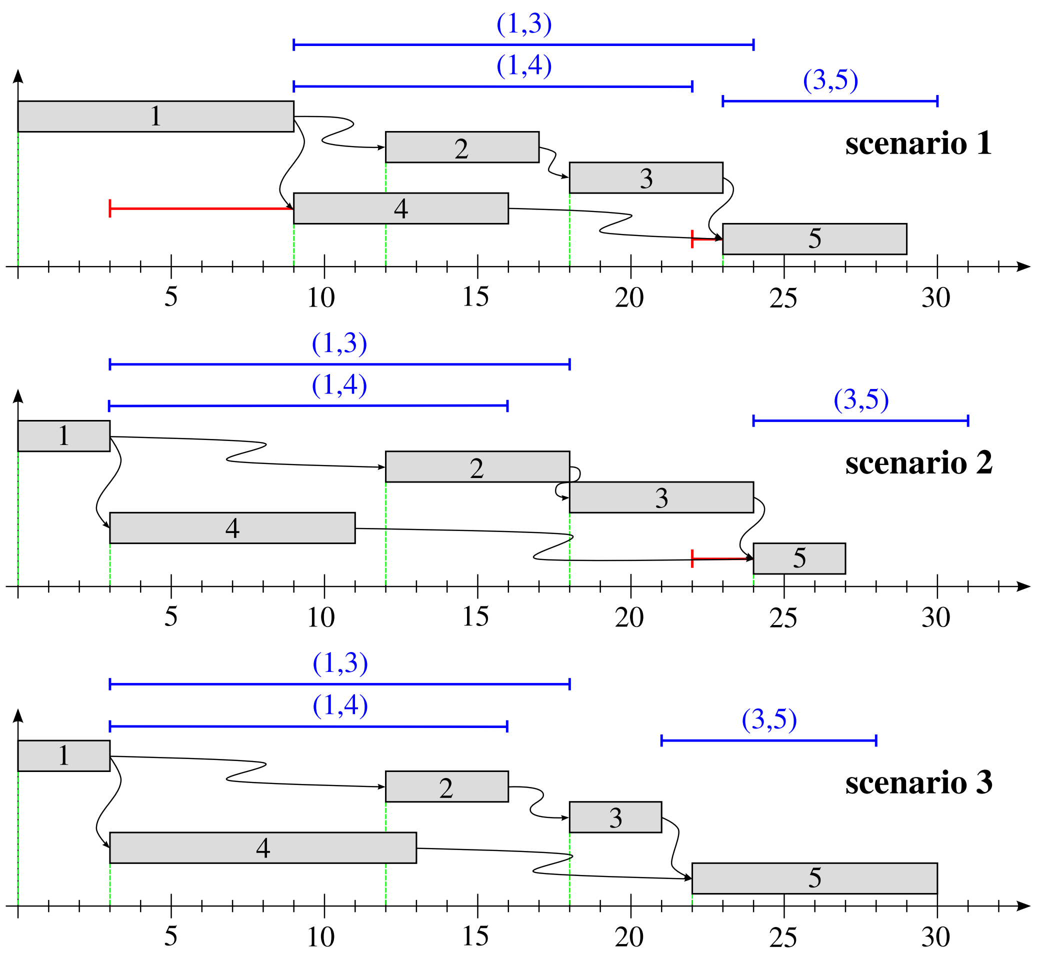


Figure 3.9: Gantt charts for the optimal TPT policy. The horizontal axis represents the elapsed time, while the vertical axis lists the network tasks. Arrows between the tasks indicate ordinary (finish–start) precedences, whereas maximum time lags are visualised by bars above the respective chart.

an expected NPV of 3.7.

Let us now determine schedules for a whole range of deadlines. Plotting the expected NPVs versus the underlying deadlines results in a curve that can be interpreted as the efficient frontier of the respective policy class. The efficient frontiers of the TPT, rigid, nominal and TD policies are shown in Figure 3.10. The TPT schedules are feasible for all considered deadlines and outperform all other schedules. TD policies perform only slightly worse than TPT policies for deadlines below 36 time units but become infeasible for larger deadlines. This undesirable effect is caused by the approximation of a stochastic problem via a deterministic one in the two-stage approach from [Ben06] (see Section 3.2) and cannot occur for the TPT policies determined by our procedure. The class of rigid policies provides feasible solutions for deadlines above 29 time units, but the resulting schedules perform substantially worse than the TPT schedules. Nominal policies, finally, yield infeasible schedules for all considered deadlines.

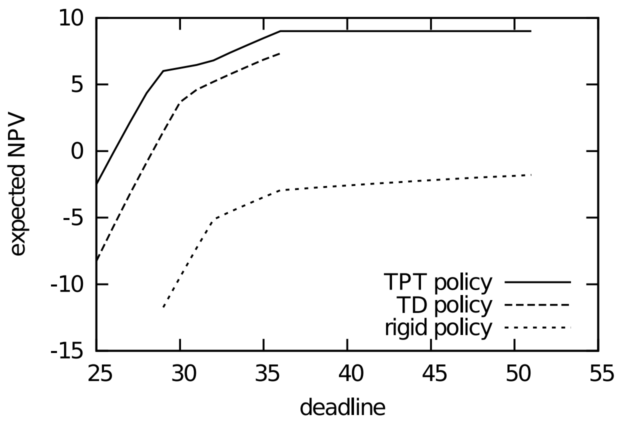


Figure 3.10: Efficient frontiers of TPT, rigid and TD policies. Nominal policies result in infeasible schedules for all considered deadlines.

Since the findings from one single test instance may not be representative, we compare the performance of the aforementioned policy classes on 500 random test instances. Every instance accommodates 3 scenarios and 10 tasks and is constructed according to the specification outlined in the beginning of Section 3.5 with $(\underline{\theta}, \overline{\theta}) = (1.25, 1.50)$. For the resulting test set, feasible TPT policies exist in 493 cases (98.6%). In contrast, feasible rigid policies can be determined for 258 instances (51.6%), feasible nominal policies for 148 instances (29.6%) and feasible TD policies for 303 instances (60.6%). For those cases where feasible policies have been found, Table 3.2 compares the resulting expected NPVs. It becomes apparent that optimal TPT policies outperform the other policy classes on the chosen test set. Although nominal and TD policies perform reasonably well on instances where they lead to feasible schedules, they are of limited use due to frequent infeasibilities.

	$q_{0.1}$	$q_{0.25}$	$q_{0.5}$	$q_{0.75}$	$q_{0.9}$
rigid policies	5.35%	8.93%	16.02%	30.24%	63.66%
nominal policies	1.71%	2.17%	4.74%	7.64%	13.47%
TD policies	−16.65%	−3.72%	2.50%	23.43%	49.32%

Table 3.2: NPV gains of TPT policies over rigid, nominal and TD policies. The entries represent the relative increase in expected NPV when optimal TPT policies are employed instead of the policy class printed in front of the respective row. q_α denotes the α-quantile over the considered instances.

3.5.2 Performance of the Branch-and-Bound Procedure

In this section, we investigate the performance of our branch-and-bound procedure and compare it with CPLEX 11.2, a state-of-the-art mixed-integer linear programming solver.[7] We also analyse the change in complexity when some of the problem parameters are varied.

We first generate random test instances of problem (3.3) with 10 scenarios, $(\underline{\theta}, \overline{\theta}) = (1.25, 1.5)$ and 10, 20, ..., 50 tasks (minimum time lags) according to the specification in the beginning of Section 3.5. For every network size, we solve 100 instances with an implementation of our branch-and-bound procedure and CPLEX 11.2 on a quad-core Intel Xeon system with 2.33GHz clock speed. In order to solve (3.3) with CPLEX, we reformulate constraint set (3.3b) via special ordered sets of type 1 [Wil99] to obtain a mixed-integer linear program. For every instance, we limit the runtime of both CPLEX and our procedure to 10 minutes and allow an optimality gap of 1%. In case an instance is not solved within this time, the respective optimisation run is considered unsuccessful and we record the incurred optimality gap. Table 3.3 summarises the test results. As expected, larger problem instances are more difficult to solve with either method. Nevertheless, our procedure was able to find optimal solutions for the majority of the test instances. In cases where an optimal solution could not be secured, the procedure determined feasible TPT policies with moderate optimality gaps. CPLEX, on the other hand, failed to find feasible TPT policies for a large percentage of the test instances. Indeed, 10 minutes runtime only proved sufficient for small instances with up to 20 tasks. We conclude that the proposed branch-and-bound procedure compares favourably to standard mixed-integer linear programming solvers.

We now investigate the impact of two important problem parameters, namely the number of scenarios and the tightness of maximum time lags. To this end, we first consider test instances with 30 tasks, $(\underline{\theta}, \overline{\theta}) = (1.25, 1.5)$ and 5, 10, 20 and 30 scenarios. Table 3.4 summarises the performance of our branch-and-bound procedure for this test set. As expected, the difficulty of problem (3.3) increases with the number of scenarios. Although the time limit is not sufficient to guarantee optimality for problem instances with 20–30 scenarios, our solution procedure

[7]CPLEX is a registered trademark of IBM ILOG.

				runtimes			optimality gaps		
size	opt.	feas.	no sol.	$q_{0.25}$	$q_{0.5}$	$q_{0.75}$	$q_{0.25}$	$q_{0.5}$	$q_{0.75}$
10	98	2	0	0.00s	0.04s	0.57s	1.3%	1.5%	1.5%
	100	0	0	0.08s	0.13s	0.41s	n/a	n/a	n/a
20	82	18	0	0.09s	0.70s	47.29s	3.1%	5.8%	7.8%
	79	11	10	2.20s	20.20s	398.96s	5.5%	172.8%	∞
30	74	26	0	0.11s	10.84s	600.00s	2.1%	4.9%	10.3%
	27	12	61	256.42s	600.00s	600.00s	∞	∞	∞
40	73	27	0	0.14s	13.04s	600.00s	1.9%	6.0%	11.5%
	19	6	75	600.00s	600.00s	600.00s	∞	∞	∞
50	69	31	0	0.11s	15.36s	600.00s	3.0%	5.0%	14.3%
	2	2	96	600.00s	600.00s	600.00s	∞	∞	∞

Table 3.3: Performance of our procedure and CPLEX for various instance sizes. Columns 2–4 describe the numbers of instances for which optimal TPT policies, suboptimal but feasible TPT policies, and no feasible TPT policies have been determined, respectively. The remaining columns document the runtimes and optimality gaps (in order) by the 0.25, 0.5 and 0.75-quantiles. For every instance size, the first and second row describe the results of our procedure and CPLEX, respectively.

consistently determined feasible TPT policies with moderate optimality gaps. Let us now consider problem instances with 30 tasks, 10 scenarios and varying values of $(\underline{\theta}, \overline{\theta})$. Table 3.5 shows that tighter maximum time lags (and deadlines) increase the difficulty of problem (3.3). Further investigations revealed that tighter maximum time lags reduce the set of feasible TPT policies, which in turn entails that the solutions $(\widehat{r}, \widehat{y})$ corresponding to the nodal upper bounds (see Step 3 of our branch-and-bound procedure) are more likely to violate constraint set (3.3b). This, however, results in a less effective pruning of the branch-and-bound tree $\mathcal{T}$ since the nodal upper bounds differ largely from the objective values of feasible TPT policies. Nevertheless, our solution procedure determined optimal or near-optimal TPT policies for all considered settings.

				runtimes			optimality gaps		
scenarios	opt.	feas.	no sol.	$q_{0.25}$	$q_{0.5}$	$q_{0.75}$	$q_{0.25}$	$q_{0.5}$	$q_{0.75}$
5	98	2	0	0.01s	0.14s	1.6s	4.5%	4.8%	4.8%
10	74	26	0	0.11s	10.84s	600.00s	2.1%	4.9%	10.3%
20	34	66	0	58.90s	600.00s	600.00s	3.0%	6.6%	21.1%
30	22	77	1	600.00s	600.00s	600.00s	4.2%	7.0%	20.7%

Table 3.4: Impact of the number of scenarios (first column) on the complexity of the problem instances. All instances exhibit 30 tasks and $(\underline{\theta}, \overline{\theta}) = (1.25, 1.5)$.

				runtimes			optimality gaps		
tightness	opt.	feas.	no sol.	$q_{0.25}$	$q_{0.5}$	$q_{0.75}$	$q_{0.25}$	$q_{0.5}$	$q_{0.75}$
$[1.00, 1.25]$	67	33	0	0.79s	21.55s	600.00s	2.7%	5.1%	13.2%
$[1.25, 1.50]$	74	26	0	0.11s	10.84s	600.00s	2.1%	4.9%	10.3%
$[1.50, 1.75]$	78	22	0	0.09s	8.62s	282.06s	1.8%	4.9%	7.9%
$[1.75, 2.00]$	88	12	0	0.01s	0.25s	11.05s	2.0%	4.9%	6.6%

Table 3.5: Impact of the maximum time lag and deadline tightness $(\underline{\theta}, \overline{\theta})$ on the complexity of the problem instances. All instances exhibit 30 tasks and 10 scenarios.

3.6 Conclusion

We proposed a model for maximising the expected NPV of a temporal network under uncertainty and developed a branch-and-bound algorithm for its solution. We illustrated the favourable performance of the model and demonstrated the superiority of the suggested solution algorithm over a state-of-the-art solver.

There is common agreement that in practise, NPV maximisation problems in temporal networks are affected by significant uncertainty. Our tests reveal that a rigorous treatment of uncertainty is necessary in order to avoid infeasible or severely suboptimal schedules. Properly accounting for uncertainty, however, inevitably leads to computationally challenging problems, even when resource restrictions are disregarded. Thus, the results in this chapter highlight the need for suitable heuristics that allow the approximate solution of large-scale (and possibly resource constrained) problem instances.

Apart from the development of heuristic solution procedures, two promising directions for future work can be identified. Firstly, although being a popular decision criterion in the literature on temporal networks, maximising the expected NPV seems to be in conflict with the risk aversion of decision makers. This problem can be alleviated by mapping cash flows to utilities (see Section 3.3), but the resulting decision criterion seems difficult to interpret. Our model and parts of the suggested solution procedure can be extended to maximise the conditional value-at-risk of the NPV. The conditional value-at-risk is a popular and well understood risk measure in the financial literature [RU00]. Secondly, formulating and solving the stochastic NPV maximisation problem as a multi-stage recourse problem with decision-dependent structure would be of interest. Albeit intractable for realistic problem sizes, such a formulation would allow the precise quantification of suboptimality incurred from the restriction to policy classes such as TPT and task delay policies. We will come back to this point in Chapter 6, where we consider dynamic NPV maximisation problems.

Chapter 4

Minimisation of Makespan Quantiles

4.1 Introduction

In this chapter, we consider temporal networks whose task durations are functions of a resource allocation that can be chosen by the decision maker. The goal is to find a feasible resource allocation that minimises the network's makespan. We focus on non-renewable resources, that is, the resources are not replenished, and specified resource budgets must be met. The resource allocation model developed in this chapter is primarily suited for project scheduling problems, and for ease of exposition we will use project scheduling terminology throughout this chapter. In project scheduling, it is common to restrict attention to non-renewable resources and disregard the per-period consumption quotas that exist for renewable and doubly-constrained resources, see Section 2.1. Apart from computational reasons, this may be justified by the fact that resource allocation decisions are often drawn at an early stage of a project's lifecycle at which the actual resource availabilities (which are unpredictable due to staff holidays, illness and other projects) are not yet known. Thus, the goal of such resource allocation models is to decide on a rough-cut plan which will be refined later.

The first resource allocation models for project scheduling have been proposed almost 50 years ago. The basic model is the linear time/cost trade-off problem [Ful61, Kel61], which considers a single resource and postulates affine relationships between investment levels and activity durations. The affinity assumption implies that the marginal costs of reducing a task's duration do not depend on the current investment level. In reality, however, the marginal costs typically increase with the investment level because additional time savings are more costly to achieve (due to reliance on overtime, rented machinery, complex process changes etc.). Indeed, linear programming theory implies that the assumption of constant marginal costs results in a pathological resource allocation behaviour: the investment level of most activities will be at one of the pre-specified investment bounds. This does not reflect reality, where prudent project managers refrain from depleting their reserves in the planning stage.

In order to overcome this weakness, several nonlinear resource allocation models have been suggested. A single-resource model with convex quadratic relationships between investment levels and task durations is presented in [DHV$^+$95]. The resulting quadratic program can be solved very efficiently. Furthermore, the marginal costs of reducing a task's duration are increasing, as desired. A resource allocation problem that assigns the single resource 'overtime' to project tasks is formulated in [JW00]. The authors postulate an inverse-proportional relationship between a task's duration and the amount of overtime spent on that task. Furthermore, the per-period costs of overtime are assumed to be quadratic in the amount of overtime, which leads to task expenditures that are linear in the investment levels. With this choice of functions, the resulting model is convex and can be solved efficiently. Apart from these two prototypical models, several solution procedures for single-resource models have been been proposed [DH02].

So far we only mentioned single-resource models. By convention, these models concentrate on the bottleneck resource within a company. In practise, however, one frequently faces situations where multiple resources (*e.g.*, both labour and capital) are scarce and need careful rationing. Note that due to market frictions different resources (such as permanent and temporary workers) are typically not equivalent or exchangeable. Hence, a multi-resource problem cannot generally be converted to a problem with a single 'canonical' resource such as capital. To the best of our knowledge, the only problem class that accounts for multiple resources is the class of discrete multi-mode problems [DH02], which also accommodates per-period consumption quotas for the resources. Multi-mode problems assume that every project task is performed in one of finitely many different execution modes, and every execution mode implies a predefined per-period consumption of every resource. Multi-mode problems are very difficult to solve due to their combinatorial nature. Firstly, it is well known that consumption quotas per unit time lead to $\mathcal{NP}$-hard 'packing' problems since the early start policy (1.2) is no longer guaranteed to be feasible, see Section 1.1. Secondly, the number of execution modes per activity is likely to be exponential in the number of resources. As a result, exact solution techniques are limited to very small projects, and one typically has to resort to heuristics.

In this chapter, we present a continuous resource allocation model for project scheduling. Contrary to existing continuous models, it can accommodate multiple resources. Unlike multi-mode problems, however, the resulting optimisation model is convex and hence computationally tractable. The relationship between investment levels and task durations is inspired by microeconomic theory, which makes the model justifiable and amenable to economic interpretation. Note that in practise, some of the resources might be discrete (such as staff or machinery). In this case, one can either solve our model as a continuous relaxation and use randomised rounding techniques, or one can treat the respective investment levels as integer variables and solve the resulting mixed-integer nonlinear program via branch-and-bound techniques.

In practise, some of the parameters of project scheduling problems (most notably the work contents of the project tasks) are subject to a high degree of uncertainty. One way to account for this uncertainty is to minimise the expected project makespan, see Chapter 3. However, as we have discussed in that chapter, the expected value is often not an appropriate decision criterion due to the non-recurring nature of projects and the high risks involved. Instead, it may be better to optimise a risk measure that also accounts for the variability of the makespan.

The classical approach to quantify the variability of a random variable is to calculate its variance [Mar52]. The variance is a reasonable risk measure when the goal is to hedge against the two-sided deviation from some target value. It is not appropriate, however, when the goal is to hedge only against the transgression of such a target: in the context of project scheduling we are concerned about the makespan exceeding a certain value, while we do not want to penalise downward deviations. Two one-sided risk measures have gained notable popularity: the value-at-risk (VaR) and the conditional value-at-risk (CVaR). The α-VaR of a random variable is defined as its α-quantile. For high values of α (*e.g.*, $\alpha \geq 0.9$), minimising the α-VaR of the project makespan leads to resource allocations that perform well in most cases. In recent years, VaR has come under criticism due to its nonconvexity, which makes the resulting optimisation models difficult to solve. Moreover, the nonconvexity implies that VaR is not sub-additive and hence not a coherent risk measure in the sense of [ADEH99]. Finally, VaR only refers to a particular quantile of a random variable but does not quantify the degree by which that quantile is exceeded 'on average', if it is exceeded. All three shortcomings are rectified by CVaR. Roughly speaking, the α-CVaR of a random variable is defined as the expected value of its $(1 - \alpha) * 100\%$ 'worst' possible realisations. Contrary to VaR, CVaR is a coherent and, a fortiori, convex risk measure, which makes it attractive for optimisation models. In the context of project scheduling, however, the advantages of CVaR over VaR seem less clear. Firstly, although the exact optimisation of the α-VaR is indeed difficult, we will see in Sections 4.3 and 4.4 that we can efficiently approximate this value with high precision. Furthermore, although being a convex risk measure, there is usually no 'attractive' closed-form expression for the CVaR, and one has to rely on costly approximation or bounding techniques. Secondly, in the context of project scheduling it is not obvious why a risk measure should be sub-additive. In a financial context, sub-additivity relates the risk of individual asset portfolios to the risk of their combination. Sub-additivity becomes more difficult to interpret in the context of managing an *individual* project, however, since such a project cannot be combined with others to form a project portfolio. Whether a quantification of the risk beyond a certain quantile of the project makespan is desirable, finally, depends strongly on the contractual agreements between client and contractor. For an overview of stochastic programming-based project scheduling techniques, see [HL05].

A popular alternative to the optimisation of VaR and CVaR is robust optimisation, see Section 2.2.2. Since robust optimisation in its 'classical' form evaluates solutions in view of their worst-case performance, it can lead to very cautious decisions. To alleviate this problem, robust optimisation has been extended to incorporate distributional information about the random variables [CSS07]. Since only partial knowledge is required about the distributions of the underlying random variables, this is particularly attractive for applications in which distributions are difficult to estimate. However, this comes at the cost of rather weak approximations of the real distributions in common cases. Indeed, as we will see in Sections 4.3 and 4.4, the use of robust optimisation techniques can result in a significant overestimation of the uncertain makespan of a project under commonly accepted distributional assumptions. Robust optimisation techniques have been applied to project scheduling problems in [CGS07, CSS07].

As part of this chapter, we extend our deterministic resource allocation model to the case of parameter uncertainty. We consider a two-stage chance constrained problem in which the resource allocation is chosen here-and-now, whereas the task start times are modelled as a wait-and-see decision, see Sections 2.2.1 and 2.3. We assume that the first and second moments of the uncertain parameters are known, and we minimise an approximation of the α-VaR of the project makespan. We also generalise our formulation to accommodate imprecise knowledge about the moments. Contrary to existing resource allocation models under uncertainty, we utilise normal approximations of the task path durations. This allows us to employ a scenario-free approach which scales favourably with the problem size. At the same time, we will see that normal approximations describe the uncertain makespan significantly better than some bounds commonly used in robust optimisation. Normal approximations of task paths have been first suggested for analysing project makespans [DH02]. Recently, they have been used to obtain bounds for 'risk-adjusted' deterministic circuit design [KBY$^+$07]. To the best of our knowledge, the use of normal approximations in the *optimisation* of temporal networks is new. Although we develop our VaR approximation in the context of project scheduling, our formulation readily applies to other application areas of temporal networks (*e.g.*, the design of digital circuits and the handling of production processes) as well.

The remainder of this chapter is organised as follows. In the next section we present our deterministic resource allocation model. In Section 4.3 we assume that some of the problem parameters are random, and we minimise an approximation of the α-VaR of the project makespan. Section 4.4 provides numerical results. In Section 4.5 we illustrate how we can accommodate imprecise moment information. We also discuss the iterative solution of our stochastic resource allocation model based on semi-infinite programming principles. We conclude in Section 4.6.

4.2 Deterministic Resource Allocation

We define a project as a temporal network $G = (V, E)$ whose nodes $V = \{1, \ldots, n\}$ denote the activities (*e.g.*, 'conduct market research' or 'develop prototype') and whose arcs $E \subseteq V \times V$ denote the temporal precedences among the activities in finish-start notation, see Section 1.1. Our goal is to find an optimal resource allocation $x \in \mathbb{R}^{mn}_{+}$, where x^k_i denotes the amount of resource $k \in K = \{1, \ldots, m\}$ assigned to activity $i \in V$. Typical project resources are capital and different categories of labour and machinery. Admissible resource allocations must satisfy process and budget constraints. We assume that the process constraints are of box type, $\underline{c} \leq x \leq \overline{c}$, where $\underline{c}$ and $\overline{c}$ are given vectors in $\mathbb{R}^{mn}_{+}$. The components $\underline{c}^k_i$ and $\overline{c}^k_i$ denote the minimal and maximal investment levels of resource k in activity i, respectively. The budget of resource k is denoted by B_k, that is, we require that $\sum_{i \in V} x^k_i \leq B_k$ for all $k \in K$. Note that all project resources are assumed to be non-renewable, which has an impact on the admissible units of measure. The resource 'labour', for example, can be measured in terms of man-hours. This implies that higher numbers of man-hours lead to shorter activity durations, which is justified by the fact that disproportionally many workers are needed in order to speed up the task execution. Indeed, if this was not the case, the resource allocation problem would (in absence of per-period consumption quotas) become trivial.

In multi-resource allocation problems, we need to specify how the joint deployment of several resources affects the duration of a project activity. In the following, we assume that activity i's duration, $d_i : \mathbb{R}^m_+ \times \mathbb{R}_{++} \mapsto \mathbb{R}_+$, is defined as $d_i(x_i; \omega_i) := \omega_i / \rho_i(x_i)$. Here, $\omega_i > 0$ denotes the work content of activity i. The work content is dimensionless and can be interpreted as the level of 'difficulty' or 'complexity' of performing task i. $x_i = (x^1_i, \ldots, x^m_i) \in \mathbb{R}^m_+$ is the subvector of x that describes the resources spent on activity i. $\rho_i : \mathbb{R}^m_+ \mapsto \mathbb{R}_{++}$ maps an investment vector x_i to its associated 'productivity'. The inverse-proportional relation between d_i and ρ_i has intuitive appeal since higher productivities should result in shorter task durations. As we will see in the following, this relation preserves desirable properties of the productivity mapping ρ_i.

We are thus led to the problem of specifying appropriate productivity mappings ρ_i. Natural candidates are production functions from microeconomics: a production function determines the output quantity (*e.g.*, the lot size of a certain product) of a production process as a function of the input factors (*e.g.*, the amount of labour and capital employed). In our case, the output is a productivity, that is, the capacity to carry out work that is related to the completion of a project task. Two classes of production functions are common in microeconomics since they describe resource interactions often observed in practise [MCWG95]. Limitational functions describe production processes which combine the input factors in a fixed proportion (*e.g.*, cars consist of four tyres and one steering wheel). Substitutional functions, on the other hand, reflect processes where the abundance of some input factors can be used to partially offset the shortage of others (*e.g.*, different types of fertiliser in the cultivation of land).

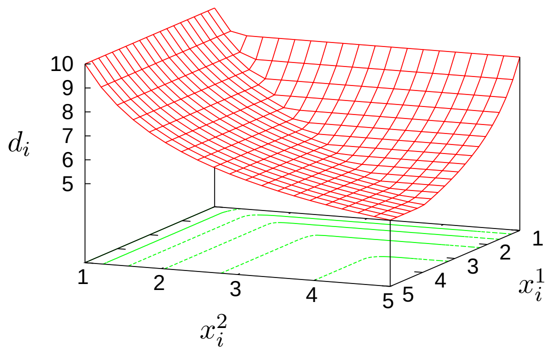
We define limitational productivity mappings as

$$\rho_i^{\mathrm{L}}(x_i) := \delta_i \min \left\{ \psi_i^k x_i^k \, : \, k \in K, \; \psi_i^k > 0 \right\}^{\gamma_i}. \tag{4.1}$$

$\delta_i > 0$ describes the efficiency of the process underlying activity i but can be omitted when ω_i is suitably scaled. $\psi_i \in \mathbb{R}_+^m$ characterises the optimal input factor ratio, that is, the investment weights that lead to zero wastage. The exponent $\gamma_i > 0$ determines the degree of homogeneity: for any scaling parameter $\lambda \geq 0$ we have $d_i(\lambda x_i; \omega_i) = \lambda^{-\gamma_i} d_i(x_i; \omega_i)$. Hence, a λ-fold increase of every input factor leads to a λ^{γ_i}-fold decrease in task duration. Limitational productivity mappings have zero substitution elasticity, that is, it is not possible to substitute one input factor by another. The left part of Figure 4.1 visualises this type of productivity mapping. In the context of project scheduling, typical examples of limitational productivity mappings are predefined team structures (*e.g.*, one foreman and five untrained workers form a team) and the incorporation of machinery or materials (*e.g.*, four workers are required to operate one flexible manufacturing system). One can show that if all activity durations are determined by limitational productivity mappings, the allocation problem can be reformulated as a single-resource problem.

We define substitutional (Cobb-Douglas) productivity mappings as

$$\rho_i^{\mathrm{S}}(x_i) := \delta_i \prod_{k \in K} (x_i^k)^{\psi_i^k}, \tag{4.2}$$



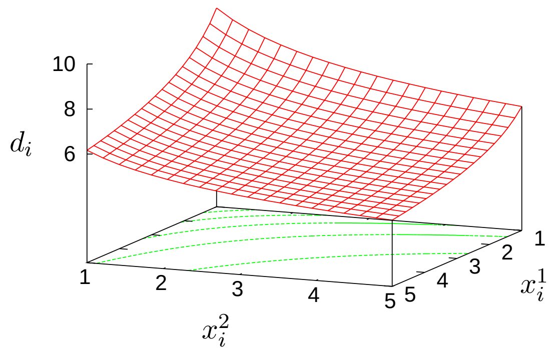


Figure 4.1: Activity duration depending on two input factors which are combined in a limitational (left) and substitutional (right) process. Abundance of a single resource leads to wastage in the former case, whereas it leads to further time savings in the latter one.

where $\delta_i > 0$ is again an efficiency parameter that can be transformed away. The exponents $\psi_i \in \mathbb{R}_+^m$ specify the partial elasticities of d_i with respect to x_i:

$$\frac{\partial d_i(x_i;\omega_i)/\partial x_i^p}{d_i(x_i;\omega_i)/x_i^p} = \frac{-(\omega_i/\delta_i)\psi_i^p (x_i^p)^{-\psi_i^p-1}\prod_{k\neq p}(x_i^k)^{-\psi_i^k}}{(\omega_i/\delta_i)(x_i^p)^{-\psi_i^p-1}\prod_{k\neq p}(x_i^k)^{-\psi_i^k}} = -\psi_i^p.$$

Hence, a marginal increase of x_i^p leads, ceteris paribus, to a ψ_i^p-fold decrease of d_i. We furthermore see that d_i is homogeneous of degree $-\sum_{k\in K}\psi_i^k$; this term has the same interpretation as $-\gamma_i$ in (4.1). The marginal rate of technical substitution (MRTS) of input p for input q amounts to

$$\mathrm{MRTS}_{p,q} = \frac{\partial d_i(x_i;\omega_i)/\partial x_i^p}{\partial d_i(x_i;\omega_i)/\partial x_i^q} = \frac{-(\omega_i/\delta_i)\psi_i^p (x_i^p)^{-\psi_i^p-1}\prod_{k\neq p}(x_i^k)^{-\psi_i^k}}{-(\omega_i/\delta_i)\psi_i^q (x_i^q)^{-\psi_i^q-1}\prod_{k\neq q}(x_i^k)^{-\psi_i^k}} = \frac{\psi_i^p x_i^q}{\psi_i^q x_i^p}.$$

Thus, in order to keep activity duration d_i unchanged, a marginal decrease of x_i^p requires a $(\psi_i^p x_i^q)/(\psi_i^q x_i^p)$-fold increase of x_i^q. The right part of Figure 4.1 visualises the Cobb-Douglas productivity mapping. In project scheduling, substitutional productivity mappings arise from outsourcing decisions (part of an activity is done in-house, the rest is outsourced), flexible degrees of automation (labour and capital are often substitutes within certain ranges) and flexible team structures (several untrained workers can replace a trained worker).

In the following, we denote by V^{L} and V^{S} the sets of activities whose durations are determined by limitational and substitutional productivity mappings, respectively. We assume that $V = V^{\mathrm{L}} \cup V^{\mathrm{S}}$ and $V^{\mathrm{L}} \cap V^{\mathrm{S}} = \emptyset$. The resulting deterministic resource allocation model can be described as follows.

$$\begin{array}{lll l}
\underset{x,y}{\text{minimise}} & y_n + d_n(x_n;\omega_n) & & \text{(4.3a)} \\
\text{subject to} & x \in \mathbb{R}_+^{mn}, \quad y \in \mathbb{R}_+^n & & \\
& y_j \geq y_i + d_i(x_i;\omega_i) & \forall\,(i,j) \in E & \text{(4.3b)} \\
& \sum_{i \in V} x_i^k \leq B_k & \forall\, k \in K & \text{(4.3c)} \\
& x_i^k \in \left[\underline{c}_i^k, \overline{c}_i^k\right] & \forall\, i \in V,\, k \in K & \text{(4.3d)}
\end{array}$$

In this model, decision vector $y \in \mathbb{R}_+^n$ contains the start times of the project activities. The objective is to minimise the project makespan, which is given by the completion time of activity n. Constraint (4.3b) enforces the temporal precedences between the project tasks, while constraints (4.3c) and (4.3d) enforce the budget and process constraints, respectively. For future use, we define $X := \left\{x \in \mathbb{R}_+^{mn} \,:\, x \text{ satisfies (4.3c) and (4.3d)}\right\}$.

From a computational viewpoint, the following observation is crucial.

Proposition 4.2.1 *(4.3) can be formulated as a convex optimisation model.*

Proof Without loss of generality, we can assume that $\omega_n = 0$. As a result, the only nonlinearity occurs in constraint (4.3b). By a slight abuse of notation, we introduce variables $d \in \mathbb{R}_+^n$ for the task durations and replace constraint (4.3b) with

$$\begin{array}{lll}
y_j \geq y_i + d_i & \forall\,(i,j) \in E & \text{(4.3b}'\text{)} \\
d_i \rho_i(x_i) \geq \omega_i & \forall\, i \in V. & \text{(4.3b}''\text{)}
\end{array}$$

Because we are minimising the project's makespan, there is always an optimal solution to the new model that satisfies (4.3b$''$) as equality. This establishes equivalence to the original model.

By construction, $d_i \rho_i(x_i)$ is log-concave in (d_i, x_i) for substitutional activities. For a limitational activity, we note that

$$d_i \rho_i^{\mathrm{L}}(x_i) \geq \omega_i \quad \Longleftrightarrow \quad d_i \delta_i (\psi_i^k x_i^k)^{\gamma_i} \geq \omega_i \quad \forall\, k \in K \,:\, \psi_i^k > 0,$$

and the left-hand sides of the latter constraint group are log-concave in (d_i, x_i) as well. Thus, the feasible region of the extended optimisation problem is convex. ∎

We illustrate model (4.3) with an example.

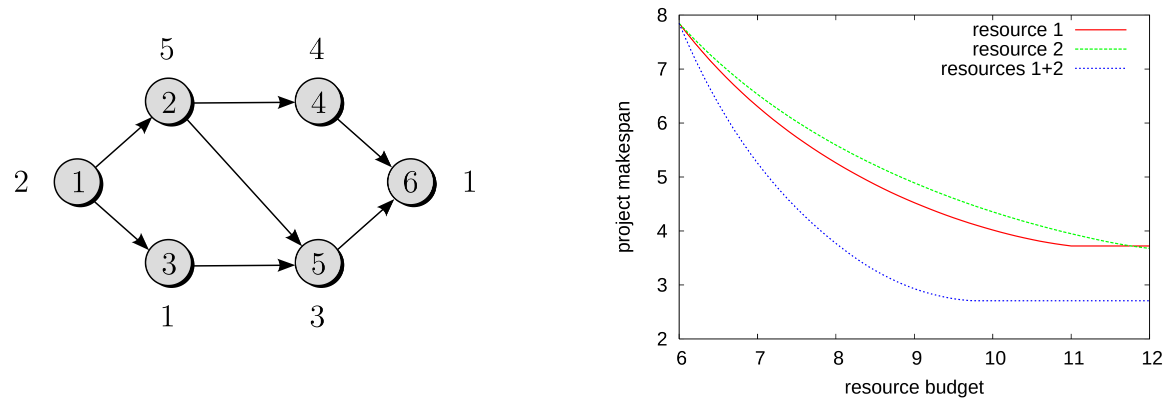


Figure 4.2: Deterministic resource allocation for an example project. The left chart illustrates the project network and the activities' work contents (attached to the nodes). The right chart presents the project's makespan as a function of the resource budgets. Two of the curves vary the budget of one resource while the other budget is fixed at 6. The third curve simultaneously varies the budget of both resources.

Example 4.2.1 *Consider the project in Figure 4.2 (left). Apart from the missing cash flows, it is identical to the temporal network in Figure 1.1. Now, however, we interpret the numbers attached to the network tasks as the work contents of the project activities. We consider two resources with process constraints* $x^k \in [(1/4)\mathrm{e}, 2\mathrm{e}]$ *and budget constraints* $\sum_{i \in V} x_i^k \leq 6$, $k \in \{1, 2\}$. *Activity 4 has a limitational productivity mapping, whereas all other activities are described by substitutional productivity mappings:*

$$\rho_i(x_i) := \begin{cases} \min\left\{2x_i^1,\ x_i^2\right\} & \textit{if } i = 4, \\ \left(x_i^1\right)^2 \left(x_i^2\right)^{3/2} & \textit{otherwise.} \end{cases}$$

Since the work content attached to the sink node 6 *is nonzero, we introduce an auxiliary variable* τ *that represents the completion of the project. Using the reformulation from Proposition 4.2.1, we can then formulate the deterministic resource allocation model (4.3) as follows.*

$$
\begin{aligned}
\underset{d,x,y,\tau}{\text{minimise}} \quad & \tau \\
\text{subject to} \quad & d \in \mathbb{R}^6_+, \quad x \in \mathbb{R}^{12}_+, \quad y \in \mathbb{R}^6_+, \quad \tau \in \mathbb{R}_+ \\
& y_2 \geq y_1 + d_1, \quad y_3 \geq y_1 + d_1, \quad y_4 \geq y_2 + d_2 \\
& y_5 \geq y_2 + d_2, \quad y_5 \geq y_3 + d_3, \quad y_6 \geq y_4 + d_4 \\
& y_6 \geq y_5 + d_5, \quad \tau \geq y_6 + d_6 \\
& d_1 \left(x_1^1\right)^2 \left(x_1^2\right)^{3/2} \geq 2, \quad d_2 \left(x_2^1\right)^2 \left(x_2^2\right)^{3/2} \geq 5, \\
& d_3 \left(x_3^1\right)^2 \left(x_3^2\right)^{3/2} \geq 1, \quad d_5 \left(x_5^1\right)^2 \left(x_5^2\right)^{3/2} \geq 3, \\
& d_6 \left(x_6^1\right)^2 \left(x_6^2\right)^{3/2} \geq 1, \\
& 2d_4 x_4^1 \geq 4, \quad d_4 x_4^2 \geq 4, \\
& x^1, x^2 \in \left[(1/4)\mathrm{e}, 2\mathrm{e}\right], \quad \sum_{i=1}^{6} x_i^1 \leq 6, \quad \sum_{i=1}^{6} x_i^2 \leq 6.
\end{aligned}
$$

The optimal resource allocation to this problem is $x^1 \approx (1.22, 1.38, 0.80, 0.52, 1.03, 1.05)$ *and* $x^2 \approx (1.10, 1.25, 0.73, 1.05, 0.93, 0.95)$*, and the associated makespan is* $\tau \approx 7.85$.

Let us now investigate the impact of the resource budgets on the project's makespan. As Figure 4.2 (right) shows, the project makespan decreases if we increase the resource budgets. If we only increase the budget of resource 1, then resource 2 soon becomes a bottleneck and we cannot decrease the makespan beyond 3.72. This is due to the fact that task 4 *requires a larger amount of resource 2. If we simultaneously increase the budget of both resources, however, we can avoid this bottleneck by substituting resource 2 with resource 1 in the activities* $i \in V \setminus \{4\}$*. We obtain the minimal project makespan* 2.71 *by assigning a budget of* 9.8 *to both resources.*

We close with three remarks about our deterministic resource allocation model.

Firstly, if all productivity mappings contain rational exponents, model (4.3) can be formulated as a conic quadratic program [AG03]. Depending on the values of these exponents, this can lead to performance improvements over solving the model with a general convex optimiser.

Secondly, model (4.3) only accommodates simple productivity mappings. Sometimes one may require nested productivity mappings that map investment levels and/or productivity values to (new) productivity values. For example, a trade-off between a limitational labour process (*e.g.*, foreworkers and untrained labour have to satisfy a proportion of 1:4) and capital (*e.g.*, an outsourcing decision) can be modelled as a two-stage process. It is easy to extend our scheme to nested productivity mappings such that the resulting model remains convex and representable as conic quadratic program.

Finally, the parameter values of the productivity mappings might be unavailable in practise. Nevertheless, one can assume that at least the type of productivity mapping (limitational or substitutional) is known for each activity. With this knowledge, one can estimate the missing parameter values based on a set of expected durations for different resource combinations.

4.3 Resource Allocation under Uncertainty

In the remainder of this chapter, we assume that the vector of work contents $\widetilde{\omega}$ constitutes a random vector with finite first and second moments. By convention, all random objects in this chapter, which are indicated by the tilde sign, are defined on an abstract probability space $(\Omega, \mathcal{F}, \mathbb{P})$. In contrast to the work contents, all other parameters remain deterministic. For references to models in which the project graph G or the process and budget constraints are uncertain, see Section 2.3. The parameters of the productivity mappings should in our view be treated as deterministic numbers. In fact, it is unlikely that the decision maker can specify meaningful distributions for them. Moreover, the impact of uncertain work contents can outweigh by far the consequences of not knowing the exact productivity mappings. Thus, little accuracy might be lost when we assume the latter to be deterministic.

We consider static resource allocations that are chosen before any of the uncertain work contents is revealed. The corresponding decision vector x is thus a here-and-now decision, see Section 2.2.1. Static allocations are frequently required when some resources (such as labour and machinery) cannot be shifted between different activities on short notice [HL05]. Even if it was admissible to adapt the resource allocation during project implementation, a static allocation might still be preferable from the viewpoint of computational tractability, see [GG06, JWW98] and Chapter 6. Contrary to the resource allocation x, we assume that the activity start times y are allowed to depend on the realisation of the uncertain work contents $\widetilde{\omega}$. In the terminology of Section 2.2.1, y is thus a wait-and-see decision. Indeed, if y was modelled as a here-and-now decision, we would seek for a schedule of a priori fixed activity start times that can always (or with high reliability) be met [HL05]. Since we assume absence of consumption quotas per unit time (see Section 4.1), however, there is no benefit in knowing the activity start times before project implementation. Thus, fixed start times would unnecessarily increase the project's makespan in our setting.

We recall the definition of the value-at-risk at level α (α-VaR) of a random variable $\widetilde{X}$:

$$\alpha\text{-VaR}\big(\widetilde{X}\big) := \min\big\{t \,:\, \mathbb{P}(\widetilde{X} \leq t) \geq \alpha\big\}.$$

Hence, the α-VaR of $\widetilde{X}$ is simply the α-quantile of the distribution of $\widetilde{X}$. In the face of uncertainty about the work contents, our new goal is to minimise the α-VaR of the random project makespan. This results in the following reformulation of the deterministic model (4.3).

$$\begin{aligned}
&\underset{x,\tau}{\text{minimise}} && \tau && \text{(4.4a)}\\
&\text{subject to} && x \in \mathbb{R}^{mn}_{+}, \quad \tau \in \mathbb{R}_{+} \\
& && \mathbb{P}\left(\exists\, y \geq 0 \,:\, \left\{\begin{array}{ll} \tau \geq y_n + d_n(x_n; \widetilde{\omega}_n) & \\ y_j \geq y_i + d_i(x_i; \widetilde{\omega}_i) & \forall\,(i,j) \in E \end{array}\right\}\right) \geq \alpha, && \text{(4.4b)}\\
& && x \in X. && \text{(4.4c)}
\end{aligned}$$

Model (4.4) constitutes a two-stage chance constrained stochastic program, see Section 2.2.1. The uncertain work contents $\widetilde{\omega}$ are revealed after the resource allocation x has been chosen, but before the activity start times y are selected. The joint chance constraint (4.4b) ensures that τ is a valid upper bound on the project makespan with probability at least α. Since τ is minimised, model (4.4) indeed minimises the α-VaR of the project makespan.

The fact that y is chosen after *all* uncertain work contents have been revealed seems to violate non-anticipativity [Pré95, RS03]: in order to be implementable, the start time y_j of activity j must only depend on work contents that are known at the time when j is started. The uncertain work content of an activity, however, is only known after its completion. Since model (4.4) principally allows y_j to depend on all components of $\widetilde{\omega}$, the resulting optimal policy could therefore be acausal. Fortunately, it turns out that the non-anticipative 'early start schedule' is always among the optimal solutions to problem (4.4). Since our project graph is acyclic, the early start schedule can be calculated recursively via

$$y_j^*(x;\omega) = \max\left\{0, \sup_{i\in V}\left\{y_i^*(x;\omega) + d_i(x_i;\omega_i) \;:\; (i,j)\in E\right\}\right\} \quad \forall\, j \in V$$

for every fixed x and ω. Note that this schedule is non-anticipative since the start time of an activity only depends on the completion times of predecessor activities, that is, only knowledge about work contents of completed activities is required. Furthermore, absence of per-period resource consumption quotas guarantees that the early start schedule is always feasible. Finally, since the makespan is a non-decreasing function of the activity start times, the early start schedule minimises the scenario makespan of the project for any fixed x and ω. Hence, if an optimal solution to problem (4.4) contains an anticipative start time schedule y, we can replace it with the (non-anticipative) early start schedule without sacrificing optimality.

Two-stage chance constrained problems of type (4.4) are notoriously difficult to solve [EI07]. Several approximate solution methods have been suggested in the literature, such as sampling-based variants of the ellipsoid method [EI07, NS06b], convex approximation via CVaR constraints [WA08] and methods based on affine decision rules [CSS07]. In the following, we will consider a reformulation of problem (4.4) that eliminates the two-stage structure. We will compare our approach with direct approximations of (4.4) via CVaR constraints in Section 4.4. Affine decision rules are studied in Chapter 5.

We eliminate the two-stage structure of problem (4.4) by enumerating the activity paths of the project graph. Apart from reducing the model to a single-stage problem, this approach enables us to employ normal approximations for the distributions of the path durations that can be justified by a generalised central limit theorem. It is well known that in the worst case, the number of activity paths is exponential in the size of the project graph. Since our reformulation will contain one constraint per activity path, this implies that our model can potentially contain an exponential number of constraints. As we will see in Section 4.4, however, typical project instances seem to contain only moderate numbers of activity paths. Furthermore, we will discuss a technique which alleviates the problem of large path numbers in Section 4.5.2. We caution the reader that in other application areas of temporal networks, the number of network paths can be huge. In Chapter 5 we will discuss a technique to minimise the worst-case makespan of networks with large numbers of paths.

We recall that a path in a directed graph $G = (V, E)$ constitutes a list of nodes $(i_1, \ldots, i_p)$ such that $(i_1, i_2), \ldots, (i_{p-1}, i_p) \in E$. We define an activity path $P = \{i_1, \ldots, i_p\} \subseteq V$ as a set of project activities that form a path in the project graph G. We denote by $\overline{\mathcal{P}}$ the set of inclusion-maximal paths, that is, $\overline{\mathcal{P}}$ contains all activity paths that are not strictly included in any other path. Observe that a project's makespan in a given scenario equals the duration of the most time-consuming path in that scenario. Hence, we can reformulate problem (4.4) equivalently as follows.

$$\begin{aligned} \underset{x,\tau}{\text{minimise}} \quad & \tau && \text{(4.5a)} \\ \text{subject to} \quad & x \in \mathbb{R}^{mn}_{+}, \quad \tau \in \mathbb{R}_{+} \\ & \mathbb{P}\Big(\tau \geq \sum_{i \in P^l} d_i(x_i; \widetilde{\omega}_i) \quad \forall\, P^l \in \overline{\mathcal{P}}\Big) \geq \alpha, && \text{(4.5b)} \\ & x \in X. && \text{(4.5c)} \end{aligned}$$

Note that (4.5) only involves here-and-now decisions (x, τ) and hence constitutes a single-stage chance constrained problem. Like constraint (4.4b), however, (4.5b) still constitutes a joint chance constraint in which the random variables cannot easily be separated from the decision variables. Apart from some benign special cases, problems of type (4.5) generically have nonconvex or even disconnected feasible sets, which severely complicates their numerical solution. Well-structured chance constrained problems that have convex feasible sets for all or for sufficiently high values of α are discussed in [HS08, Pré95]. One readily verifies, however, that model (4.5) does not belong to these problem classes. The following example shows that model (4.5) is indeed nonconvex.

Example 4.3.1 *Consider the project $G = (V, E)$ with node set $V = \{1, \ldots, 4\}$ and precedences $E = \{(1,2), (1,3), (2,4), (3,4)\}$. For the sake of simplicity, let us assume that $\widetilde{\omega}_1 = \widetilde{\omega}_4 = 0$ almost surely, $\widetilde{\omega}_2$ and $\widetilde{\omega}_3$ follow independent standard normal distributions and $\rho_i(x_i) = x_i$, $i = 2, 3$. The process constraints are $1/2 \leq x_2, x_3 \leq 2$, and there is no resource budget. We want to investigate the convexity of the feasible region*

$$\begin{aligned} X(\alpha) &:= \left\{(\tau, x_2, x_3) \in \mathbb{R}_+ \times [1/2, 2]^2 \, : \, \mathbb{P}\left(\tau \geq \widetilde{\omega}_2/x_2, \; \tau \geq \widetilde{\omega}_3/x_3\right) \geq \alpha\right\} \\ &= \left\{(\tau, x_2, x_3) \in \mathbb{R}_+ \times [1/2, 2]^2 \, : \, \Phi(\tau x_2)\, \Phi(\tau x_3) \geq \alpha\right\}. \end{aligned}$$

It is easy to see that $X(\alpha)$ is generically nonconvex. Indeed, for $\alpha = 2/3$ one can verify that $(\tau^1, x_2^1, x_3^1) = (1, 2, 1/2), (\tau^2, x_2^2, x_3^2) = (1, 1/2, 2) \in X(2/3)$, but $(\tau, x_2, x_3) = 1/2(\tau^1, x_2^1, x_3^1) + 1/2(\tau^2, x_2^2, x_3^2) = (1, 5/4, 5/4) \notin X(2/3)$.

Recently, sample approximation [LA08] and scenario approximation techniques [CC05, CC06] have been proposed for solving joint chance constrained problems of type (4.5). Applied to our setting, however, sample approximation would lead to large mixed-integer nonlinear programs (even in absence of discrete resources), which themselves constitute difficult optimisation problems. Likewise, solving (4.5) with scenario approximation techniques would result in a problem whose number of constraints is proportional to the cardinality of $\overline{\mathcal{P}}$ times the number of scenarios employed. Since this product is large in realistic settings, this approach seems primarily interesting for small projects.

In this chapter, we employ Boole's inequality to approximate (4.5) as follows.

$$
\begin{aligned}
\underset{x,\beta,\tau}{\text{minimise}} \quad & \tau && (4.6\text{a})\\
\text{subject to} \quad & x \in \mathbb{R}_+^{mn}, \quad \beta \in \mathbb{R}_+^{|\overline{\mathcal{P}}|}, \quad \tau \in \mathbb{R}_+ \\
& \mathbb{P}\Big(\tau \geq \sum_{i \in P^l} d_i(x_i; \widetilde{\omega}_i)\Big) \geq \beta_l \qquad \forall\, P^l \in \overline{\mathcal{P}}, && (4.6\text{b})\\
& \sum_{P^l \in \overline{\mathcal{P}}} \beta_l \geq \alpha + (|\overline{\mathcal{P}}| - 1), && (4.6\text{c})\\
& \beta_l \in [0,1] \qquad \forall\, P^l \in \overline{\mathcal{P}}, && (4.6\text{d})\\
& x \in X. && (4.6\text{e})
\end{aligned}
$$

For future use, we define $B := \big\{\beta \in \mathbb{R}_+^{|\overline{\mathcal{P}}|} \ : \ \beta \text{ satisfies (4.6c) and (4.6d)}\big\}$. Note that in (4.6b) we have split up the joint chance constraint of model (4.5) into independent separated chance constraints. The following proposition shows that model (4.6) constitutes a conservative approximation of (4.5), see also [NS06a].

Proposition 4.3.1 *If (x, β, τ) is a feasible solution to model (4.6), then (x, τ) is also feasible in model (4.5).*

Proof Using the feasibility of (x, β, τ) in problem (4.6), we find that

$$
\begin{aligned}
&\mathbb{P}\Big(\tau \geq \sum_{i \in P^l} d_i(x_i; \widetilde{\omega}_i)\ \forall\, P^l \in \overline{\mathcal{P}}\Big) \ = \ 1 - \mathbb{P}\Big(\bigcup_{P^l \in \overline{\mathcal{P}}} \Big\{\tau < \sum_{i \in P^l} d_i(x_i; \widetilde{\omega}_i)\Big\}\Big)\\
&\geq 1 - \sum_{P^l \in \overline{\mathcal{P}}} \mathbb{P}\Big(\tau < \sum_{i \in P^l} d_i(x_i; \widetilde{\omega}_i)\Big) \ = \ 1 - \sum_{P^l \in \overline{\mathcal{P}}} \Big[1 - \mathbb{P}\Big(\tau \geq \sum_{i \in P^l} d_i(x_i; \widetilde{\omega}_i)\Big)\Big]\\
&\geq 1 - \sum_{P^l \in \overline{\mathcal{P}}} (1 - \beta_l) \ = \ 1 - |\overline{\mathcal{P}}| + \sum_{P^l \in \overline{\mathcal{P}}} \beta_l \ \geq \ \alpha.
\end{aligned}
$$

Here, the first inequality follows from Boole's inequality.[1] ∎

[1] Boole's inequality: For a countable set of events $A_1, A_2, \ldots \in \mathcal{F}$, $\mathbb{P}\left(\bigcup_i A_i\right) \leq \sum_i \mathbb{P}(A_i)$.

Observe that for $\alpha < 1$, both (4.5) and (4.6) are feasible if and only if $X \neq \emptyset$. The optimal objective value of (4.6), however, is greater than or equal to the optimal objective value of (4.5). The approximation (4.6) can in principle be tightened by incorporating pairs of activity paths via Bonferroni's inequalities [Pré95]. This, however, either requires an a priori fixed choice of admissible path pairs or a selection procedure that determines optimal pairs in an iterative manner [Pré95]. The former approach is likely to result in a substantial increase of problem size, while the latter technique requires the repeated solution of model (4.6). Since Boole's approximation turns out to be remarkably tight in our numerical tests (see Section 4.4), the potential gains of either approach are likely to be outweighed by the increase in complexity. Hence, we settle for Boole's inequality in the following.

Model (4.6) still constitutes a generically nonconvex problem. More so, even the verification whether a given point is feasible requires the evaluation of multi-dimensional integrals and thus becomes prohibitively expensive for realistic problem sizes. In recent years, several inequalities from probability theory have been employed to obtain conservative convex approximations of separated chance constraints [CSS07, NS06a]. We will not pursue these approaches here. Instead, we simplify constraint (4.6b) by approximating the path durations $\sum_{i \in P^l} d_i(x_i; \widetilde{\omega}_i)$, $P^l \in \overline{\mathcal{P}}$, via normal distributions. As we will see, this approximation has theoretical appeal and leads to superior results in numerical tests.

Let the first and second moments of $\widetilde{\omega}$ be given by $\mu = (\mathbb{E}[\widetilde{\omega}_1], \ldots, \mathbb{E}[\widetilde{\omega}_n])^\top$ and $\Sigma \in \mathbb{R}^{n \times n}$, $\Sigma_{ij} = \operatorname{Cov}(\widetilde{\omega}_i, \widetilde{\omega}_j)$. In order to simplify the notation, we furthermore introduce functions $\varrho_l : \mathbb{R}^{mn}_+ \mapsto \mathbb{R}^n_+$, $P^l \in \overline{\mathcal{P}}$, with

$$[\varrho_l(x)]_i = \begin{cases} 1/\rho_i(x_i) & \text{if } i \in P^l, \\ 0 & \text{otherwise.} \end{cases}$$

Using this notation, we can express the mean and variance of the path duration $\sum_{i \in P^l} d_i(x_i; \widetilde{\omega}_i)$ as $\mu^\top \varrho_l(x)$ and $\varrho_l(x)^\top \Sigma \, \varrho_l(x)$, respectively, for each $P^l \in \overline{\mathcal{P}}$. Our proposed solution method for problem (4.6) approximates the duration of path P^l by a normal distribution with the same first and second moments. The following generalised central limit theorem justifies the use of such normal approximations in project scheduling under three alternative regularity conditions.

Theorem 4.3.1 *Let $P_\nu = \{1, \ldots, \nu\}$, $\nu = 1, 2, \ldots$, be an inclusion-increasing sequence of project paths with task durations $d_i(x_i; \widetilde{\omega}_i) = \widetilde{\omega}_i / \rho_i$, $\rho_i \in \left[\underline{\rho}, \overline{\rho}\right]$, $\underline{\rho} > 0$, and $\widetilde{\omega}_i \geq \underline{\omega} > 0$ $\mathbb{P}$-a.s. for all i. Assume that the first three moments of $\widetilde{\omega}_i$ are finite and satisfy*

$$\begin{aligned} \mu_i &= \mathbb{E}(\widetilde{\omega}_i) \leq \overline{\mu}, \\ \sigma_i^2 &= \mathrm{Var}(\widetilde{\omega}_i) \in [\underline{\sigma}^2, \overline{\sigma}^2] \quad \textit{with } \underline{\sigma}^2 > 0 \\ \textit{and} \quad \gamma_i^3 &= \mathbb{E}\big(\,|\widetilde{\omega}_i - \mu_i|^3\,\big) \leq \overline{\gamma}^3. \end{aligned}$$

Then for any fixed resource allocation, the standardised path durations converge in distribution to a standard normal distribution as $\nu \longrightarrow \infty$ if either of the following three conditions holds:

(C1) The components of $\widetilde{\omega}$ follow a multivariate normal distribution.

(C2) The components of $\widetilde{\omega}$ are independent.

(C3) There is a time lag $T \in \mathbb{R}_+$ such that $\widetilde{\omega}_i$ and $\widetilde{\omega}_j$ are independent if the start times of tasks i and j differ by at least T time units. Furthermore, the covariances of dependent work contents are bounded from above by some $\zeta \in \mathbb{R}_+$ and $\lim_{\nu \longrightarrow \infty} \nu^{-1} \mathrm{Var}(\sum_{i \in P_\nu} d_i(x_i; \widetilde{\omega}_i))$ exists and is nonzero.

Proof Since the duration of any project path is linear in $\widetilde{\omega}$, it is normally distributed if $\widetilde{\omega}$ follows a multivariate normal distribution. Thus, we obtain the stronger result that under (C1), all path durations are normally distributed.

In the following, we abbreviate $d_i(x_i; \widetilde{\omega}_i)$ by $\widetilde{d}_i$. Under assumption (C2), the assertion follows from Lyapunov's central limit theorem [Pet75]. Apart from finite first and second moments (which are implied by our assumptions), this theorem requires that

$$\lim_{\nu \longrightarrow \infty} \frac{\Big(\sum_{i \in P_\nu} \mathbb{E}\Big(\big|\widetilde{d}_i - \mathbb{E}(\widetilde{d}_i)\big|^3\Big)\Big)^{1/3}}{\Big(\sum_{i \in P_\nu} \mathrm{Var}\big(\widetilde{d}_i\big)\Big)^{1/2}} = 0.$$

Employing the estimates

$$\mathbb{E}\Big(\big|\widetilde{d}_i - \mathbb{E}(\widetilde{d}_i)\big|^3\Big) \leq \frac{1}{\underline{\rho}^3}\mathbb{E}\big(\,|\widetilde{\omega}_i - \mathbb{E}(\widetilde{\omega}_i)|^3\,\big) \leq \frac{\overline{\gamma}^3}{\underline{\rho}^3}$$

and

$$\mathrm{Var}(\widetilde{d}_i) \geq \frac{1}{\overline{\rho}^2}\mathrm{Var}(\widetilde{\omega}_i) \geq \frac{\underline{\sigma}^2}{\overline{\rho}^2},$$

we obtain

$$\frac{\Big(\sum_{i\in P_\nu}\mathbb{E}\Big(\big|\widetilde{d}_i - \mathbb{E}(\widetilde{d}_i)\big|^3\Big)\Big)^{1/3}}{\Big(\sum_{i\in P_\nu}\mathrm{Var}\big(\widetilde{d}_i\big)\Big)^{1/2}} \leq \frac{\big(\nu\overline{\gamma}^3/\underline{\rho}^3\big)^{1/3}}{\big(\nu\underline{\sigma}^2/\overline{\rho}^2\big)^{1/2}} = \nu^{-\frac{1}{6}}\frac{\overline{\gamma}\,\overline{\rho}}{\underline{\sigma}\,\underline{\rho}},$$

and the last term indeed converges to zero for $\nu \longrightarrow \infty$.

Under condition (C3), the claim follows from Berk's central limit theorem for m-dependent random variables, see [Ber73]. Translated into our context, this theorem is based on the following assumptions:

(A1) There is $m \in \mathbb{N}_0$ such that $\widetilde{d}_i$ and $\widetilde{d}_j$ are independent if $|i - j| > m$.

(A2) $\mathbb{E}(|\widetilde{d}_i|^3)$ is uniformly bounded for all i.

(A3) $\mathrm{Var}(\widetilde{d}_{i+1} + \ldots + \widetilde{d}_j) \leq (j - i)M$ for some $M \in \mathbb{R}_+$ and all i, j.

(A4) $\lim_{\nu\longrightarrow\infty}\nu^{-1}\mathrm{Var}(\sum_{i\in P_\nu}\widetilde{d}_i)$ exists and is nonzero.

By condition (C3), $\widetilde{\omega}_i$ and $\widetilde{\omega}_j$ are independent if the start times of the respective activities differ by at least T time units. For $i < j$, the start time difference between activities i and j amounts to at least $\sum_{l=i}^{j-1}\widetilde{d}_l \geq (j - i)\underline{\omega}/\overline{\rho}$. Thus, $m = \lceil (T\overline{\rho})/\underline{\omega}\rceil$ is sufficient to guarantee independence of $\widetilde{\omega}_i$ and $\widetilde{\omega}_j$ (and hence, of $\widetilde{d}_i$ and $\widetilde{d}_j$) whenever $|i - j| > m$, as required by (A1). Concerning (A2), we see that

$$0 \leq \mathbb{E}\Big(|\widetilde{d}_i|^3\Big) \leq \frac{1}{\underline{\rho}^3}\mathbb{E}\Big(\widetilde{\omega}_i^3\Big).$$

Since $\mathbb{E}(|\widetilde{\omega}_i - \mu_i|^3)$ is uniformly bounded for all i, so is $\mathbb{E}(\widetilde{\omega}_i^3)$. As for (A3), we note that

$$\begin{aligned}
\mathrm{Var}\big(\widetilde{d}_{i+1} + \ldots + \widetilde{d}_j\big) &= \sum_{p=i+1}^{j} \Big(\frac{1}{\rho_p^2} \mathrm{Var}(\widetilde{\omega}_p) + 2 \sum_{q=p+1}^{\min\{j,p+m\}} \frac{1}{\rho_p \rho_q} \mathrm{Cov}(\widetilde{\omega}_p, \widetilde{\omega}_q) \Big) \\
&\leq \frac{1}{\underline{\rho}^2} \sum_{p=i+1}^{j} \Big(\mathrm{Var}(\widetilde{\omega}_p) + 2 \sum_{q=p+1}^{\min\{j,p+m\}} \mathrm{Cov}(\widetilde{\omega}_p, \widetilde{\omega}_q) \Big) \\
&\leq \frac{1}{\underline{\rho}^2} \sum_{p=i+1}^{j} \big(\overline{\sigma}^2 + 2m\zeta \big) \leq (j-i)M
\end{aligned}$$

for $M = (\overline{\sigma}^2 + 2m\zeta)/\underline{\rho}^2$. Finally, (A4) directly follows from (C3). ∎

Condition (C3) is particularly appealing for project scheduling since typical sources of uncertainty (such as weather conditions, staff holidays and illness) tend to be of temporary nature. Apart from the requirement that the limit of $\nu^{-1}\mathrm{Var}(\sum_{i \in P_\nu} d_i(x_i; \widetilde{\omega}_i))$ for $\nu \longrightarrow \infty$ exists, the assumptions of (C3) are rather mild and do not require further explanation. Note that the aforementioned limit is likely to exist in all but pathological cases. It exists, for example, when the (co-)variances of dependent work contents can themselves be regarded as random variables with distributions that satisfy the assumptions of a central limit theorem. However, the limit does not exist, for example, if the task durations are independent random variables with variances

$$\mathrm{Var}\big(d_i(x_i; \widetilde{\omega}_i)\big) = \begin{cases} a & \text{if } i \in [2^{2k}, 2^{2k+1}) \text{ for some } k \in \mathbb{N}_0, \\ b & \text{if } i \in [2^{2k+1}, 2^{2k+2}) \text{ for some } k \in \mathbb{N}_0 \end{cases} \qquad \text{with } a < b.$$

Indeed, one can show that in this case

$$\frac{1}{\nu}\mathrm{Var}\Big(\sum_{i \in P_\nu} d_i(x_i; \widetilde{\omega}_i) \Big) \begin{cases} \leq \frac{5}{8}a + \frac{3}{8}b & \text{if } \nu = 2^{2k+1} - 1 \text{ for some } k \in \mathbb{N}_0, \\ \geq \frac{3}{8}a + \frac{5}{8}b & \text{if } \nu = 2^{2k+2} - 1 \text{ for some } k \in \mathbb{N}_0. \end{cases}$$

Due to the challenges involved in solving chance constrained problems directly, separated chance constraints are frequently approximated by conservative convex constraints that can be expressed in closed form, that is, without sampling. Such approximations are based on inequalities from probability theory [CSS07, NS06a]. In the following, we compare the quality of several such approximations with our approach. We consider a project path with five activities and a fixed resource allocation. Figure 4.3 illustrates the probability density functions of the activity durations. Note that we deliberately chose distributions that significantly deviate from normal distributions. Furthermore, a path with five activities is very short and hence seemingly unsuited for normal approximation. Figure 4.4 compares the error of several popular approximations with our approach for both independent and dependent activity durations. Unlike these approximations, our approach does not provide a conservative estimate of the path duration. However, it approximates the true cumulative distribution function significantly better than all other approximations considered. This might be surprising since normal approximations cannot be expected to correctly predict the tail probabilities of generic random variables. The reason for the high accuracy observed here is that project path durations are composite random variables whose components (*i.e.*, the activity durations) are typically of the same order of magnitude and follow smooth, close-to-unimodal distributions. Furthermore, although activity durations may exhibit interdependencies, durations of tasks that are well separated in time can essentially be regarded as independent. We remark that for the probabilities of interest (*i.e.*, $\alpha \geq 0.9$), the only reasonably tight bound is obtained by Chernoff's inequality. This inequality, however, requires complete knowledge about the moment generating function of the path duration. Compared to this, the normal approximation poses a very modest burden to the decision maker by requiring information about the first two moments of the work contents. Summing up, our preliminary conclusion (which will be supported by the numerical results in Section 4.4) is that normal approximations seem well suited to simplify the chance constraints appearing in (4.6b).

Under our normal approximation, the individual (pathwise) chance constraints in (4.6b) are replaced with

$$\Phi\left(\frac{\tau - \mu^\top \varrho_l(x)}{\sqrt{\varrho_l(x)^\top \Sigma \, \varrho_l(x)}}\right) \geq \beta_l \quad \iff \quad \tau \geq \mu^\top \varrho_l(x) + \Phi^{-1}(\beta_l)\sqrt{\varrho_l(x)^\top \Sigma \, \varrho_l(x)},$$

where Φ denotes the cumulative distribution function of the standard normal distribution. This

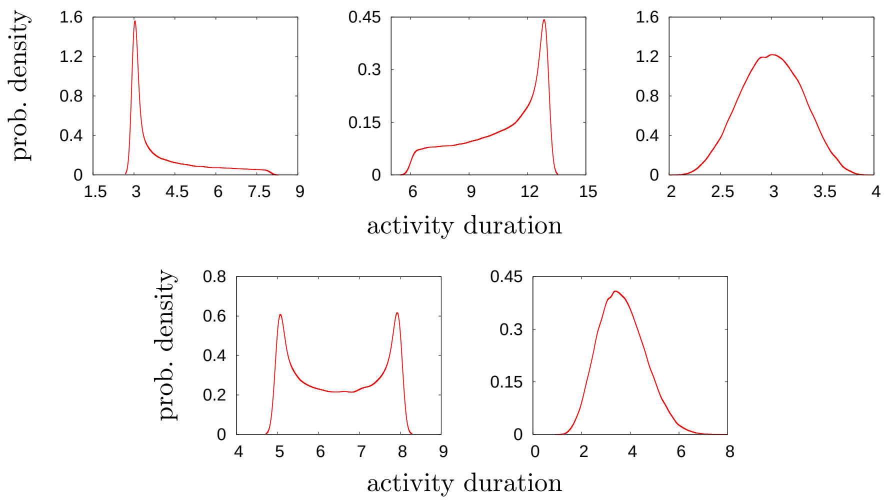


Figure 4.3: Probability density functions for five activity durations $\{\tilde{d}_i\}_{i=1}^5$. In the subsequent comparison, we use these durations both directly ('independent' durations) and as disturbances in a first-order autoregressive process $\{\tilde{d}'_i\}_{i=1}^5$ ('dependent' durations) where $\tilde{d}'_1 = \tilde{d}_1$ and $\tilde{d}'_i = 1/3\tilde{d}'_{i-1} + 2/3\tilde{d}_i$, $i = 2, \ldots, 5$.

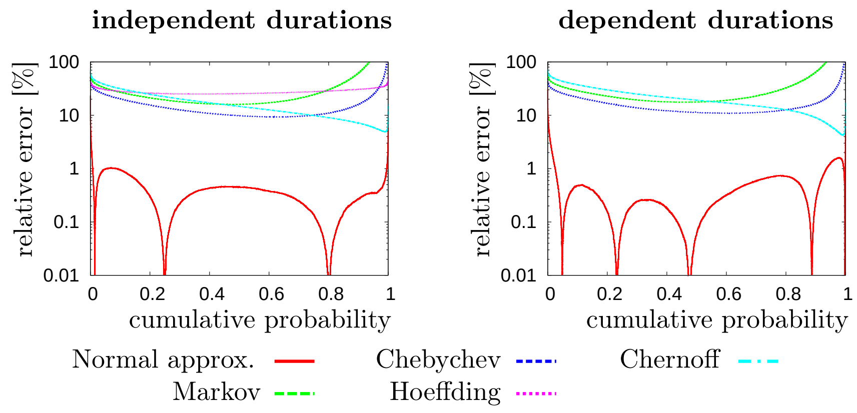


Figure 4.4: Approximation error of inequalities from probability theory and our normal approximation for independent (left) and dependent (right) activity durations. Chebychev's (single-sided) inequality and our normal approximation assume knowledge about the first two moments, Markov's inequality about the first moments and Hoeffding's inequality about the first moments and the supports of the activity durations. Chernoff's inequality, on the other hand, requires the specification of the complete moment generating function of the path duration. Note that Hoeffding's inequality requires independence among the activity durations.

leads us to the following approximation of (4.6):

$$\begin{aligned}
&\underset{x,\beta,\tau}{\text{minimise}} && \tau && \text{(4.7a)}\\
&\text{subject to} && x \in \mathbb{R}^{mn}_{+}, \quad \beta \in \mathbb{R}^{|\overline{\mathcal{P}}|}_{+}, \quad \tau \in \mathbb{R}_{+} && \\
& && \tau \geq \mu^{\top} \varrho_l(x) + \Phi^{-1}(\beta_l) \sqrt{\varrho_l(x)^{\top}\, \Sigma\, \varrho_l(x)} \qquad \forall\, P^l \in \overline{\mathcal{P}}, && \text{(4.7b)}\\
& && x \in X, \quad \beta \in B. && \text{(4.7c)}
\end{aligned}$$

The following example shows that model (4.7) is still generically nonconvex.

Example 4.3.2 *Consider again the project $G = (V, E)$ from Example 4.3.1, that is, $V = \{1, \ldots, 4\}$, $E = \{(1,2), (1,3), (2,4), (3,4)\}$, $\widetilde{\omega}_1 = \widetilde{\omega}_4 = 0$ almost surely and $\widetilde{\omega}_2$ and $\widetilde{\omega}_3$ follow independent standard normal distributions. One readily verifies that the set*

$$Y(\alpha) := \left\{(\tau, x_2, x_3) \in \mathbb{R}_+ \times [1/2, 2]^2 \ : \ \Phi(\tau x_2) + \Phi(\tau x_3) \geq \alpha + 1\right\}$$

is generically nonconvex. Indeed, for $\alpha = 2/3$, $(\tau^1, x_2^1, x_3^1) = (1, 2, 1/2)$ and $(\tau^2, x_2^2, x_3^2) = (1, 1/2, 2)$ are elements of $Y(2/3)$, but their convex combination $(\tau, x_2, x_3) = 1/2(\tau^1, x_2^1, x_3^1) + 1/2(\tau^2, x_2^2, x_3^2) = (1, 5/4, 5/4)$ is not part of $Y(2/3)$. However, the set $Y(\alpha)$ represents the projection of

$$Z(\alpha) := \left\{(\tau, x, \beta) \in \mathbb{R}_+ \times [1/2, 2]^4 \times B \ : \ \tau \geq \Phi^{-1}(\beta_1)/x_2, \ \tau \geq \Phi^{-1}(\beta_2)/x_3\right\}$$

onto (τ, x_2, x_3), and $Z(\alpha)$ equals the feasible region of (4.7) for this example. We conclude that (4.7) is generically nonconvex.

The next proposition further analyses the convexity of model (4.7).

Proposition 4.3.2 *Assume that $\alpha \geq 1/2$ and $\Sigma \geq 0$ component-wise, and let $(\widehat{x}, \widehat{\beta}, \widehat{\tau})$ be feasible in (4.7). With the additional constraints $x = \widehat{x}$ or $\beta = \widehat{\beta}$, (4.7) becomes a convex optimisation problem in (β, τ) or (x, τ), respectively.*

Proof First note that for $\alpha \geq 1/2$, the requirement $\beta \in B$ implies that $\beta_l \geq 1/2$, $P^l \in \overline{\mathcal{P}}$, in every feasible solution (x, β, τ). In the following, we will exploit the fact that Φ^{-1} is non-negative and convex on the interval $[1/2, 1]$.

We only need to investigate constraint (4.7b) since the other constraints and the objective function are clearly convex in (x, β, τ). For $x = \widehat{x}$ fixed, convexity of constraint (4.7b) in τ and β follows from the convexity of Φ^{-1} for $\beta_l \geq 1/2$. For $\beta = \widehat{\beta}$ fixed, on the other hand, we introduce auxiliary variables $y \in \mathbb{R}^n_+$ and $z \in \mathbb{R}^{n^2}_+$ with auxiliary constraints

$$y_i \, \rho(x_i) \geq \mu_i \quad \forall \, i \in V \quad \text{and} \quad z_{ij}^2 \, \rho_i(x_i) \, \rho_j(x_j) \geq \sigma_{ij} \quad \forall \, i, j \in V. \tag{4.7e}$$

Similar arguments as in Proposition 4.2.1 can be used to prove the convexity of constraints (4.7e). Note that the right-hand sides of these constraints are non-negative and hence, there are always variables y and z that satisfy (4.7e) as equalities. We replace constraint (4.7b) with

$$\tau \geq \sum_{i \in V} y_i + \Phi^{-1}(\widehat{\beta}_l) \sqrt{\sum_{i,j \in P^l} z_{ij}^2} \qquad \forall \, P^l \in \overline{\mathcal{P}}. \tag{4.7f}$$

Since the right-hand sides of (4.7f) are non-decreasing in y and z and we minimise the maximum of these right-hand sides, there is always an optimal solution to (4.7) that satisfies (4.7e) as equalities. Hence, (4.7e)–(4.7f) is indeed an equivalent reformulation of (4.7b). The first term on the right-hand side of (4.7f) is linear, while the second one represents a product of a non-negative scalar with the Frobenius norm of the matrix (z_{ij}). Both terms are manifestly convex. ∎

Since the activity durations are nonlinear functions of the decision variables, we need to require that the components of Σ are non-negative in order to guarantee convexity of (4.7) in (x, τ) for fixed β. Hence, we have to assume that the work contents of different activities have non-negative covariances, that is, all activity durations are either independent or positively correlated. This is not a very restrictive assumption when considering typical sources of uncertainty, such as motivational factors, staff availability, weather conditions and interactions between concurrent projects. It is rather unlikely that such a phenomenon increases the difficulty of some tasks but decreases the complexity of other tasks. In fact, it is a standard assumption in the literature that activity durations are independent [CSS07, DH02, HL05], which is a special case of our non-negativity assumption. An inspection of Proposition 4.3.2 reveals that if $\Sigma \ngeq 0$, replacing Σ with $\Sigma^+ = \left([\sigma_{ij}]^+\right)$, $[\sigma_{ij}]^+ = \max\{0, \sigma_{ij}\}$, results in a conservative approximation of (4.7). Hence, even if the assumption of non-negative correlations is violated, a reasonable surrogate problem that satisfies this assumption can be readily constructed.

Proposition 4.3.2 suggests a sequential convex optimisation scheme which optimises over (x, τ) and (β, τ) in turns, keeping either β or x fixed to the optimal value of the previous iteration. The following algorithm provides an outline of such a procedure.

Algorithm 4.1 Sequential convex optimisation procedure for model (4.7).

1. **Initialisation.** If $\underline{c} \notin X$, then abort: (4.7) is infeasible. Otherwise, set $x^0 = \underline{c}$, $\tau^0 = \infty$ (current objective value) and $t = 1$ (iteration counter).

2. **Optimisation over $(\boldsymbol{\beta}, \boldsymbol{\tau})$.** Solve problem (4.7) in (β, τ) with $x = x^{t-1}$ fixed. If the optimal solution (β^*, τ^*) satisfies $\tau^* < \tau^{t-1}$, then set $\beta^t = \beta^*$, otherwise keep $\beta^t = \beta^{t-1}$.

3. **Optimisation over $(\boldsymbol{x}, \boldsymbol{\tau})$.** Solve problem (4.7) in (x, τ) with $\beta = \beta^t$ fixed. If the optimal solution (x^*, τ^*) satisfies $\tau^* < \tau^{t-1}$, then set $x^t = x^*$, otherwise keep $x^t = x^{t-1}$. Set $\tau^t = \tau^*$.

4. **Termination.** If $(x^t, \beta^t) = (x^{t-1}, \beta^{t-1})$, then stop. Otherwise, set $t = t + 1$ and go back to Step 2.

Algorithm 4.1 is in the spirit of alternate convex search procedures. In the following, we discuss the main properties of this algorithm. For a more detailed study of alternate convex search procedures, see [KPK07]. We say that a feasible solution (x^*, β^*, τ^*) to (4.7) is a partial optimum if (x^*, τ^*) minimises (4.7) for $\beta = \beta^*$ fixed and (β^*, τ^*) minimises (4.7) for $x = x^*$ fixed. The following lemma shows that partial optimality is a necessary (but not sufficient) condition for local optimality.

Lemma 4.3.1 *For $\Sigma \geq 0$ component-wise, a local optimum (x^*, β^*, τ^*) of model (4.7) is a partial optimum.*

Proof Let (x^*, β^*, τ^*) be a local optimum. Then (x^*, τ^*) is a local optimum for $\beta = \beta^*$ fixed. Due to Proposition 4.3.2, (x^*, τ^*) is then a global minimiser of (4.7) for $\beta = \beta^*$ fixed. The same reasoning applies to (β^*, τ^*) if we fix x to x^*. Hence, (x^*, β^*, τ^*) satisfies the definition of a partial optimum. ∎

However, a partial optimum need not be locally optimal even for convex problems [KPK07]. The following proposition summarises the key properties of Algorithm 4.1.

Proposition 4.3.3 *Algorithm 4.1 identifies the (in-)feasibility of an instance of (4.7) in Step 1. For feasible instances, the following properties are satisfied:*

(P1) A different feasible solution is identified in every (but the last) iteration.

(P2) The objective values $\{(\tau^t)\}_t$ are monotonically decreasing and convergent.

(P3) If the algorithm terminates in finite time, then the final iterate is a partial optimum of (4.7). If the algorithm does not terminate, then every accumulation point of $\{(x^t, \beta^t, \tau^t)\}_t$ is a partial optimum of (4.7). Furthermore, all accumulation points have the same objective value.

Proof If $\underline{c} \in X$, then (x^0, β^0, τ^0) defined through $x^0 := \underline{c}$, $\beta^0_l := 1 - (1-\alpha)/\left|\overline{\mathcal{P}}\right|$ for $P^l \in \overline{\mathcal{P}}$ and

$$\tau^0 := \max_{P^l \in \overline{\mathcal{P}}} \left\{ \mu^\top \varrho_l(x^0) + \Phi^{-1}(\beta^0_l) \sqrt{\varrho_l(x^0)^\top \, \Sigma \, \varrho_l(x^0)} \right\}$$

constitutes a feasible solution to (4.7). If $\underline{c} \notin X$, on the other hand, then $X = \emptyset$. Thus, (4.7) is feasible if and only if $\underline{c} \in X$, and hence the algorithm correctly identifies the (in-)feasibility of a problem instance in Step 1. Furthermore, the algorithm determines a feasible solution in every iteration since $\tau^t = \tau^{t-1}$ together with $\beta^t = \beta^{t-1}$ and $x^t = x^{t-1}$ are feasible for $x^t = x^{t-1}$ and $\beta^t = \beta^{t-1}$ fixed, respectively.

Since the algorithm stops in Step 4 once $\tau^t \geq \tau^{t-1}$, $\{(\tau^t)\}_t$ is strictly monotonically decreasing until the penultimate iteration. Together with the feasibility of (x^t, β^t, τ^t) for all t, this proves (P1). Since the sequence $\{(\tau^t)\}_t$ is also bounded from below (for example by zero), assertion (P2) follows. If the algorithm terminates after finitely many iterations, then (P3) is satisfied by construction. Assume that the algorithm does not terminate. One can show that the algorithmic map of the procedure is closed [BSS06, KPK07]. This implies that $(x^{t+1}, \beta^{t+1}, \tau^{t+1})$ satisfies the termination criterion in Step 4 if we set (x^t, β^t, τ^t) to any accumulation point $(\widehat{x}, \widehat{\beta}, \widehat{\tau})$ of the sequence $\{(x^t, \beta^t, \tau^t)\}_t$. Hence, $(\widehat{x}, \widehat{\beta}, \widehat{\tau})$ satisfies that $(\widehat{x}, \widehat{\tau})$ is a minimiser of (4.7) for $\beta = \widehat{\beta}$ fixed and $(\widehat{\beta}, \widehat{\tau})$ is a minimiser of (4.7) for $x = \widehat{x}$ fixed. This, however, is just the definition of a partial optimum. ∎

For a given instance of (4.7) one can easily find finite a priori bounds on the problem variables that do not change the set of optimal solutions. In this case, the feasible set of (4.7) is compact and the constructed solution sequence contains accumulation points if Algorithm 4.1 does not terminate. We emphasise again that partial optima need not constitute local optima of (4.7). Note, however, that even the verification whether a particular solution to a biconvex problem is locally optimal is $\mathcal{NP}$-complete.[2] Thus, it seems justified to settle for the modest goal to find a partial optimum here.

Instead of employing an alternating search on x and β as outlined above, we can locally optimise over (x, β, τ). Note that in this case, the feasible region is generically nonconvex, and there is no guarantee that a local search procedure determines a local optimum or even a feasible solution to (4.7). In the next section, we will compare both solution approaches on a large set of problem instances.

We close with an example that illustrates model (4.7) and Algorithm 4.1.

Example 4.3.3 *Consider again the deterministic resource allocation problem described in Example 4.2.1. We now assume that the work content of each task $i \in V$ is a uniformly distributed*

[2]Indeed, a procedure that decides local optimality in bilinear problems can be used to verify local optimality in indefinite quadratic problems. The latter problem, however, is known to be $\mathcal{NP}$-complete [HPT00].

random variable $\widetilde{\omega}_i$ with support $[(1-\zeta)\omega_i, (1+\zeta)\omega_i]$, where ω_i denotes the nominal work content (taken from Example 4.2.1) and $\zeta = 0.2$. For ease of exposition, we assume that the work contents of different project activities are independent.

The expected value and variance of a uniform distribution with support $[(1-\zeta)\omega_i, (1+\zeta)\omega_i]$ is ω_i and $(\zeta\omega_i)^2/3$, respectively. We therefore have $\mu = (2,5,1,4,3,1)^\top$, while Σ is given by $\Sigma_{11} \approx 0.053$, $\Sigma_{22} \approx 0.333$, $\Sigma_{33} \approx 0.013$, $\Sigma_{44} \approx 0.213$, $\Sigma_{55} \approx 0.120$, $\Sigma_{66} \approx 0.013$ and $\Sigma_{ij} = 0$ for all $i \neq j$. The project in Example 4.2.1 has activity paths $\overline{\mathcal{P}} = \{P^1, P^2, P^3\}$ with $P^1 = \{1,2,4,6\}$, $P^2 = \{1,2,5,6\}$ and $P^3 = \{1,3,5,6\}$.

For $\alpha = 0.95$, model (4.7) reads as follows.

$$
\begin{aligned}
\underset{r,x,\beta,\tau}{\text{minimise}} \quad & \tau \\
\text{subject to} \quad & r \in \mathbb{R}^6_+, \quad x \in \mathbb{R}^{12}_+, \quad \beta \in \mathbb{R}^3_+, \quad \tau \in \mathbb{R}_+ \\
& \tau \geq 2r_1 + 5r_2 + 4r_4 + r_6 + \Phi^{-1}(\beta_1)\sqrt{0.053r_1^2 + 0.333r_2^2 + 0.213r_4^2 + 0.013r_6^2}, \\
& \tau \geq 2r_1 + 5r_2 + 3r_5 + r_6 + \Phi^{-1}(\beta_2)\sqrt{0.053r_1^2 + 0.333r_2^2 + 0.120r_5^2 + 0.013r_6^2}, \\
& \tau \geq 2r_1 + r_3 + 3r_5 + r_6 + \Phi^{-1}(\beta_3)\sqrt{0.053r_1^2 + 0.013r_3^2 + 0.120r_5^2 + 0.013r_6^2}, \\
& r_1\left(x_1^1\right)^2\left(x_1^2\right)^{3/2} \geq 1, \quad r_2\left(x_2^1\right)^2\left(x_2^2\right)^{3/2} \geq 1, \quad r_3\left(x_3^1\right)^2\left(x_3^2\right)^{3/2} \geq 1, \\
& r_5\left(x_5^1\right)^2\left(x_5^2\right)^{3/2} \geq 1, \quad r_6\left(x_6^1\right)^2\left(x_6^2\right)^{3/2} \geq 1, \quad 2r_4x_4^1 \geq 1, \quad r_4x_4^2 \geq 1, \\
& x^1, x^2 \in [(1/4)\mathrm{e}, 2\mathrm{e}], \quad \sum_{i=1}^{6} x_i^1 \leq 6, \quad \sum_{i=1}^{6} x_i^2 \leq 6, \\
& \beta_1 + \beta_2 + \beta_3 \geq 2.95, \quad \beta \in [0, \mathrm{e}].
\end{aligned}
$$

Table 4.1 documents the steps of Algorithm 4.1 when being applied to this instance. Note that apart from the increased objective value, the determined solution is very similar to the deterministic resource allocation found in Example 4.2.1. This is due to the fact that the variance of activity i's duration is chosen to be proportional to the expected value of i's duration.

x_1^1	x_1^2	x_2^1	x_2^2	x_3^1	x_3^2	x_4^1	x_4^2	x_5^1	x_5^2	x_6^1	x_6^2	β_1	β_2	β_3	τ
0.25	**0.25**	**0.25**	**0.25**	**0.25**	**0.25**	**0.25**	**0.25**	**0.25**	**0.25**	**0.25**	**0.25**	**n/a**	**n/a**	**n/a**	∞
0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	**≈ 1**	**0.950**	**≈ 1**	1483.41
1.20	**1.08**	**1.40**	**1.26**	**0.79**	**0.71**	**0.55**	**1.10**	**1.03**	**0.93**	**1.03**	**0.92**	≈ 1	0.950	≈ 1	8.72
1.20	1.08	1.40	1.26	0.79	0.71	0.55	1.10	1.03	0.93	1.03	0.92	**0.997**	**≈ 1**	**0.953**	8.38
1.21	**1.09**	**1.40**	**1.26**	**0.79**	**0.71**	**0.54**	**1.09**	**1.02**	**0.92**	**1.04**	**0.93**	0.997	≈ 1	0.953	8.37
1.21	1.09	1.40	1.26	0.79	0.71	0.54	1.09	1.02	0.92	1.04	0.93	**0.997**	**≈ 1**	**0.953**	8.37
1.21	**1.09**	**1.40**	**1.26**	**0.79**	**0.71**	**0.54**	**1.09**	**1.02**	**0.92**	**1.04**	**0.93**	0.997	≈ 1	0.953	8.37

Table 4.1: Application of Algorithm 4.1 to the project in Example 4.3.3. The first data row documents the initial solution determined in Step 1. The following rows present the intermediate solutions generated in three consecutive iterations of Steps 2 and 3. Variables printed in bold are updated in the respective step of the procedure. The algorithm terminates because the improvement of the objective value τ does not exceed a tolerance of 10^{-4}.

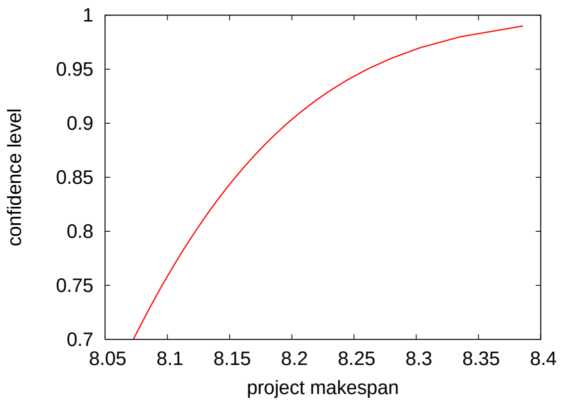


Figure 4.5: Impact of the confidence level α on the optimal solution to problem (4.7). For higher confidence levels, the estimated makespan increases disproportionally.

In general, this is not the case, and the determined resource allocations differ significantly. Figure 4.5 shows the impact of the confidence level α on the optimal solution to problem (4.7).

4.4 Numerical Results

In the following, we provide numerical results for the stochastic resource allocation problem (4.4). We do not consider the deterministic model (4.3) for two reasons. Firstly, (4.3) is a convex problem of moderate size and as such, it is clear that it can be solved efficiently even for large projects. Secondly, it is difficult to compare (4.3) with other deterministic models (as the ones discussed in the introduction) which rely on different assumptions.

This section is structured as follows. We start with a comparison of sequential convex and local optimisation for solving problem (4.7). We remind the reader that model (4.7) constitutes the approximation of the original resource allocation problem under uncertainty (4.4) obtained by separating the joint chance constraint and approximating the path durations via normal distributions. Afterwards, we compare model (4.7) with alternative approaches to solve problem (4.4). All numerical results are based on randomly generated projects with n activities and $2n$ precedences. The project graphs are constructed with a variant of the deletion method presented in [DDH93]. The work contents follow independent normal or Beta distributions with randomly selected parameters. All instances involve two resources, and resource consumption is limited to a third of the sum of upper investment bounds. The activity types (*i.e.*, substitutional or limitational) and the parameters $(\underline{c}_i, \overline{c}_i)$, ψ_i, δ_i and γ_i are also chosen randomly. Throughout this section, our goal is to find a resource allocation that minimises the 0.95-VaR of the uncertain project makespan. All results in this section were obtained with the freely available optimisation package Ipopt.[3]

In Section 4.3 we proposed two alternative methods for solving problem (4.7): sequential convex optimisation (Algorithm 4.1) determines a partial optimum by solving a series of convex optimisation problems, whereas a local search procedure jointly optimises over all problem variables. Since problem (4.7) is generically nonconvex (see Example 4.3.2), neither approach is guaranteed to provide globally optimal solutions. More so, the local search procedure cannot even guarantee to provide a feasible solution. Table 4.2 compares both approaches on a set of test instances with normally distributed work contents. The quality of the resulting approximate solutions is measured relative to the true global optima, which we determine exactly for these small problem instances ($n \leq 20$) by means of a branch-and-bound algorithm [HPT00]. Table 4.2 reveals that the local search procedure found global optima in all test cases. Although this procedure is more likely to fail on larger problems, it turns out to be very reliable on all considered instances. For sequential convex optimisation, we provide the results for a single trial ('1x') and several multi-start ('10x' and '100x') versions. As expected, repeating the search with different start points leads to better solutions. Although sequential convex optimisation manages to find good solutions, it is clearly outperformed by the local search procedure. Thus, we will employ the local search procedure in all subsequent tests.

[3]Ipopt homepage: `https://projects.coin-or.org/Ipopt`.

		instance size n 5	10	15	20
Local search procedure	*# trials*	*1.00*	*1.03*	*1.04*	1.05
	suboptimality	0.00%	0.00%	0.00%	0.00%
Sequential convex optimisation	*# iterations*	*3.00*	*3.09*	*3.25*	*3.98*
	suboptimality (1x)	0.74%	2.96%	3.79%	3.61%
	suboptimality (10x)	0.36%	1.55%	2.75%	2.79%
	suboptimality (100x)	0.01%	0.86%	1.94%	2.07%

Table 4.2: Comparison of sequential convex and local optimisation. The table provides the number of optimisation runs required to determine a feasible solution (for the local search procedure) and the number of iterations required to determine a partial optimum (for the sequential convex optimisation algorithm). Furthermore, the relative suboptimality of the solution is given for both approaches. All results represent average values over 100 randomly generated test instances.

In the remainder of this section, we compare our model (4.7) with three alternative approaches to approximate the original problem (4.4): a nominal problem formulation, a convex approximation via CVaR constraints and a formulation based on robust optimisation. In the nominal problem formulation, the uncertain work contents are replaced with their expected values. The resulting model is a deterministic resource allocation problem of type (4.3). This approach is very simple, but it completely ignores the risk inherent to the chosen resource allocation. Nevertheless, nominal formulations are very popular in both theory and practise, and they allow us to quantify the benefits of an honest treatment of uncertainty. As for the CVaR approximation, we replace the joint chance constraint (4.4b) by a related CVaR constraint, which results in a conservative approximation [RU00]. Although the CVaR constraint does not require enumeration of the activity paths, it has no closed-form representation, and we need to employ scenario approximation techniques. In our tests, we approximate the CVaR via 1,000, 2,500 and 5,000 scenarios and a Benders decomposition scheme [Pré95]. As for the approximation based on robust optimisation, finally, we use the approach presented in [CSS07]. Since the activity durations fail to be conic functions of the resource investments (in the sense of [CSS07]), we need to enumerate the activity paths in a similar manner as in model (4.7). Contrary to the other formulations, the robust optimisation approach is only applicable in the presence of Beta-distributed work contents. This is due to the fact that robust optimisation requires all random variables to possess bounded supports. For a further discussion on this topic and possible remedies, see [CSS07].

Our comparison proceeds in two steps. First, we consider instances with normally distributed work contents. It follows from Section 4.3 that in this case model (4.7) provides a conservative approximation of the VaR. Afterwards, we compare the formulations on instances with Beta-distributed work contents. In this case, model (4.7) does not provide a conservative approximation anymore.

Figure 4.6 and Table 4.3 summarise the results for normally distributed work contents. The ‘prediction error’ denotes the relative difference between the a priori 0.95-VaR implied by the solutions of the respective optimisation models and the a posteriori 0.95-VaR determined by Monte Carlo sampling. We also compare the solutions obtained from the various models in terms of their 0.95-VaR and average makespan, again using Monte Carlo sampling. Both the 0.95-VaR and the average makespan are measured relative to the solution to model (4.7).

The results reveal that the nominal problem grossly underestimates the makespan. This is caused by two factors. Firstly, the nominal problem considers the expected makespan, which is in most cases significantly smaller than the 0.95-VaR. Secondly, by interchanging maximum and expectation operators in the problem formulation, the nominal model underestimates the expected makespan due to Jensen’s inequality [DH02]. This underestimation leads to substantial prediction errors and a poor performance of the resulting resource allocations. Indeed, nominal solutions are only acceptable for very large projects, where the assumption of independent work contents makes it increasingly unlikely that the project duration differs significantly from the expected makespan. Note that model (4.7) and the CVaR approximations perform more or less equally well on the considered test instances.

Table 4.4 compares the computational requirements of the considered approaches. Problem (4.7) needs to be solved only once, but it can involve a large number of activity paths. The table shows, however, that the number of paths remains moderate even for large instances. The CVaR approximations, on the other hand, require the repeated solution of certain Benders subproblems. It turns out that the number of subproblems increases rapidly with the project size. Thus, the CVaR approximations require substantially more computing resources than the (local) solution of formulation (4.7). This becomes particularly important if some of the considered project resources in model (4.4) are discrete. Note that the nominal model is a deterministic resource allocation problem of type (4.3) and can hence be solved efficiently for all considered sizes.

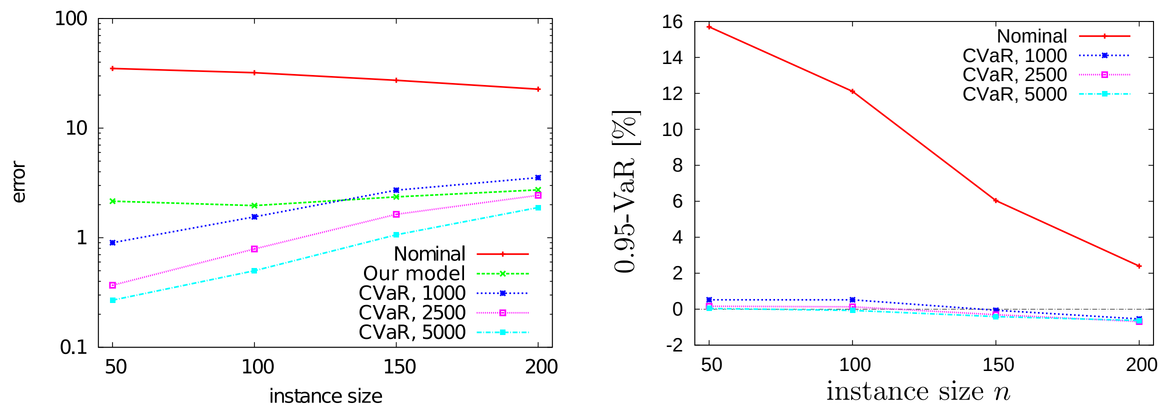


Figure 4.6: Comparison of model (4.7) with alternative problem formulations for normally distributed work contents. The left graph shows the relative difference between the estimated and exact 0.95-VaR. The right graph relates the 0.95-VaR of the solutions determined by the alternative formulations to the one obtained from solving model (4.7). All results represent average values over 100 randomly generated test instances.

Figure 4.7 and Table 4.5 summarise the results for Beta-distributed work contents. In this setting, the nominal problem performs even worse than before: both the prediction errors and the 0.95-VaR have deteriorated. Again, model (4.7) and the CVaR approximations perform more of less equally well. It becomes apparent that robust optimisation leads to large prediction errors. In contrast to the nominal problem, however, robust optimisation *over*estimates the 0.95-VaR. Although the obtained resource allocations are better than the nominal solutions, they are still substantially worse than the allocations obtained from model (4.7) and the CVaR approximations. Table 4.6 compares the computational requirements of model (4.7) and the CVaR approximations. The results are similar to those of Table 4.4, although the CVaR approximations require slightly more cuts than before. Note that the computational requirements for solving model (4.7) and the robust optimisation problem are roughly similar since both formulations scale with the number of activity paths.

In conclusion, although model (4.7) is nonconvex, it seems very well-behaved: in our numerical tests, the model could be solved efficiently and reliably by standard local optimisation tech-

		instance size n			
		50	100	150	200
Our model	*prediction error*	*2.15%*	*1.96%*	*2.36%*	*2.73%*
Nominal	*prediction error*	*35.02%*	*32.05%*	*27.34%*	*22.68%*
	0.95-VaR	+15.70%	+12.11%	+6.04%	+2.40%
	average makespan	+8.45%	+6.03%	+1.07%	-1.44%
CVaR (1000)	*prediction error*	*0.90%*	*1.55%*	*2.71%*	*3.53%*
	0.95-VaR	+0.52%	+0.52%	-0.06%	-0.54%
	average makespan	+0.55%	+0.42%	-0.36%	-0.37%
CVaR (2500)	*prediction error*	*0.37%*	*0.79%*	*1.63%*	*2.44%*
	0.95-VaR	+0.16%	+0.13%	-0.30%	-0.69%
	average makespan	+0.05%	+0.16%	-0.51%	-0.60%
CVaR (5000)	*prediction error*	*0.27%*	*0.50%*	*1.06%*	*1.88%*
	0.95-VaR	+0.04%	-0.07%	-0.42%	-0.63%
	average makespan	-0.14%	-0.09%	-0.71%	-0.58%

Table 4.3: Comparison of model (4.7) with alternative problem formulations for normally distributed work contents. ‘Prediction error’ refers to the relative difference between the estimated and exact 0.95-VaR. The 0.95-VaR and average makespan are measured relative to the optimal solution to model (4.7). All results represent average values over 100 randomly generated test instances.

	instance size n			
	50	100	150	200
Our model	46.34	115.77	189.35	270.96
CVaR (1000)	139.53	283.54	521.78	692.63
CVaR (2500)	141.41	294.06	537.50	678.95
CVaR (5000)	142.56	284.00	522.22	708.35

Table 4.4: Computational requirements of the various problem formulations for normally distributed work contents. For model (4.7), the table documents the cardinality of $\overline{\mathcal{P}}$. For the CVaR approximations, the table provides the number of introduced Benders cuts. All results represent average values over 100 randomly generated test instances.

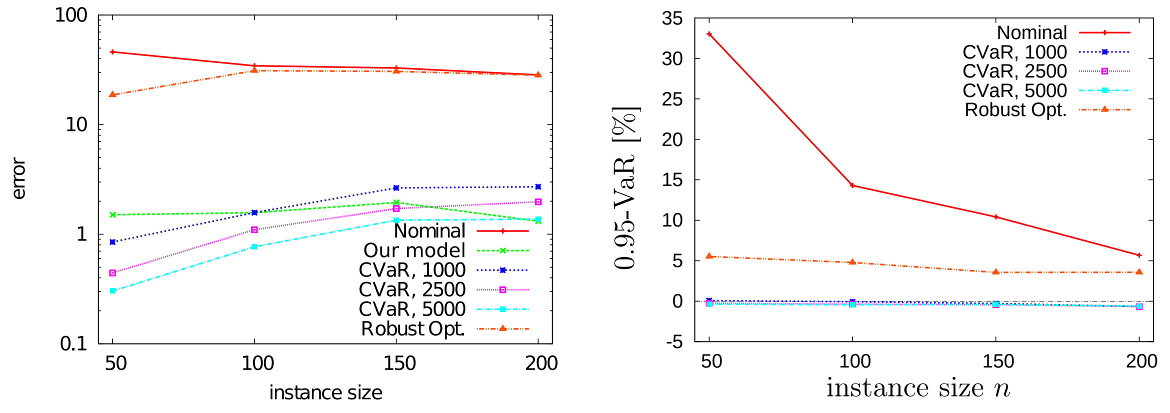


Figure 4.7: Comparison of model (4.7) with alternative problem formulations for Beta-distributed work contents. See Figure 4.6 for further explanations.

		instance size n			
		50	100	150	200
Our model	*prediction error*	*1.51%*	*1.57%*	*1.95%*	*1.31%*
Nominal	*prediction error*	*46.11%*	*34.45%*	*32.92%*	*28.43%*
	0.95-VaR	+33.02%	+14.29%	+10.42%	+5.68%
	average makespan	+15.28%	+5.70%	+2.32%	-0.42%
CVaR (1000)	*prediction error*	*0.85%*	*1.57%*	*2.64%*	*2.71%*
	0.95-VaR	+0.08%	-0.06%	-0.28%	-0.66%
	average makespan	+1.59%	+0.74%	+0.23%	-0.39%
CVaR (2500)	*prediction error*	*0.44%*	*1.10%*	*1.71%*	*1.97%*
	0.95-VaR	-0.25%	-0.37%	-0.44%	-0.67%
	average makespan	+1.19%	+0.42%	+0.17%	-0.10%
CVaR (5000)	*prediction error*	*0.30%*	*0.77%*	*1.35%*	*1.37%*
	0.95-VaR	-0.37%	-0.44%	-0.40%	-0.60%
	average makespan	+1.02%	+0.31%	+0.11%	+0.07%
Robust Optimisation	*prediction error*	*18.70%*	*31.10%*	*30.61%*	*28.26%*
	0.95-VaR	+5.53%	+4.78%	+3.56%	+3.57%
	average makespan	+7.91%	+4.28%	+3.14%	+2.35%

Table 4.5: Comparison of model (4.7) with alternative problem formulations for Beta-distributed work contents. See Table 4.3 for further explanations.

	instance size n			
	50	100	150	200
Our model	46.26	114.87	190.58	266.43
CVaR (1000)	139.43	319.78	574.38	771.26
CVaR (2500)	139.14	318.89	587.91	808.19
CVaR (5000)	140.41	318.62	582.09	794.38

Table 4.6: Computational requirements of the various problem formulations for Beta-distributed work contents. See Table 4.4 for further explanations.

niques. Furthermore, the solution quality is comparable to that obtained by convex CVaR approximations of the original model (4.4), even though model (4.7) requires significantly fewer computational resources. The nominal problem and the approximation based on robust optimisation can both be solved very efficiently, but they lead to poor makespan estimates and thus suggest severely suboptimal resource allocations.

4.5 Extensions

In this section, we first illustrate how one can robustify model (4.7) against uncertainty in the first and second moments of the work contents. Afterwards, we present an iterative solution procedure for model (4.7) which applies to projects with large numbers of activity paths. In the following, we abbreviate the β_l-quantile of the duration of activity path $P^l \in \overline{\mathcal{P}}$ by

$$q_l(x, \beta_l; \mu, \Sigma) := \mu^\top \varrho_l(x) + \Phi^{-1}(\beta_l) \sqrt{\varrho_l(x)^\top \, \Sigma \, \varrho_l(x)}.$$

4.5.1 Moment Ambiguity

The stochastic resource allocation model (4.7) minimises the α-VaR of the project makespan and therefore hedges against the uncertainty underlying the factual work contents. The model assumes rather detailed knowledge about the nature of this uncertainty, though, since it requires precise specification of its first two moments. Here, we relax this assumption and require instead that these moments are merely known to be contained in the set $\mathcal{U} = \mathcal{U}_\mu \times \mathcal{U}_\Sigma$ with

$$\begin{aligned} \mathcal{U}_\mu &= \left\{ \mu \in \mathbb{R}^n_+ \, : \, \mu = \mu_0 + w_\mu \bullet \widehat{\mu}, \, \left\| \widehat{\mu} \right\|_2 \leq 1, \, \widehat{\mu} \in \mathbb{R}^n \right\} \\ \text{and} \quad \mathcal{U}_\Sigma &= \left\{ \Sigma \in \mathbb{S}^n_+ \, : \, \Sigma = \Sigma_0 + W_\Sigma \bullet \widehat{\Sigma}, \, \left\| \widehat{\Sigma} \right\|_2 \leq 1, \, \widehat{\Sigma} \in \mathbb{S}^n_+ \right\}, \end{aligned}$$

where $\mu_0 \in \mathbb{R}^n_+$ and $\Sigma_0 \in \mathbb{S}^n_+$. The operator '$\bullet$' denotes the element-wise (Hadamard) product, while $\mathbb{S}^n_+$ denotes the subspace of symmetric and positive semidefinite matrices in $\mathbb{R}^{n \times n}$. The parameters μ_0 and Σ_0 can be interpreted as nominal values, while $w_\mu \in \mathbb{R}^n_+$ and $W_\Sigma \in \mathbb{R}^{n \times n}_+$

represent ‘degrees of ambiguity’.

In the spirit of robust optimisation [BTN98, BS06], our goal is to minimise the worst-case α-VaR of the project makespan under the assumption that the true moments (μ, Σ) can be any element of $\mathcal{U}$. This can be expressed as

$$\min_{\substack{x \in X, \\ \beta \in B}} \max_{P^l \in \overline{\mathcal{P}}} \underbrace{\max_{(\mu,\Sigma) \in \mathcal{U}} q_l(x, \beta_l; \mu, \Sigma)}_{\varphi_l(x,\beta_l)} .$$

Due to the separability of $\mathcal{U}$ with respect to the first and second moments, the worst-case α-VaR is representable as

$$\begin{aligned} \varphi_l(x, \beta_l) &= \max_{(\mu,\Sigma) \in \mathcal{U}} \left\{ \mu^\top \varrho_l(x) + \Phi^{-1}(\beta_l) \sqrt{\varrho_l(x)^\top \, \Sigma \, \varrho_l(x)} \right\} \\ &= \max_{\mu \in \mathcal{U}_\mu} \left\{ \mu^\top \varrho_l(x) \right\} + \Phi^{-1}(\beta_l) \max_{\Sigma \in \mathcal{U}_\Sigma} \left\{ \sqrt{\varrho_l(x)^\top \, \Sigma \, \varrho_l(x)} \right\}. \end{aligned}$$

The first maximisation term reduces to

$$\begin{aligned} \max_{\mu \in \mathcal{U}_\mu} \left\{ \mu^\top \varrho_l(x) \right\} &= (\mu_0)^\top \varrho_l(x) + \max_{\substack{\|\widehat{\mu}\|_2 \leq 1, \\ \mu_0 + \widehat{\mu} \geq 0}} \left(w_\mu \bullet \widehat{\mu} \right)^\top \varrho_l(x) \\ &= (\mu_0)^\top \varrho_l(x) + \max_{\substack{\|\widehat{\mu}\|_2 \leq 1, \\ \mu_0 + \widehat{\mu} \geq 0}} \widehat{\mu}^\top \left[w_\mu \bullet \varrho_l(x) \right] \\ &= (\mu_0)^\top \varrho_l(x) + \left\| w_\mu \bullet \varrho_l(x) \right\|_2 . \end{aligned}$$

Concerning the last identity, note that all components of $w_\mu \bullet \varrho_l(x)$ are non-negative, and hence $\widehat{\mu} \geq 0$ and $\mu_0 + \widehat{\mu} \geq 0$ are vacuously satisfied at optimality. By applying similar transformations as described in Proposition 4.3.2, the last term can be expressed by convex constraints.

Similarly, one can show that the Σ-term reduces to

$$\max_{\Sigma \in \mathcal{U}_\Sigma} \left\{ \sqrt{\varrho_l(x)^\top \, \Sigma \, \varrho_l(x)} \right\} = \sqrt{\varrho_l(x)^\top \, \Sigma_0 \varrho_l(x) + \left\| W_\Sigma \bullet \left[\varrho_l(x) \varrho_l(x)^\top \right] \right\|_2}.$$

For $\Sigma_0 \succeq 0$ (see Section 4.3), the latter term can be expressed by convex constraints, too. Hence, the convexity properties of model (4.7) are preserved when the moments of the work contents are ambiguous, and the increase in model size is moderate. This contrasts with the optimisation of CVaR under distributional ambiguity, which is considerably more involved [PW07, ZF09].

4.5.2 Iterative Path Selection Procedure

The runtime behaviour of the stochastic resource allocation model (4.7) depends on the number of activity paths in $\overline{\mathcal{P}}$. Recall that $\overline{\mathcal{P}}$ has been of moderate size in all of our numerical tests (see Section 4.4). However, it is well-known that in the worst-case the number of activity paths can be exponential in the size of the project graph. In this section, we propose an iterative solution procedure for (4.7) based on the principles of semi-infinite programming. The outline of the procedure is as follows.

Algorithm 4.2 Iterative path selection procedure for model (4.7).

1. Initialise $\mathcal{L}$ as a (nonempty) subset of $\overline{\mathcal{P}}$. Choose $\epsilon \in (0, (1-\alpha)/\left|\overline{\mathcal{P}} \setminus \mathcal{L}\right|)$.

2. Determine a feasible (possibly suboptimal) solution (x^*, β^*) to

$$\begin{array}{ll} \underset{x,\beta}{\text{minimise}} & \max\limits_{P^l \in \mathcal{L}} q_l(x, \beta_l; \mu, \Sigma) \\ \text{subject to} & x \in \mathbb{R}^{mn}_+, \quad \beta \in \mathbb{R}^{|\mathcal{L}|}_+ \\ & \sum\limits_{P^l \in \mathcal{L}} \beta_l \geq \alpha + (\left|\overline{\mathcal{P}}\right| - 1) - \left|\overline{\mathcal{P}} \setminus \mathcal{L}\right| (1 - \epsilon), \\ & x \in X, \;\; \beta \in [0, \mathrm{e}] \,. \end{array}$$

   Let τ^* denote the resulting objective value.

3. Check whether there is a path $P^s \in \overline{\mathcal{P}} \setminus \mathcal{L}$ with $q_s(x^*, 1-\epsilon; \mu, \Sigma) > \tau^*$. If this is the case, then add one such path to $\mathcal{L}$ and return to Step 2. Otherwise, stop: x^* represents the best resource allocation found.

Step 1 initialises $\mathcal{L}$, the subset of activity paths $P^l \in \overline{\mathcal{P}}$ currently considered. It also assigns a value to ϵ, the probability assigned to paths in $\overline{\mathcal{P}} \setminus \mathcal{L}$ not (yet) considered. In Step 2, model

(4.7) is solved for the activity paths $P^l \in \mathcal{L}$. Note that the first constraint implicitly assigns a probability of $1-\epsilon$ to every path in $\overline{\mathcal{P}} \setminus \mathcal{L}$. Step 3 checks whether there is a path in $\overline{\mathcal{P}} \setminus \mathcal{L}$ whose $(1-\epsilon)$-duration quantile exceeds the α-VaR determined in the previous step. If this is the case, then one such path is added to $\mathcal{L}$, and the procedure iterates. Otherwise, the procedure terminates. We will present a strategy to determine a suitable path in $\overline{\mathcal{P}} \setminus \mathcal{L}$ below.

If the subproblem in Step 2 is infeasible, then $X = \emptyset$ and (4.7) does not possess a feasible solution. For any $\epsilon \in (0, (1-\alpha)/\left|\overline{\mathcal{P}} \setminus \mathcal{L}\right|)$, the final resource allocation obtained by Algorithm 4.2 is feasible in (4.7), and τ^* represents a conservative estimate of its objective value. Note that for fixed ϵ, only a near-optimal solution is determined if $\mathcal{L} \neq \overline{\mathcal{P}}$ at termination. This statement is true even if the subproblems arising in Step 2 are solved to global optimality. Indeed, a better 'probability arrangement' can potentially be obtained by assigning $\beta_l > 1-\epsilon$ to paths $P^l \in \overline{\mathcal{P}} \setminus \mathcal{L}$. This is not restrictive for practical applications, however, since optimisation algorithms typically require an upper bound strictly below 1 for β_l, $P^l \in \overline{\mathcal{P}}$, anyway (since $\Phi(\beta_l) \longrightarrow \infty$ for $\beta_l \longrightarrow 1$). For any given value of ϵ, let $x(\epsilon)$ and $f(\epsilon)$ denote any final resource allocation and its objective value, respectively, that are determined by Algorithm 4.2 when solving the subproblems to global optimality. One can show that the sequence $\{f(\epsilon)\}_{\epsilon \longrightarrow 0}$ converges monotonically to the optimal objective value of (4.7). Furthermore, every accumulation point of $\{x(\epsilon)\}_{\epsilon \longrightarrow 0}$ constitutes a globally optimal resource allocation for (4.7).

Note that in the third step, we have to examine a potentially large number of paths $P^s \in \overline{\mathcal{P}} \setminus \mathcal{L}$. We can obtain an upper bound on the $(1-\epsilon)$-duration quantile of path $P^s \in \overline{\mathcal{P}} \setminus \mathcal{L}$ as follows.

$$\begin{aligned}
&q_s(x^*, 1-\epsilon; \mu, \Sigma) = \mu^\top \varrho_s(x^*) + \Phi^{-1}(1-\epsilon)\sqrt{\varrho_s(x^*)^\top \Sigma\, \varrho_s(x^*)} \\
&\leq \mu^\top \varrho_s(x^*) + \Phi^{-1}(1-\epsilon)\sqrt{\sum_{i \in P^s} \eta_i} \quad \text{with} \quad \eta_i = \left[\max_{\substack{P^l \in \overline{\mathcal{P}}: \\ i \in P^l}} \sum_{j \in P^l} \sigma_{ij} / \left[\rho_i(x_i^*)\rho_j(x_j^*)\right] \right]^+ \\
&\leq \sum_{i \in P^s} \phi_i \quad \text{with} \quad \phi_i = \mu_i / \rho_i(x_i^*) + \Phi^{-1}(1-\epsilon)\sqrt{\eta_i}.
\end{aligned}$$

Here, we use the abbreviation $[\cdot]^+ := \max\{0, \cdot\}$. The first inequality holds because $i \in P^s$ implies that $P^s \in \left\{P^l \in \overline{\mathcal{P}} : i \in P^l\right\}$. The second inequality follows from the fact that the 2-norm of a vector is no larger than its 1-norm and $\eta_i \geq 0$. Note that η_i (and hence, ϕ_i) can be determined in polynomial time (relative to the size of the project graph) for all $i \in V$. The described upper bound allows us to construct a deterministic project with durations ϕ_i for $i \in V$. Every path duration in this project yields an upper bound on the $(1-\epsilon)$-duration quantile of the respective path in model (4.7). Thus, we can use techniques for determining the κ largest paths in a directed, acyclic graph to obtain candidates paths $P^s \in \overline{\mathcal{P}} \setminus \mathcal{L}$ for inclusion in $\mathcal{L}$. In particular, we can stop our search in Step 3 once we have examined all paths $P^s \in \overline{\mathcal{P}} \setminus \mathcal{L}$ with $\sum_{i \in P^s} \phi_i > \tau^*$. A method for determining the κ largest paths of a project graph $G = (V, E)$ in time $\mathcal{O}(|E| + \kappa)$ is presented in [Epp94].

We will refine Algorithm 4.2 in the next chapter, where we use a variant of this algorithm to generate convergent lower bounds on the optimal objective value of two-stage robust resource allocation problem. In that chapter, we will also provide a numerical example of the algorithm.

4.6 Conclusion

Resource allocation problems constitute a vital class of project scheduling problems. To the best of our knowledge, this chapter proposes the first deterministic multi-resource allocation model that is convex and hence tractable for realistic problem sizes. Resource allocation models have to stipulate functional relations between resource investments and task durations. We employed production functions from microeconomic theory, which lead to intuitively appealing duration functions that are amenable to economic interpretation.

In a second step, we extended our model to accommodate uncertainty. Our formulation assumes knowledge about the first two moments of the uncertain parameters and optimises the α-VaR of the project makespan. Although VaR is a nonconvex risk measure, we showed that the specific properties of project scheduling problems enable us to approximately optimise it very efficiently. Furthermore, the resulting model readily accommodates distributional ambiguity. This is crucial in project scheduling, because the moments of the uncertain parameters are often unknown due to the lack of historical data.

While the proposed resource allocation model seems to be primarily suitable for project scheduling problems, the VaR approximation readily applies to other application areas of temporal networks such as process scheduling [JM99] and digital circuit design [KBY$^+$07]. It would therefore be instructive to apply variants of our approximate problem formulation (4.7) to models in these application areas as well.

Chapter 5

Minimisation of the Worst-Case Makespan

5.1 Introduction

In this chapter we study a robust resource allocation problem that minimises the worst-case makespan. As in the previous chapter, we assume that the resource allocation is a here-and-now decision, whereas the task start times are modelled as a wait-and-see decision [RS03], which may depend on random parameters affecting the task durations. In the terminology of Section 2.2.2, we therefore study a two-stage robust optimisation problem. In contrast to its stochastic counterpart, the complexity of the robust resource allocation problem is unknown [Hag88]. All existing solution approaches have in common that they determine suboptimal solutions without bounding the incurred optimality gap. In this chapter, we show that the robust resource allocation problem is $\mathcal{NP}$-hard, which explains the lack of exact solution approaches in the literature. We then develop two hierarchies of approximate problems that provide convergent lower and upper bounds on the optimal value of the original problem. The upper bounds correspond to feasible allocations whose objective values are bracketed by the bounds. Hence, we obtain a sequence of feasible allocations that are asymptotically optimal and whose optimality gaps can be quantified at any time.

There are three robust resource allocation problems in temporal networks that directly relate to our problem. A production scheduling problem that minimises the worst-case makespan under uncertain processing times, product demands and market prices is proposed in [JLF07, LJF04]. The decision maker can influence the makespan by choosing a processing sequence and assigning resources to the individual processing steps, and the optimal process start times are approximated by constant decision rules. A robust variant of the time/cost trade-off problem in project scheduling is discussed in [CSS07]. Assuming that the durations of the project activities are uncertain, this model determines a resource allocation that minimises the worst-case makespan. To obtain a tractable optimisation problem, the optimal task start times are approximated by affine decision rules. A related time/cost trade-off problem is studied in [CGS07], where the resource allocation for a specific activity is allowed to adapt to all uncertain parameters that have been observed until the respective task start time. Affine decision rules are used to obtain a tractable approximation for the problem. We review decision rules in Section 5.2.2.

Research in the wider area of robust network optimisation started with the seminal paper [BS03], which develops solution techniques for single-stage robust network flow problems. In recent years, several two-stage robust network optimisation problems have been solved under the name of recoverable robust optimisation. In [LLMS09], a railway scheduling problem is considered which selects a here-and-now timetable that can be made feasible for a range of train delays in the second stage. A two-stage robust freight transportation problem is studied in [EMS09]. This model determines a here-and-now repositioning plan for empty containers that can be recovered for a range of supply and demand scenarios in the second stage. In both papers, tractable optimisation problems are derived through carefully chosen problem reformulations. In [AZ07, OZ07], a two-stage robust network optimisation problem is proposed which treats the network design as a here-and-now decision, while the network flows are modelled as wait-and-see decisions which are chosen after the uncertain parameters have been observed. Recently, approximation algorithms have been developed for two-stage robust combinatorial problems [FJMM07, KKMS08]. Here, a feasible solution to the combinatorial problem has to be found for any possible realisation of the random parameters. Since the second stage decision incurs a higher cost, there is a trade-off between over-protection in the first stage and a costly recovery in the second stage. Finally, there is an extensive literature on network problems that optimise the worst-case regret, see [Ave01].

The remainder of this chapter is organised as follows. In the next section, we define the robust resource allocation problem. After a review of popular approximations for the problem, we show that the robust resource allocation problem is generically $\mathcal{NP}$-hard. In Section 5.3 we discuss a path-wise formulation that provides the basis for our solution technique. In Sections 5.4 and 5.5 we develop families of optimisation problems that provide convergent lower and upper bounds, respectively. Section 5.6 presents the results of a numerical evaluation on randomly generated test instances, and Section 5.7 applies our bounding scheme to VLSI design. We conclude in Section 5.8.

In addition to the notation introduced in Section 1.3, this chapter uses the following convention. For a set $A \subseteq \{1, \ldots, n\}$, we denote by $\mathbb{I}_A$ the n-dimensional vector with $(\mathbb{I}_A)_i = 1$ if $i \in A$ and $(\mathbb{I}_A)_i = 0$ otherwise. As we will see shortly, this allows us to express the sum of task durations on a network path $P \subseteq V$ as the inner product between the indicator vector $\mathbb{I}_P$ of the path and the vector of all task durations.

5.2 Robust Resource Allocations

We first define the robust resource allocation problem that we consider in this chapter. We then review how decision rules can be applied to obtain a tractable approximation for this problem. In Section 5.2.3 we analyse the complexity of the robust resource allocation problem.

5.2.1 The Robust Resource Allocation Problem

We assume that the structure of the temporal network (*i.e.*, V and E) is deterministic, whereas the task durations are uncertain, see Section 2.3. We model the duration of task $i \in V$ by a continuous function $d_i : X \times \Xi \mapsto \mathbb{R}_+$ that maps resource allocations $x \in X$ and realisations of the uncertain parameters $\xi \in \Xi$ to non-negative durations. We assume that both X, the set of admissible resource allocations, and Ξ, the support of the uncertain parameters, are nonempty and compact subsets of finite-dimensional spaces. Having in mind the application areas outlined in Section 1.1, we assume that ξ cannot be observed directly, but that it can only be gradually inferred from the durations of completed tasks, see Section 2.3. In strategic decision problems, Ξ is sometimes specified as a discrete set of rival scenarios (*e.g.*, different forecasts of market developments). We will see that under rather general convexity assumptions, robust allocation problems that minimise the worst-case makespan over finite discrete supports Ξ can be formulated as explicit convex programs. Often, however, Ξ is better described by a set of infinite cardinality, such as an ellipsoid around a nominal parameter vector. In this chapter, we focus on uncertainty sets that are of infinite cardinality but specific structure.

We define the *r̲obust resource allocation problem on t̲emporal n̲etworks* as

$$\min_{x \in X} \max_{\xi \in \Xi} \min_{y \in Y(x,\xi)} \left\{ y_n + d_n(x;\xi) \right\}, \qquad (\mathcal{RTN})$$

where

$$Y(x,\xi) = \left\{ y \in \mathbb{R}^n_+ \, : \, y_j \geq y_i + d_i(x;\xi) \;\; \forall \, (i,j) \in E \right\}. \qquad (5.1)$$

For $x \in X$ and $\xi \in \Xi$, $Y(x,\xi)$ denotes the set of admissible start time vectors for the network tasks. $\mathcal{RTN}$ is a two-stage robust optimisation problem: the uncertain parameters $\xi \in \Xi$ are revealed after the allocation x has been chosen, but before the task start times y have been decided upon. Hence, we are interested in a static resource allocation which cannot be adapted once information about ξ becomes available. We have already mentioned the reasons for our interest in static allocations in Chapter 4: resource allocations are frequently required to be static due to the inflexibility of resources and limitations of the manufacturing process, or to enhance the planning security and the compatibility with concurrent operations outside the scope of the model. Even in situations where recourse decisions are principally possible, static allocations might be preferable to ensure computational tractability [GG06, JWW98]. To illustrate the importance of static resource allocations, consider the gate sizing problem outlined in Section 1.1. The gate sizes have to be chosen before the impact of process deviations is known. Hence, only static allocations are meaningful in digital circuit design. In the applications described in Section 1.1, unlike the resource allocation x, the task start times y may depend on the available knowledge about ξ. Note that every component of y is chosen after *all* uncertain parameters are revealed, which seems to violate non-anticipativity [RS03]: the uncertain parameters are revealed gradually when tasks are completed, and y_j, $j \in V$, must only depend on information that is available at the time when task j is started. The justification for the chosen two-stage structure is the same as in the previous chapter. The early start schedule $y^* : X \times \Xi \mapsto \mathbb{R}^n_+$ with $y^*_1(x, \xi) = 0$ and

$$y^*_j(x, \xi) = \max_{i \in V} \{y^*_i(x, \xi) + d_i(x; \xi) \; : \; (i, j) \in E\} \quad \text{for all } j \in V \setminus \{1\}$$

is non-anticipative since the task start times only depend on the completion times of predecessor tasks. Moreover, since the makespan is a non-decreasing function of the task start times, the early start schedule is also optimal. Hence, if a solution to $\mathcal{RTN}$ employs an anticipative start time schedule y, then we can replace it with the corresponding (non-anticipative) early start schedule without sacrificing optimality.

The robust resource allocation problem treated in this chapter has relevance in all application areas outlined in Section 1.1. The solution approach proposed in this chapter is also suited for several variants of $\mathcal{RTN}$, such as multi-objective problems that contain the makespan as one of several goals and problems with makespan restrictions as side constraints. We will see an example of such an extension in our case study in Section 5.7.

5.2.2 Decision Rule Approximations

$\mathcal{RTN}$ constitutes a min-max-min problem with coupled constraints and is as such not amenable to standard optimisation techniques. Most existing solution approaches rely on the following observation to obtain a tractable approximation to $\mathcal{RTN}$.

Observation 5.2.1 *For the robust resource allocation problem $\mathcal{RTN}$, we have*

$$\min_{x \in X} \max_{\xi \in \Xi} \min_{y \in Y(x,\xi)} \{y_n + d_n(x;\xi)\} = \min_{\substack{x \in X, \\ y \in \mathcal{Y}(x)}} \max_{\xi \in \Xi} \{y_n(\xi) + d_n(x;\xi)\}, \tag{5.2a}$$

where for $x \in X$,

$$\mathcal{Y}(x) = \left\{(y : \Xi \mapsto \mathbb{R}^n_+) \, : \, y(\xi) \in Y(x,\xi) \ \ \forall \xi \in \Xi\right\}. \tag{5.2b}$$

For a resource allocation $x \in X$, $\mathcal{Y}(x)$ denotes the space of all functions on Ξ that map parameter realisations to feasible start time vectors for the tasks.

Note that the identity (5.2a) holds regardless of the properties of X and d because $\mathcal{Y}(x)$ does not impose any structure on the decision rules (such as measurability). Observation 5.2.1 allows us to reduce the min-max-min problem $\mathcal{RTN}$ to a min-max problem at the cost of augmenting the set of first-stage decisions. We have already encountered this transformation in Section 2.2.2 when we discussed generic two-stage robust optimisation problems. A function y is called a *decision rule* because it specifies the second-stage decision as a function of the uncertain parameters. Note that the choice of an appropriate decision rule is part of the first-stage decision. Since $\mathcal{Y}(x)$ constitutes a function space, further assumptions are required to ensure solvability. For example, if Ξ contains finitely many scenarios, $\Xi = \left\{\xi^1, \ldots, \xi^L\right\}$, then $\mathcal{Y}(x)$ is isomorphic to a subset of $\mathbb{R}^{Ln}_+$ and we can reformulate $\mathcal{RTN}$ as

$$\min_{\substack{x \in X, \\ y \in \mathbb{R}^{Ln}_+}} \left\{ \max_{l=1,\ldots,L} \{y^l_n + d_n(x,\xi^l)\} \, : \, y^l_j \geq y^l_i + d_i(x;\xi^l) \ \ \forall \, l = 1, \ldots, L, \ (i,j) \in E \right\}.$$

This problem is convex if X is convex and d is convex in its first component for all $\xi^l \in \Xi$. Similar finite-dimensional problems arise when a semi-infinite programming algorithm is used to solve $\mathcal{RTN}$ with an uncertainty set of infinite cardinality [HK93]. This approach, however, would only provide lower bounds on the optimal value of $\mathcal{RTN}$, and it is not clear how to efficiently obtain upper bounds.[1] Furthermore, one would not be able to exploit structural properties of Ξ and d beyond convexity. Finally, the number of constraints and variables grows

[1]As we will see in Section 5.2.3, evaluating the worst-case makespan of the optimal second-stage policy in $\mathcal{RTN}$ constitutes a difficult problem even for fixed $x \in X$.

with L, which itself is likely to become large for tight approximations.

Due to the absence of standard optimisation techniques for the solution of $\mathcal{RTN}$ when Ξ has infinite cardinality, one commonly settles for feasible but suboptimal solutions. These are obtained from conservative approximations of $\mathcal{RTN}$ that restrict the set of admissible second-stage decisions. For example, it has been suggested in [LJF04] to restrict $\mathcal{Y}$ to *constant decision rules*, that is, to

$$\mathcal{Y}^0(x) = \{y \in \mathcal{Y}(x) \, : \, \exists \, \gamma \in \mathbb{R}^n \;\text{ such that }\; y(\xi) = \gamma \;\; \forall \, \xi \in \Xi\} \quad \text{for } x \in X.$$

In this case, $\mathcal{RTN}$ is equivalent to

$$\begin{aligned}
&\min_{\substack{x \in X, \\ y \in \mathcal{Y}^0(x)}} \; \max_{\xi \in \Xi} \, \{y_n(\xi) + d_n(x;\xi)\} \\
= &\min_{\substack{x \in X, \\ \gamma \in \mathbb{R}^n_+}} \left\{ \max_{\xi \in \Xi} \, \{\gamma_n + d_n(x;\xi)\} \, : \, \gamma_j \geq \gamma_i + d_i(x;\xi) \;\; \forall \, \xi \in \Xi, \, (i,j) \in E \right\} \\
= &\min_{\substack{x \in X, \\ \gamma \in \mathbb{R}^n_+}} \left\{ \gamma_n + \max_{\xi \in \Xi} \, \{d_n(x;\xi)\} \, : \, \gamma_j - \gamma_i \geq \max_{\xi \in \Xi} \, \{d_i(x;\xi)\} \;\; \forall \, (i,j) \in E \right\}.
\end{aligned}$$

The tractability of this problem is determined by the properties of X and the functions $\max_{\xi \in \Xi} \{d_i(x;\xi)\}$ for $i \in V$. For general Ξ and d the problem can be formulated as a semi-infinite program [HK93]. For specific choices of Ξ and d, robust optimisation techniques can be used to obtain equivalent (or approximate) explicit reformulations [BS06, BTGN09]. Although they are computationally attractive, constant decision rules can result in poor approximations of the optimal second-stage policies and – as a consequence – the optimal resource allocations.

Example 5.2.1 *Consider the temporal network $G = (V, E)$ with tasks $V = \{1, \ldots, n\}$ and precedence relations $E = \{(i, i+1) \, : \, 1 \leq i < n\}$. Let $\Xi = \left\{\xi \in \mathbb{R}^n_+ \, : \, \mathrm{e}^\top \xi \leq 1\right\}$ and the (decision-independent) task durations be defined as $d_i(x;\xi) = \xi_i$ for $i \in V$. The optimal second-stage policy incurs a worst-case makespan of 1, whereas the restriction to constant decision rules results in a worst-case makespan of n.*

In order to improve on the approximation quality of constant decision rules, it has been suggested in [BTGN09, CSS07] to approximate $\mathcal{Y}(x)$ by a set of *affine decision rules*: for $x \in X$ and $\Xi \subseteq \mathbb{R}^k$, we define

$$\mathcal{Y}^1(x) = \left\{ y \in \mathcal{Y}(x) \, : \, \exists\, \Gamma \in \mathbb{R}^{n \times k}, \gamma \in \mathbb{R}^n \;\text{ such that }\; y(\xi) = \Gamma\xi + \gamma \;\; \forall\, \xi \in \Xi \right\}.$$

Under this approximation, $\mathcal{RTN}$ reduces to

$$\begin{aligned} &\min_{\substack{x \in X, \\ y \in \mathcal{Y}^1(x)}} \; \max_{\xi \in \Xi} \left\{ y_n(\xi) + d_n(x;\xi) \right\} \\ = \; &\min_{\substack{x \in X, \\ \Gamma \in \mathbb{R}^{n \times k}, \\ \gamma \in \mathbb{R}^n}} \; \left\{ \gamma_n + \max_{\xi \in \Xi} \left\{ \Gamma_n^\top \xi + d_n(x;\xi) \right\} \, : \, (\Gamma, \gamma) \in \mathcal{S}_+ \cap \mathcal{S}_E(x) \right\} \end{aligned}$$

with

$$\begin{aligned} \mathcal{S}_+ &= \left\{ (\Gamma, \gamma) \, : \, \Gamma\xi + \gamma \geq 0 \;\; \forall\, \xi \in \Xi \right\} \\ &= \left\{ (\Gamma, \gamma) \, : \, \gamma_i \geq \max_{\xi \in \Xi} \left\{ -\Gamma_i^\top \xi \right\} \;\; \forall\, i \in V \right\} \\ \text{and} \quad \mathcal{S}_E(x) &= \left\{ (\Gamma, \gamma) \, : \, \Gamma_j^\top \xi + \gamma_j \geq \Gamma_i^\top \xi + \gamma_i + d_i(x;\xi) \;\; \forall\, \xi \in \Xi, \, (i,j) \in E \right\} \\ &= \left\{ (\Gamma, \gamma) \, : \, \gamma_j - \gamma_i \geq \max_{\xi \in \Xi} \left\{ (\Gamma_i - \Gamma_j)^\top \xi + d_i(x;\xi) \right\} \;\; \forall\, (i,j) \in E \right\}. \end{aligned}$$

Here, $\Gamma_i^\top$ denotes the ith row of matrix Γ. As in the case of constant decision rules, this model can be solved via semi-infinite programming, and under certain conditions we can employ robust optimisation techniques to obtain explicit reformulations. Much like constant decision rules, however, affine decision rules can lead to poor approximations of $\mathcal{RTN}$.

Example 5.2.2 *Consider the class of temporal networks illustrated in Figure 5.1. For $k \in \mathbb{N}$, the network structure is given by $V = \{1, \ldots, 3k+1\}$ and*

$$E = \left\{ (3l+1, 3l+p), (3l+p, 3l+4) \, : \, 0 \leq l < k, \, p = 2, 3 \right\}.$$

Let $d_{3l+2} = \xi_{l+1}$ and $d_{3l+3} = 1 - \xi_{l+1}$ for $0 \leq l < k$, while the remaining task durations are zero. For $\Xi = \left\{ \xi \in \mathbb{R}^k_+ \, : \, \lVert \xi - (1/2)\mathrm{e} \rVert_1 \leq 1/2 \right\}$, the optimal second-stage policy leads to a worst-case

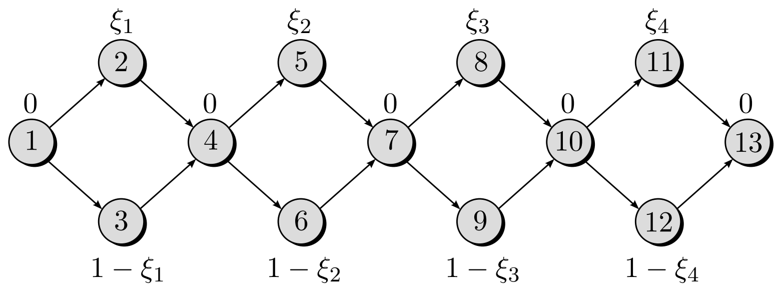


Figure 5.1: Example temporal network that illustrates the suboptimality of affine decision rules. The graph visualises the network structure for $k = 4$. The task durations (next to the nodes) are defined in the text.

makespan of $(k+1)/2$. *For* $0 \leq l < k$, *we obtain* $y_{3l+4}(\xi) \geq y_{3l+1}(\xi) + \max\{\xi_{l+1}, 1-\xi_{l+1}\}$ *for all* $\xi \in \Xi$. *In particular, this inequality holds for* $\xi \in \{(1/2)\mathrm{e} \pm (1/2)\mathrm{e}_{l+1}\}$, *where* e_{l+1} *denotes the* $(l+1)$*th vector of the standard basis in* $\mathbb{R}^k$. *If we restrict* y *to be affine in* ξ, *the previous observation implies that* $y_{3l+4}(\xi) \geq y_{3l+1}(\xi) + 1$ *for* $\xi = (1/2)\mathrm{e} \in \Xi$ *and*

$$y_{3k+1}(\xi) \geq y_{3k-2}(\xi) + 1 \geq \ldots \geq y_1(\xi) + k \geq k \quad \textit{for } \xi = (1/2)\mathrm{e}.$$

Here, the last inequality holds by non-negativity of y. *Thus, the restriction to affine decision rules results in a worst-case makespan of at least* k.

Recently, the use of piecewise affine decision rules has been advocated to overcome some of the deficiencies of affine decision rules [CSSZ08].

Examples 5.2.1 and 5.2.2 show that the existing solution approaches for $\mathcal{RTN}$ can lead to poor approximations of the optimal decisions. This is supported by our numerical results in Section 5.6. In the next section, we show that $\mathcal{RTN}$ constitutes a difficult optimisation problem, which explains the lack of exact solution procedures in the literature.

5.2.3 Complexity Analysis

It is clear that $\mathcal{RTN}$ is difficult to solve if we impose no further regularity conditions beyond compactness of X and Ξ. In the following, we show that evaluating the worst-case makespan of the optimal second-stage policy constitutes an $\mathcal{NP}$-complete problem even when the resource allocation $x \in X$ is fixed, while Ξ and d have 'simple' descriptions. This implies that $\mathcal{RTN}$ is $\mathcal{NP}$-hard since we can restrict X to a singleton and thus obtain a procedure that evaluates the worst-case makespan of the optimal second-stage policy.

In view of the aforementioned objective, we define the *<u>w</u>orst-<u>c</u>ase <u>m</u>akespan of a <u>t</u>emporal <u>n</u>etwork* (WCMTN) problem as follows.
INSTANCE. *A temporal network $G = (V, E)$ with $V = \{1, \ldots, n\}$ and 1 and n as unique source and sink, respectively. Vectors $w, u \in \mathbb{N}_0^n$ and scalars $W, U \in \mathbb{N}_0$.*
QUESTION. *Is there a $\xi \in \Xi = \left\{\xi \in \mathbb{R}_+^n \,:\, \xi \leq \mathrm{e},\, w^\top \xi \leq W\right\}$ such that*

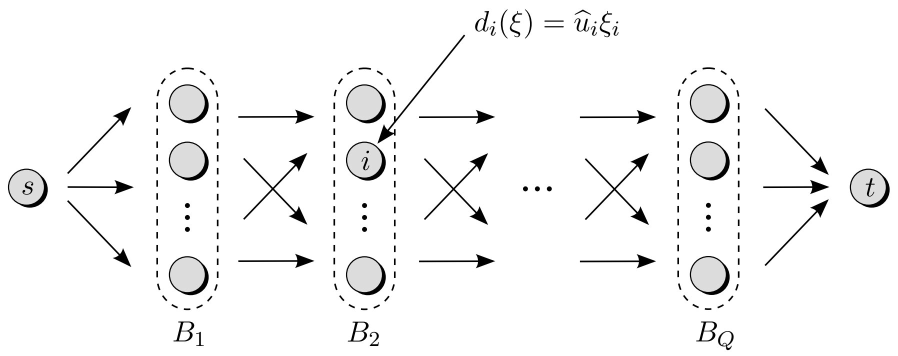
$$\min_{y \in \mathbb{R}_+^n} \{y_n + u_n \xi_n \,:\, y_j \geq y_i + u_i \xi_i \;\; \forall\, (i,j) \in E\} \;\; \geq \;\; U? \tag{5.3}$$

WCMTN considers instances of $\mathcal{RTN}$ with a fixed resource allocation $x \in X$, task durations that are linear in ξ and a support that results from intersecting the unit hypercube with a halfspace. WCMTN asks whether the worst-case makespan exceeds U when an optimal start time schedule is implemented.

Theorem 5.2.1 *WCMTN is $\mathcal{NP}$-complete.*

Proof We first show that WCMTN belongs to $\mathcal{NP}$. Afterwards, we prove $\mathcal{NP}$-hardness of WCMTN by constructing a polynomial transformation of the Continuous Multiple Choice Knapsack problem to WCMTN. In this proof, we abbreviate 'polynomial in the input length of WCMTN' by 'polynomial'.

To establish WCMTN's membership in $\mathcal{NP}$, we show that we can guess a ξ, check whether $\xi \in \Xi$, construct an admissible y^* that minimises the left-hand side of (5.3) and verify whether $y_n^* + u_n \xi_n \geq U$ in polynomial time. Assume that we can restrict attention to values of ξ whose bit lengths are polynomial. Then we can check in polynomial time whether $\xi \in \Xi$. Moreover, optimality of the early start schedule (see Section 5.2.1) ensures that y^* with $y_1^* = 0$ and $y_j^* = \max_{i \in V} \{y_i^* + u_i \xi_i \,:\, (i,j) \in E\}$ for $j \in V \setminus \{1\}$ minimises the left-hand side of (5.3). In particular, this y^* also possesses a polynomial bit length and can be determined in polynomial



$d_i(\xi) = \widehat{u}_i \xi_i$
s
i
t
B_1
B_2
B_Q
$\Xi = \left\{\xi \in \mathbb{R}^{m+2}_+ \,:\, \xi \leq \mathrm{e},\, \sum_{i=1}^m \widehat{w}_i \xi_i \leq \widehat{W}\right\}$

Figure 5.2: WCMTN instance constructed from a CMCK instance.

time. This implies that the validity of (5.3) can be verified in polynomial time, which in turn implies membership of WCMTN in $\mathcal{NP}$. It remains to be shown that we can indeed restrict attention to values of ξ with polynomial bit lengths. Note that (5.3) is satisfied for some $\xi \in \Xi$ if and only if

$$\max_{\xi \in \Xi} \min_{y \in \mathbb{R}^n_+} \left\{ y_n + u_n \xi_n \,:\, y_j \geq y_i + u_i \xi_i \;\; \forall (i,j) \in E \right\} \quad \geq \quad U.$$

Since the inner minimisation represents a convex function of ξ, its maximum over Ξ is attained by at least one extreme point of Ξ [HPT00]. Since Ξ is a polyhedron, however, all of its extreme points possess polynomial bit lengths [LP94].

In order to prove $\mathcal{NP}$-hardness of WCMTN, we consider the Continuous Multiple Choice Knapsack (CMCK) problem [GJ79, Iba80]:

INSTANCE. *A set $\mathcal{B} = \{1, \ldots, m\}$, together with weights $\widehat{w}_i \in \mathbb{N}_0$ and utilities $\widehat{u}_i \in \mathbb{N}_0$ for $i \in \mathcal{B}$. A partition $\{B_q\}_{q=1}^Q$ of $\mathcal{B}$, that is, $\bigcup_q B_{q=1}^Q = \mathcal{B}$ and $B_q \cap B_r = \emptyset$ for $q \neq r$. A maximum weight $\widehat{W} \in \mathbb{N}_0$ and a minimum utility $\widehat{U} \in \mathbb{N}_0$.*

QUESTION. *Is there a choice of $b_q \in B_q$ and $\widehat{\xi}_q \in [0,1]$, $q = 1, \ldots, Q$, such that $\sum_{q=1}^Q \widehat{w}_{b_q} \widehat{\xi}_q \leq \widehat{W}$ and $\sum_{q=1}^Q \widehat{u}_{b_q} \widehat{\xi}_q \geq \widehat{U}$?*

We construct a polynomial-time transformation that converts a CMCK instance to a WCMTN instance such that the answer to the former problem is affirmative if and only if the answer to the latter one is.

The desired WCMTN instance is defined by $G = (V, E)$, $V = \{s, 1, \ldots, m, t\}$ and $E = E_B \cup E_G$ with $E_B = \{(i,j) \,:\, (i,j) \in B_q \times B_{q+1}, q = 1, \ldots, Q-1\}$ and $E_G = \{(s,i) \,:\, i \in B_1\} \cup$

$\{(i,t) : i \in B_Q\}$. The nodes s and t represent the unique source and sink of G, respectively. We set $w_i = \widehat{w}_i$ and $u_i = \widehat{u}_i$ for $i = 1, \ldots, m$, while $w_i = u_i = 0$ for $i \in \{s,t\}$. We identify W and U with $\widehat{W}$ and $\widehat{U}$, respectively. The transformation is illustrated in Figure 5.2.

For the constructed WCMTN instance, assume that there is a $\xi \in \Xi$ which satisfies (5.3). Let y^* be a minimiser for the left-hand side of (5.3). By construction of G and optimality of y^*, there is a critical path $(s, b_1, \ldots, b_Q, t)$ in G with $b_q \in B_q$ for $q = 1, \ldots, Q$, $y^*_s = y^*_{b_1} = 0$, $y^*_{b_{q+1}} = y^*_{b_q} + u_{b_q}\xi_{b_q}$ for $q = 1, \ldots, Q-1$ and $y^*_t = y^*_{b_Q} + u_{b_Q}\xi_{b_Q}$ [DH02]. Since $y^*_t \geq U$, we conclude that $\sum_{q=1}^{Q} u_{b_q}\xi_{b_q} = \sum_{q=1}^{Q} \widehat{u}_{b_q}\xi_{b_q} \geq U = \widehat{U}$. Similarly, we have $\sum_{q=1}^{Q} w_{b_q}\xi_{b_q} = \sum_{q=1}^{Q} \widehat{w}_{b_q}\xi_{b_q} \leq W = \widehat{W}$ because $\xi \in \Xi$. Thus, b and $\widehat{\xi}$ with $\widehat{\xi}_q = \xi_{b_q}$, $q = 1, \ldots, Q$, certify that the answer to the CMCK instance is affirmative as well. In the same way, one can show that the absence of a $\xi \in \Xi$ which satisfies (5.3) implies that the answer to the CMCK instance is negative. ■

Theorem 5.2.1 extends to problem instances whose uncertainty sets are polyhedral [BS06] or that result from intersections of general ellipsoids as in [BTGN09]. However, it is easy to see that WCMTN can be decided in polynomial time for box uncertainty sets of the form $\Xi = \left\{\xi : \underline{\xi} \leq \xi \leq \overline{\xi}\right\}$ with $\underline{\xi}, \overline{\xi} \in \mathbb{R}^k$. The same holds true for the special case of WCMTN in which $w = \alpha \mathrm{e}$ and $u = \beta \mathrm{e}$ for $\alpha, \beta \in \mathbb{N}_0$.

We close with a review of two related complexity results. The complexity of optimisation problems in temporal networks with probabilistic uncertainty is investigated in [Hag88]. In this paper the task durations are modelled as independent random variables with known, discrete distributions, and it is shown that calculating the mean or certain quantiles of the makespan distribution is #PSPACE-hard. We remark, however, that the worst-case duration (*i.e.*, the 100%-quantile of the makespan distribution) can be calculated in polynomial time in that setting. In contrast, the additional complexity of WCMTN is due to the fact that our task durations are related through Ξ. The $\mathcal{NP}$-hardness of a generic robust resource allocation problem is proven in [KY97]. However, this problem is not defined on a network, and it assumes that X and Ξ are discrete.

5.3 Path-Wise Problem Formulation

In contrast to the techniques reviewed in Section 5.2.2, our solution approach for $\mathcal{RTN}$ does not approximate the optimal second-stage decision by decision rules. Instead, we eliminate the inner minimisation in $\mathcal{RTN}$ by enumerating the task paths of the network. We have seen a solution scheme based on path enumeration for two-stage chance constrained problems in the previous chapter. In this section, we present a path-wise reformulation of $\mathcal{RTN}$ and argue that its direct solution is prohibitive for temporal networks with large numbers of task paths. In the next two sections, we will use this path-wise reformulation to derive convergent bounds on the optimal value of $\mathcal{RTN}$.

We recall that a path in a directed graph $G = (V, E)$ constitutes a list of nodes $(i_1, \ldots, i_p)$ such that $(i_1, i_2), \ldots, (i_{p-1}, i_p) \in E$. Accordingly, we define a *task path* $P = \{i_1, \ldots, i_p\} \subseteq V$ as a set of tasks whose nodes form a path in the temporal network. We denote by $\mathcal{P}$ the set of all task paths. The following observation re-iterates the well-known fact (see for example [DH02]) that for fixed x and ξ, the minimal makespan of a temporal network equals the sum of all task durations along any of its critical (*i.e.*, most time-consuming) task paths.

Observation 5.3.1 *For a temporal network $G = (V, E)$ with fixed resource allocation $x \in X$ and parameters $\xi \in \Xi$, the minimal makespan is given by*

$$\min_{y \in Y(x,\xi)} \{y_n + d_n(x;\xi)\} \;=\; \max_{P \in \mathcal{P}} \left\{\mathbb{I}_P^\top d(x;\xi)\right\}, \tag{5.4}$$

where $d(x;\xi) = (d_1(x;\xi), \ldots, d_n(x;\xi))^\top$ and $Y(x,\xi)$ is defined in (5.1).

Note that the maximum on the right-hand side of (5.4) can be attained by several task paths $P \in \mathcal{P}$. Observation 5.3.1 is crucial as it allows us to replace the inner minimisation in $\mathcal{RTN}$ with a maximisation. In analogy to Observation 5.2.1, this reduces the two-stage robust optimisation problem to an equivalent single-stage problem. Readers familiar with robust optimisation may wonder whether a similar reduction can be achieved through duality arguments, see Section 2.2.2. Due to the structure of $Y(x,\xi)$, this approach results in a maximisation problem whose objective function is nonconvex, and the resulting single-stage robust optimisation problem would be difficult to solve. Observation 5.3.1 bypasses this problem at the expense of optimising over a potentially large number of task paths.

Example 5.3.1 *Consider the temporal network defined by the subgraph that contains the first four nodes in Figure 5.1. Its minimal makespan is given by*

$$\min_{y \in \mathbb{R}^4_+} \big\{ y_4 + d_4(x;\xi) \; : y_j \geq y_1 + d_1(x;\xi) \;\; \textit{for } j = 1, 2,$$
$$y_4 \geq y_j + d_j(x;\xi) \;\; \textit{for } j = 1, 2 \big\}.$$

By linear programming duality, this problem is equivalent to

$$\max_{\lambda \in \mathbb{R}^2_+} \big\{ \, [d_1(x;\xi) + d_2(x;\xi)] \, \lambda_1 + [d_1(x;\xi) + d_3(x;\xi)] \, \lambda_2 + d_4(x;\xi) \; : \; \lambda_1 + \lambda_2 \leq 1 \big\}.$$

For most task duration functions of interest, the objective function of this problem is nonconvex in ξ and λ. In contrast, enumerating the tasks paths yields

$$\max \left\{ d_1(x;\xi) + d_2(x;\xi) + d_4(x,\xi), \; d_1(x;\xi) + d_3(x;\xi) + d_4(x,\xi) \right\}.$$

The expressions in this maximisation are convex in ξ if $d(x;\xi)$ is convex in ξ.

Applying Observation 5.3.1 to $\mathcal{RTN}$, we find

$$\min_{x \in X} \max_{\xi \in \Xi} \min_{y \in Y(x,\xi)} \{ y_n + d_n(x;\xi) \} \; = \; \min_{x \in X} \max_{P \in \mathcal{P}} \max_{\xi \in \Xi} \big\{ \mathbb{I}_P^\top \, d(x;\xi) \big\}.$$

In the following, we will employ robust optimisation techniques to replace the maximisation over Ξ. We are thus concerned with the following *approximate robust resource allocation problem on temporal networks*:

$$\min_{x \in X} \max_{P \in \mathcal{P}} \; \phi(x;P), \qquad (\mathcal{ARTN})$$

where $\phi(\cdot;P)$ represents a real-valued function on X. We call $\mathcal{ARTN}$ a *conservative reformu-*

lation of $\mathcal{RTN}$ if

$$\phi(x; P) \geq \max_{\xi \in \Xi} \left\{ \mathbb{I}_P^\top d(x; \xi) \right\} \quad \text{for } x \in X,\ P \subseteq V. \tag{5.5}$$

If (5.5) holds, optimal allocations for $\mathcal{ARTN}$ constitute suboptimal but feasible allocations for $\mathcal{RTN}$, and the optimal value of $\mathcal{ARTN}$ overestimates the worst-case makespan in $\mathcal{RTN}$. If the inequality in (5.5) can be replaced with an equality, we call $\mathcal{ARTN}$ an *exact reformulation* of $\mathcal{RTN}$. In this case, $\mathcal{ARTN}$ and $\mathcal{RTN}$ are equivalent. Our bounding approach is applicable to exact and conservative reformulations of $\mathcal{RTN}$ alike. Note, however, that our method provides upper and lower bounds on $\mathcal{ARTN}$, and that these bounds will only bracket the optimal value of $\mathcal{RTN}$ if $\mathcal{ARTN}$ constitutes an exact reformulation.

Apart from $\mathcal{ARTN}$ being an exact or conservative reformulation of $\mathcal{RTN}$, our bounding approach requires ϕ to satisfy the following two properties:

(A1) **Monotonicity.** If $P \subset P' \subseteq V$, then $\phi(x; P) \leq \phi(x; P')$ for all $x \in X$.

(A2) **Sub-Additivity.** If $P \subset P' \subseteq V$, then $\phi(x; P) + \phi(x; P' \setminus P) \geq \phi(x; P')$ for all $x \in X$.

We call $P \in \mathcal{P}$ an *inclusion-maximal path* if there is no $P' \in \mathcal{P}$, $P' \neq P$, such that $\mathbb{I}_P \leq \mathbb{I}_{P'}$. As in the previous chapter, we denote the set of inclusion-maximal paths by $\overline{\mathcal{P}} \subseteq \mathcal{P}$. If (A1) is satisfied, then the optimal allocations and the optimal value of $\mathcal{ARTN}$ do not change if we replace $\mathcal{P}$ with $\overline{\mathcal{P}}$. (A2) implies that $\phi(x; P)$ is bounded from above by $\sum_{r=1}^{R} \phi(x; P_r)$ for all $x \in X$ if $\{P_r\}_{r=1}^{R}$ forms a partition of P. As we will see, this bounding property facilitates the construction of lower and upper bounds on the optimal value of $\mathcal{ARTN}$. The following proposition shows that exact reformulations of $\mathcal{RTN}$ necessarily satisfy (A1) and (A2).

Proposition 5.3.1 *If $\mathcal{ARTN}$ is an exact reformulation of $\mathcal{RTN}$, then (A1) and (A2) are satisfied.*

Proof For $P \subset P' \subseteq V$ and $x \in X$, we obtain

$$\phi(x; P') \;=\; \max_{\xi \in \Xi} \left\{ \mathbb{I}_{P'}^\top d(x; \xi) \right\} \;\geq\; \max_{\xi \in \Xi} \left\{ \mathbb{I}_P^\top d(x; \xi) \right\} \;=\; \phi(x; P),$$

where the inequality follows from $\mathbb{I}_{P'} \geq \mathbb{I}_P$ and non-negativity of d. Similarly, for $P \subset P' \subseteq V$ and $x \in X$, we obtain

$$\begin{aligned}
\phi(x; P') &= \max_{\xi \in \Xi} \left\{ \mathbb{I}_{P'}^\top d(x; \xi) \right\} \\
&= \max_{\xi \in \Xi} \left\{ \mathbb{I}_P^\top d(x; \xi) + \mathbb{I}_{[P' \setminus P]}^\top d(x; \xi) \right\} \\
&\leq \max_{\xi \in \Xi} \left\{ \mathbb{I}_P^\top d(x; \xi) \right\} + \max_{\xi \in \Xi} \left\{ \mathbb{I}_{[P' \setminus P]}^\top d(x; \xi) \right\} \\
&= \phi(x; P) + \phi(x; P' \setminus P).
\end{aligned}$$

■

In the following, we focus on instances of $\mathcal{ARTN}$ that can be reformulated as explicit convex optimisation problems. More precisely, we assume that

(A3) **Tractability.** X and $\phi(\cdot; P)$, $P \subseteq V$, possess tractable representations.

Remember that a set has a tractable representation if set membership can be described by finitely many convex constraints and auxiliary variables. Likewise, a function has a tractable representation if its epigraph does. Although our solution approach does not rely on (A3), the repeated solution of lower and upper bound problems becomes computationally prohibitive if (A3) fails to hold. In the following, we show that robust optimisation techniques allow us to construct exact or conservative reformulations of $\mathcal{RTN}$ that satisfy (A1)–(A3) for natural choices of X, Ξ and d.

Proposition 5.3.2 *If X has a tractable representation, then the following choices of Ξ and d allow for exact reformulations of $\mathcal{RTN}$ that satisfy (A1)–(A3):*

1. ***Affine Uncertainty.*** *$d_i(x; \xi) = \delta_i^0(x) + \xi^\top [\delta_i^1(x)]$ with $\delta_i^0 : X \mapsto \mathbb{R}$ tractable, $\delta_i^1 : X \mapsto \mathbb{R}^k$ affine and $\xi \in \Xi = \bigcap_{l=1}^L \Xi_l \subseteq \mathbb{R}^k$ with*

$$\Xi_l = \left\{ \xi \in \mathbb{R}^k \, : \, \exists \, u \in \mathbb{R}^{J_l} \;\; \textit{such that} \;\; \xi = \sigma^l + \Sigma^l u, \; \left\| \Pi^l u \right\|_2 \leq 1 \right\},$$

where $\sigma^l \in \mathbb{R}^k$, $\Sigma^l \in \mathbb{R}^{k\times J_l}$ and Π^l denotes a projection of $\mathbb{R}^{J_l}$ onto a subspace, $l = 1,\ldots,L$. We require Ξ to be bounded and to have a nonempty relative interior.

2. ***Quadratic Uncertainty.*** *$d_i(x;\xi) = \delta_i^0(x) + \xi^\top[\delta_i^1(x)] + \|[\Delta_i^2(x)]\,\xi\|_2^2$ with $\delta_i^0 : X \mapsto \mathbb{R}$ tractable, $\delta_i^1 : X \mapsto \mathbb{R}^k$ and $\Delta_i^2 : X \mapsto \mathbb{R}^{l\times k}$ affine and $\xi \in \Xi \subseteq \mathbb{R}^k$ with*

$$\Xi = \Big\{\xi \in \mathbb{R}^k \;:\; \exists\, u \in \mathbb{R}^J \;\; \textit{such that} \;\; \xi = \sigma + \Sigma u,\; \|u\|_2 \leq 1\Big\},$$

where $\sigma \in \mathbb{R}^k$ and $\Sigma \in \mathbb{R}^{k\times J}$.

Proof Let $\delta^0(x) = \big[\delta_1^0(x),\ldots,\delta_n^0(x)\big]^\top$. In the case of affine uncertainty, we define ϕ through

$$\phi(x;P) = \mathbb{I}_P^\top\big[\delta^0(x)\big] + \max_{\xi\in\Xi}\Big\{\xi^\top\Big(\sum_{i\in P}\big[\delta_i^1(x)\big]\Big)\Big\} \quad \text{for } x \in X,\; P \in \mathcal{P},$$

and in the case of quadratic uncertainty, we define ϕ through

$$\begin{aligned}\phi(x;P) = \mathbb{I}_P^\top\big[\delta^0(x)\big] + \max_{\xi\in\Xi}\Big\{&\xi^\top\Big(\sum_{i\in P}\big[\delta_i^1(x)\big]\Big)+\\ &\big\|\mathrm{vec}\big([\mathbb{I}_P]_1\,\big[\Delta_1^2(x)\big]\xi,\ldots,[\mathbb{I}_P]_n\,\big[\Delta_n^2(x)\big]\xi\big)\big\|_2^2\Big\} \text{ for } x \in X,\; P \in \mathcal{P}.\end{aligned}$$

Here, the operator 'vec' returns the concatenation of its arguments as a column vector. We have $[\mathbb{I}_P]_i = 1$ if P contains task i and $[\mathbb{I}_P]_i = 0$ otherwise. Note that both definitions of $\phi(x;P)$ constitute exact reformulations of $\max_{\xi\in\Xi}\big\{\mathbb{I}_P^\top\, d(x;\xi)\big\}$. In either case, the epigraph of ϕ can be described by a semi-infinite constraint which has to hold for all $\xi \in \Xi$. Robust optimisation techniques [BTGN09] enable us to reformulate these semi-infinite constraints such that (A3) is satisfied. Due to Proposition 5.3.1, (A1) and (A2) are satisfied as well. ∎

The uncertainty set considered in the first part of Proposition 5.3.2 covers all bounded polyhedra as special cases. Sometimes, the durations of network tasks are approximated by conic-quadratic functions, see Chapter 4. It is therefore desirable to extend the results of Proposition 5.3.2 also to problems with conic-quadratic uncertainty.

Proposition 5.3.3 (Conic-Quadratic Uncertainty) *Assume that the task durations are described by $d_i(x;\xi) = \delta_i^0(x) + \xi^\top[\delta_i^1(x)] + \|[\Delta_i^2(x)]\,\xi\|_2$ with $\delta_i^0 : X \mapsto \mathbb{R}$ tractable, $\delta_i^1 : X \mapsto \mathbb{R}^k$ and $\Delta_i^2 : X \mapsto \mathbb{R}^{l\times k}$ affine and $\xi \in \Xi \subseteq \mathbb{R}^k$ with*

$$\Xi = \Big\{\xi \in \mathbb{R}^k \,:\, \exists\, u \in \mathbb{R}^J \;\; \textit{such that} \;\; \xi = \sigma + \Sigma u, \; \|u\|_2 \le 1\Big\},$$

where $\sigma \in \mathbb{R}^k$, $\Sigma \in \mathbb{R}^{k\times J}$. If X has a tractable representation, then Ξ and d allow for a conservative reformulation of $\mathcal{RTN}$ that satisfies (A1)–(A3).

Remark 5.3.1 *In contrast to the case of quadratic uncertainty, the last term of the task duration is not squared under conic-quadratic uncertainty.*

Proof of Proposition 5.3.3 We construct an upper bound on

$$\max_{\xi\in\Xi} \Big\{ \sum_{i\in P} \Big(\delta_i^0(x) + \xi^\top\big[\delta_i^1(x)\big] + \big\|\big[\Delta_i^2(x)\big]\xi\big\|_2\Big)\Big\} \quad \text{for } x \in X,\ P \in \mathcal{P}. \tag{5.6}$$

The terms in the objective of this problem either do not depend on ξ, or they are convex and linear homogeneous in ξ. Thus, we can apply the results from [BS06] and bound (5.6) from above by

$$\phi(x;P) = \max_{\widehat{u}\in\widehat{\mathcal{U}}} \big\{\mathbb{I}_P^\top\big[\widehat{d}(x;\widehat{u})\big]\big\}, \tag{5.7}$$

where $\widehat{u} = (\widehat{u}^+, \widehat{u}^-)$, and $\widehat{\mathcal{U}}$ is defined through

$$\widehat{\mathcal{U}} = \big\{\widehat{u} = (\widehat{u}^+, \widehat{u}^-) \in \mathbb{R}_+^J \times \mathbb{R}_+^J \,:\, \big\|\widehat{u}^+ + \widehat{u}^-\big\|_2 \le 1\big\}\,.$$

Moreover, $\widehat{d} : X \times \mathbb{R}_+^{2J} \mapsto \mathbb{R}^n$ has components $\widehat{d}(x;\widehat{u}) = \big[\widehat{d}_1(x;\widehat{u}), \ldots, \widehat{d}_n(x;\widehat{u})\big]^\top$ that are defined through

$$\widehat{d}_i(x;\widehat{u}) = \delta_i^0(x) + \Big[\sigma + \Sigma(\widehat{u}^+ - \widehat{u}^-)\Big]^\top \big[\delta_i^1(x)\big] + \underbrace{\big\|\big[\Delta_i^2(x)\big]\sigma\big\|_2 + \sum_{j=1}^{J} \big\|\big[\Delta_i^2(x)\big]\Sigma_j\big\|_2\,(\widehat{u}_j^+ + \widehat{u}_j^-)}_{\alpha_i(x;\widehat{u})},$$

where Σ_j denotes the jth column of Σ. The epigraph of $\phi(x; P)$ can be described by a semi-infinite constraint that has to hold for all $\widehat{u} \in \widehat{\mathcal{U}}$. Due to the specific shape of $\widehat{\mathcal{U}}$ and the fact that $\widehat{d}$ is affine in $\widehat{u}$, robust optimisation techniques can be employed to reformulate this semi-infinite constraint such that (A3) is satisfied. It remains to be shown that ϕ also satisfies (A1) and (A2).

As for (A1), we show that $\widehat{d}_i(x; \widehat{u})$, $i \in V$, is non-negative for all $x \in X$ and $\widehat{u} \in \widehat{\mathcal{U}}$. To this end, we fix some $\widehat{u} = (\widehat{u}^+, \widehat{u}^-) \in \widehat{\mathcal{U}}$ and set $u = \widehat{u}^+ - \widehat{u}^-$. Then $\xi = \sigma + \Sigma u$ is contained in Ξ since $\|u\|_2 \leq 1$. Hence, for $x \in X$,

$$d_i(x; \xi) = \delta_i^0(x) + \Big[\sigma + \Sigma u\Big]^\top \left[\delta_i^1(x)\right] + \underbrace{\left\|\left[\Delta_i^2(x)\right]\left[\sigma + \Sigma u\right]\right\|_2}_{\beta_i(x;u)} \geq 0$$

by non-negativity of d. Note that $\widehat{d}_i(x; \widehat{u}) - d_i(x; \xi) = \alpha_i(x; \widehat{u}) - \beta_i(x; u)$ for this choice of ξ. Since $d_i(x; \xi) \geq 0$, non-negativity of $\widehat{d}_i(x; \widehat{u})$ is ensured if $\alpha_i(x; \widehat{u}) \geq \beta_i(x; u)$. The latter inequality follows the triangle inequality, the positive homogeneity of norms and the fact that $|u_j| \leq \widehat{u}_j^+ + \widehat{u}_j^-$.

As for (A2), we need to show that $\phi(x; P) + \phi(x; P' \setminus P) \geq \phi(x; P')$ for $x \in X$ and $P \subset P' \subseteq V$. This is the case since

$$\max_{\widehat{u} \in \widehat{\mathcal{U}}} \left\{\mathbb{I}_P^\top \left[\widehat{d}(x; \widehat{u})\right]\right\} + \max_{\widehat{u} \in \widehat{\mathcal{U}}} \left\{\mathbb{I}_{[P' \setminus P]}^\top \left[\widehat{d}(x; \widehat{u})\right]\right\} \geq \max_{\widehat{u} \in \widehat{\mathcal{U}}} \left\{\mathbb{I}_{P'}^\top \left[\widehat{d}(x; \widehat{u})\right]\right\}.$$

$\blacksquare$

Proposition 5.3.3 provides a conservative reformulation of $\mathcal{RTN}$. Exact reformulations of robust optimisation problems subject to conic-quadratic uncertainty are discussed in [BTGN09]. However, the path durations $\phi(x; P)$ resulting from conic-quadratic uncertainty are not of the form required in [BTGN09], and the corresponding reformulation does not seem to be applicable to our context.

Note that even if (A3) is satisfied, $\mathcal{ARTN}$ remains generically intractable since its size grows with the the cardinality of $\mathcal{P}$, which in turn can be exponential in the size of G. Indeed, the expected number of paths in a uniformly sampled random temporal network is exponential. We defer the proof of this statement to Appendix A. Hence, even though $\mathcal{ARTN}$ can be expressed as an explicit convex optimisation problem, it remains difficult to solve.

We close with an example that illustrates our path-wise problem formulation $\mathcal{ARTN}$.

Example 5.3.2 *Consider the temporal network in Figure 5.3. Apart from the missing cash flows, it is identical to the temporal network in Figure 1.1. Now, however, we interpret the number attached to task $i \in V$ as the nominal duration of task i. We consider a resource allocation problem with one resource and task durations*

$$d_i(x;\xi) := d_i^0\,(1 - x_i)\,(1 + \xi_i) \qquad \text{for } i \in V,$$

where d_i^0 denotes the nominal task duration from Figure 5.3, x_i the amount of the resource that is assigned to task i, and ξ_i the uncertainty inherent to the task duration. We set

$$\begin{aligned} X &:= \left\{x \in \mathbb{R}^6_+ \;:\; x_i \leq 1/2,\ \mathrm{e}^\top x \leq 1\right\} \\ \text{and} \quad \Xi &:= \left\{\xi \in \mathbb{R}^6_+ \;:\; \xi_i \leq 1/2,\ \mathrm{e}^\top \xi \leq 1\right\}. \end{aligned}$$

Thus, the duration of task i can fall below or exceed its nominal duration d_i^0 by 50%, depending on the resource allocation and the realisation of the uncertain parameter vector ξ. Up to two tasks can be sped up to their minimal durations, and up to two tasks on each inclusion-maximal path can attain their worst-case durations.

For the network in Figure 5.3, the set $\mathcal{P}$ of all task paths contains 22 elements. Elements of $\mathcal{P}$ are, amongst others, $\{1\}$, $\{2\}$, ..., $\{6\}$, $\{1,2\}$, $\{1,3\}$, ..., $\{5,6\}$ and $\{1,2,5\}$. The set $\overline{\mathcal{P}}$ of inclusion-maximal task paths only contains three elements, namely $\{1,2,4,6\}$, $\{1,2,5,6\}$ and $\{1,3,5,6\}$. Since our problem instance satisfies the conditions of the first part of Proposition 5.3.2, we can develop an exact reformulation of $\mathcal{RTN}$ that satisfies (A1)–(A3). Indeed, *for our choice of functions we have*

$$\begin{aligned}
\phi(x;P) &= \max_{\xi_i \in \mathbb{R}_+ \,:\, i \in P} \left\{ \sum_{i \in P} d_i^0 (1 - x_i)(1 + \xi_i) \;:\; \xi_i \leq 1/2 \;\; \forall\, i \in P,\; \sum_{i \in P} \xi_i \leq 1 \right\} \\
&= \min_{\substack{\lambda_i \in \mathbb{R}_+ \,:\, i \in P, \\ \gamma \in \mathbb{R}_+}} \left\{ \left[\sum_{i \in P} d_i^0 (1 - x_i) + \lambda_i/2 \right] + \gamma \;:\; \lambda_i + \gamma \geq d_i^0 (1 - x_i) \;\; \forall\, i \in P \right\},
\end{aligned}$$

where the first identity holds by definition, and the second one follows from linear programming duality. Note that our reformulation $\mathcal{ARTN}$ satisfies (A3) since

$$\begin{aligned}
\tau \geq \phi(x;P) \quad \Leftrightarrow \quad \exists\, (\lambda_i \in \mathbb{R}_+ \,:\, i \in P),\; \gamma \in \mathbb{R}_+ \;: \tau &\geq \left[\sum_{i \in P} d_i^0 (1 - x_i) + \lambda_i/2 \right] + \gamma, \\
\lambda_i + \gamma &\geq d_i^0 (1 - x_i) \;\; \forall\, i \in P,
\end{aligned}$$

and the right-hand side of this equivalence can be expressed by finitely many linear constraints and auxiliary variables. For the temporal network in Figure 5.3, our reformulation $\mathcal{ARTN}$ results in the following optimisation problem.

$$\begin{aligned}
\underset{\tau, x, \lambda, \gamma}{\text{minimise}} \quad & \tau \\
\text{subject to} \quad & \tau \in \mathbb{R}_+, \quad x \in \mathbb{R}^6_+, \quad \lambda \in \mathbb{R}^{12}_+, \quad \gamma \in \mathbb{R}^3_+ \\
& \tau \geq 2(1 - x_1) + \lambda_1^1/2 + 5(1 - x_2) + \lambda_2^1/2 + 4(1 - x_4) + \lambda_4^1/2 + 1(1 - x_6) + \lambda_6^1/2 + \gamma^1, \\
& \lambda_1^1 + \gamma^1 \geq 2(1 - x_1), \quad \lambda_2^1 + \gamma^1 \geq 5(1 - x_2), \quad \lambda_4^1 + \gamma^1 \geq 4(1 - x_4), \quad \lambda_6^1 + \gamma^1 \geq 1(1 - x_6), \\
& \tau \geq 2(1 - x_1) + \lambda_1^2/2 + 5(1 - x_2) + \lambda_2^2/2 + 3(1 - x_5) + \lambda_5^2/2 + 1(1 - x_6) + \lambda_6^2/2 + \gamma^2, \\
& \lambda_1^2 + \gamma^2 \geq 2(1 - x_1), \quad \lambda_2^2 + \gamma^2 \geq 5(1 - x_2), \quad \lambda_5^2 + \gamma^2 \geq 3(1 - x_5), \quad \lambda_6^2 + \gamma^2 \geq 1(1 - x_6), \\
& \tau \geq 2(1 - x_1) + \lambda_1^3/2 + 1(1 - x_3) + \lambda_3^3/2 + 3(1 - x_5) + \lambda_5^3/2 + 1(1 - x_6) + \lambda_6^3/2 + \gamma^3, \\
& \lambda_1^3 + \gamma^3 \geq 2(1 - x_1), \quad \lambda_3^3 + \gamma^3 \geq 1(1 - x_3), \quad \lambda_5^3 + \gamma^3 \geq 3(1 - x_5), \quad \lambda_6^3 + \gamma^3 \geq 1(1 - x_6), \\
& x_i \leq 1/2 \;\; \forall\, i \in \{1, \ldots, 6\}, \quad \sum_{i=1}^{6} x_i \leq 1.
\end{aligned}$$

The optimal allocation to this problem is $x = (0, 0.50, 0, 0.36, 0.14, 0)^\top$ *and leads to a worst-case makespan of* 10.61.

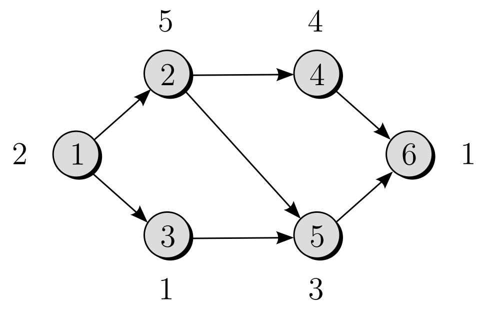


Figure 5.3: Example temporal network. The chart illustrates the network structure and the nominal durations d_i^0 of the network tasks $i \in V$ (attached to the nodes).

5.4 Lower Bounds

We determine convergent lower bounds on $\mathcal{ARTN}$ by solving relaxations that omit some of the paths in $\mathcal{ARTN}$:

Algorithm 5.1 Convergent lower bounds on $\mathcal{ARTN}$.

1. **Initialisation.** Choose a subset $\mathcal{P}_1 \subseteq \overline{\mathcal{P}}$, for example $\mathcal{P}_1 = \emptyset$. Set $t = 1$.

2. **Master Problem.** Solve $\mathcal{ARTN}$, restricted to the paths in $\mathcal{P}_t$:

$$\min_{\substack{x \in X,\\ \tau \in \mathbb{R}_+}} \left\{ \tau \, : \, \tau \geq \phi(x; P) \;\; \forall \, P \in \mathcal{P}_t \right\}. \tag{$\mathcal{LARTN}_t$}$$

   Let x^t denote an optimal solution to $\mathcal{LARTN}_t$ and τ^t its objective value.

3. **Subproblem.** Determine a path $P \in \overline{\mathcal{P}} \setminus \mathcal{P}_t$ with $\phi(x^t; P) > \tau^t$.

   (a) If no such path exists, stop: $x^* = x^t$ constitutes an optimal solution to $\mathcal{ARTN}$ and $\tau^* = \tau^t$ its objective value.

   (b) Otherwise, set $\mathcal{P}_{t+1} = \mathcal{P}_t \cup \{P\}$, $t \to t + 1$ and go to Step 2.

The following proposition is an immediate consequence of the algorithm outline.

Proposition 5.4.1 *Algorithm 5.1 terminates with an optimal allocation x^* for $\mathcal{ARTN}$, together with its worst-case makespan τ^*. Furthermore, $\{\tau^t\}_t$ represents a monotonically non-decreasing sequence of lower bounds on τ^*.*

Proof Since $t \leq t'$ implies that $\mathcal{P}_t \subseteq \mathcal{P}_{t'}$, $\mathcal{LARTN}_t$ constitutes a relaxation of $\mathcal{LARTN}_{t'}$. Hence, $\tau^t \leq \tau^{t'}$, that is, $\{\tau^t\}_t$ is monotonically non-decreasing. Similarly, every τ^t constitutes a lower bound on the optimal value of $\mathcal{ARTN}$, because the latter problem considers all paths in $\overline{\mathcal{P}}$ and $\mathcal{P}_t \subseteq \overline{\mathcal{P}}$ for all t.

In iteration t, Step 3 either terminates or adds a path $P \in \overline{\mathcal{P}} \setminus \mathcal{P}_t$ to $\mathcal{P}_t$. Hence, the algorithm terminates after $T \leq \left|\overline{\mathcal{P}} \setminus \mathcal{P}_1\right| + 1$ iterations. It is clear that x^* is optimal if $\mathcal{P}_T = \overline{\mathcal{P}}$ in the last iteration. Otherwise, $\phi(x^*; P) \leq \tau^*$ for all $P \in \overline{\mathcal{P}} \setminus \mathcal{P}_T$. Thus, (x^*, τ^*) minimises the relaxation $\mathcal{LARTN}_T$ and x^* is feasible in $\mathcal{ARTN}$. Since x^* attains the same objective value τ^* in $\mathcal{ARTN}$, x^* is an optimal allocation and τ^* the optimal value of $\mathcal{ARTN}$. ■

The size of $\mathcal{LARTN}_t$, $t \geq 1$, grows with the cardinality of $\mathcal{P}_t$. Hence, Algorithm 5.1 allows us to determine coarse initial lower bounds with little effort, whereas tighter lower bounds become increasingly difficult to obtain.

The quality of the lower bounds determined by Algorithm 5.1 crucially depends on the path selection in Step 3. In iteration t it seems natural to select a path P that maximises $\phi(x^t; P)$ over $\overline{\mathcal{P}} \setminus \mathcal{P}_t$. Theorem 5.2.1 implies that this choice may require the solution of an $\mathcal{NP}$-hard optimisation problem. A naive alternative is to enumerate all paths in $\overline{\mathcal{P}} \setminus \mathcal{P}_t$ and stop once a path P is found that satisfies $\phi(x^t; P) > \tau^t$. This 'first fit' method, however, suffers from two limitations. Firstly, this approach is likely to require many iterations since there is no prioritisation among the paths P that satisfy $\phi(x^t; P) > \tau^t$. Secondly, in the last (Tth) iteration of Algorithm 5.1 all paths in $\overline{\mathcal{P}} \setminus \mathcal{P}_T$ are investigated before the procedure can terminate. This implies that the algorithm needs to inspect all elements of $\overline{\mathcal{P}}$ at least once. In view of the cardinality of $\overline{\mathcal{P}}$ (see Section 5.3), this is computationally prohibitive. To alleviate both problems, we replace Step 3 of Algorithm 5.1 with the following procedure.

Algorithm 5.2 Determine $P \in \overline{\mathcal{P}} \setminus \mathcal{P}_t$ with $\phi(x^t; P) > \tau^t$.

3(a) **Initialisation.** Construct the temporal network $G = (V, E)$ with deterministic task durations $\delta = (\delta_1, \ldots, \delta_n)^\top$, where $\delta_i = \max \{\phi(x^t; \{i\}), \epsilon\}$. Here, $\{i\}$ represents a degenerate path that contains a single task $i \in V$, while ϵ denotes a small positive constant. Set $s = 1$.

3(b) **Path Selection.** Let P_s be the sth longest path in G, where the length of a path $P \in \mathcal{P}$ is defined as $\mathbb{I}_P^\top \delta$.

(i) <u>If $\mathbb{I}_{P_s}^\top \delta \leq \tau^t$ or G contains less than s paths</u>, stop: $x^* = x^t$ is an optimal allocation in $\mathcal{ARTN}$ and $\tau^* = \tau^t$ its worst-case makespan.

(ii) <u>If $\phi(x^t; P_s) > \tau^t$</u>, set $\mathcal{P}_{t+1} = \mathcal{P}_t \cup \{P_s\}$, $t \to t+1$ and go to Step 2 of Algorithm 5.1.

(iii) <u>Otherwise</u>, set $s \to s+1$ and repeat Step 3(b).

The algorithm uses $\mathbb{I}_P^\top \delta$ as an overestimator for $\phi(x^t; P)$. Indeed, we have $\mathbb{I}_P^\top \delta \geq \sum_{i \in P} \phi(x^t; \{i\})$ by definition of δ, while $\sum_{i \in P} \phi(x^t; \{i\})$ exceeds $\phi(x^t; P)$ due to (A2). Note that $\phi(x^t; \{i\})$ represents the worst-case duration of task i.

Depending on the problem instance, Algorithm 5.2 may certify the optimality of x^t without inspecting all paths in $\mathcal{P}$. Furthermore, if ϵ is sufficiently small, then the paths $P \in \mathcal{P}$ are inspected in the order of decreasing task-wise worst-case durations $\sum_{i \in P} \phi(x^t; \{i\})$. Thus, as long as these quantities approximate $\phi(x^t; P)$, $P \in \mathcal{P}$, reasonably well, one can expect Algorithm 5.1 to outperform the 'first fit' approach outlined above. Note that the s longest paths in a directed, acyclic graph $G = (V, E)$ can be enumerated in time $\mathcal{O}(|E| + s\,|V|)$, see [Epp94]. The following proposition establishes the correctness of Algorithm 5.2.

Proposition 5.4.2 *Algorithm 5.2 terminates and either correctly concludes that x^t is an optimal allocation in $\mathcal{ARTN}$ or it determines a path $P \in \overline{\mathcal{P}} \setminus \mathcal{P}_t$ with $\phi(x^t; P) > \tau^t$.*

Proof G contains a finite number of paths, and hence the algorithm terminates. In the following, we denote by P_s the sth longest path in G according to the metric defined in Step 3(b) of the algorithm. Furthermore, we assume that the algorithm terminates in iteration S.

Assume that the algorithm terminates in case (i) of Step 3(b) because G contains less than S paths. In this case, all paths $P \in \mathcal{P}$ satisfy $\tau^t \geq \phi(x^t; P)$ since otherwise the algorithm would have terminated in case (ii) of Step 3(b) of an earlier iteration. From Proposition 5.4.1 we conclude that x^t constitutes an optimal allocation in $\mathcal{ARTN}$.

If the algorithm terminates in case (i) of Step 3(b) because $\mathbb{I}_{P_S}^\top \delta \leq \tau^t$, we know that $\tau^t \geq \phi(x^t; P_s)$ for all $s < S$. Also, $\tau^t \geq \mathbb{I}_{P_s}^\top \delta$ for $s \in \{S+1, \ldots, |\mathcal{P}|\}$ since these paths are not longer than P_S. This, however, implies that for $P \in \left\{P_S, \ldots, P_{|\mathcal{P}|}\right\}$, we have

$$\tau^t \;\geq\; \mathbb{I}_P^\top \delta \;\geq\; \sum_{i \in P} \phi(x^t; \{i\}) \;\geq\; \phi(x^t; P),$$

where δ is defined in Step 3(a) of Algorithm 5.2. The second inequality follows from the definition of δ, while the third one is due to (A2). We conclude that $\tau^t \geq \phi(x^t; P)$ for all $P \in \mathcal{P}$, and hence Proposition 5.4.1 ensures that x^t is an optimal allocation in $\mathcal{ARTN}$.

If the algorithm terminates in case (ii) of Step 3(b), it has determined a task path $P_S \in \mathcal{P}$ with $\phi(x^t; P_S) > \tau^t$. We need to show that P_S is inclusion-maximal, that is, $P_S \in \overline{\mathcal{P}}$. Assume to the contrary that $P_S \in \mathcal{P} \setminus \overline{\mathcal{P}}$. Then there is a task path $P \in \mathcal{P}$ with $P \neq P_S$ and $\mathbb{I}_P \geq \mathbb{I}_{P_S}$. Since $\delta > 0$ component-wise, $\mathbb{I}_P^\top \delta = \left(\mathbb{I}_{P_S} + \mathbb{I}_{[P \setminus P_S]}\right)^\top \delta > \mathbb{I}_{P_S}^\top \delta$. Hence, P must have been considered in some iteration $s < S$. Due to (A1), however, $\phi(x^t; P) \geq \phi(x^t; P_S)$, and the algorithm must have terminated in case (ii) of Step 3(b) of that iteration because $\phi(x^t; P) \geq \phi(x^t; P_S) > \tau^t$. Since this yields a contradiction, we conclude that P_S is indeed inclusion-maximal. ∎

Note that prior to its termination, Algorithm 5.1 only provides monotonically increasing *lower* bounds on the optimal value of $\mathcal{ARTN}$. Since the intermediate allocations x^t are feasible, their worst-case makespans in $\mathcal{ARTN}$ also constitute *upper* bounds on the optimal value of $\mathcal{ARTN}$. From Theorem 5.2.1, however, we know that evaluating the worst-case makespan of x^t in $\mathcal{ARTN}$ may require the solution of an $\mathcal{NP}$-hard optimisation problem. Hence, we need

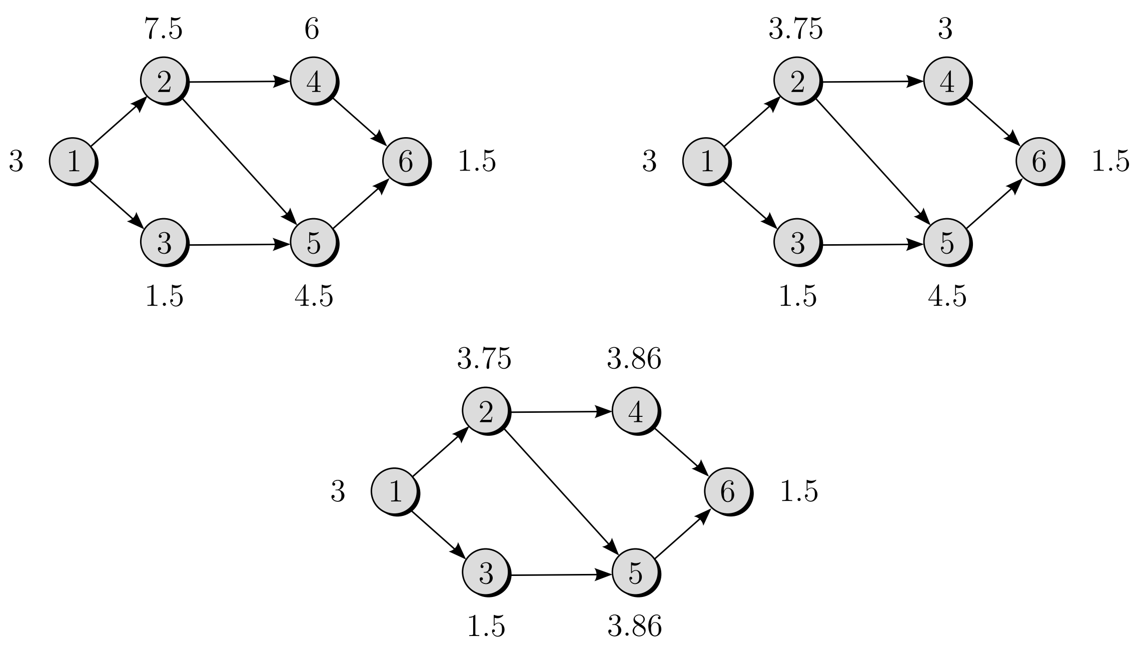


Figure 5.4: Auxiliary deterministic temporal networks generated by Algorithm 5.2. The upper left, upper right and bottom chart visualises the auxiliary graph in iteration $t = 1$, $t = 2$ and $t = 3$, respectively. Attached to each node $i \in V$ is its task-wise worst-case duration δ_i.

to pursue a different approach to generate upper bounds efficiently.

We close with an example that illustrates Algorithms 5.1 and 5.2.

Example 5.4.1 *Consider again the resource allocation problem defined in Example 5.2.1. We generate lower bounds on the optimal objective value of this problem with Algorithms 5.1 and 5.2.*

We start with <u>*Step 1*</u> *of Algorithm 5.1, in which we choose the subset $\mathcal{P}_1 = \emptyset$ and set $t = 1$.*

In <u>*Step 2*</u> *we solve the following lower bound problem $\mathcal{LARTN}_1$.*

$$
\begin{array}{ll}
\underset{\tau, x}{\text{minimise}} & \tau \\
\text{subject to} & \tau \in \mathbb{R}_+, \quad x \in \mathbb{R}^6_+ \\
& x_i \leq 1/2 \ \ \forall i \in \{1, \ldots, 6\}, \quad \sum_{i=1}^{6} x_i \leq 1.
\end{array}
$$

The optimal allocation is $x^1 = (0, 0, 0, 0, 0, 0)^\top$ with an estimated worst-case makespan of $\tau^1 = 0$.

We now enter <u>*Step 3(a)*</u> *of Algorithm 5.2. Figure 5.4 (upper left) illustrates the deterministic temporal network with worst-case task durations $\delta = (3, 7.5, 1.5, 6, 4.5, 1.5)^\top$. We set $s = 1$.*

In Step 3(b), we identify $P_1 = \{1, 2, 4, 6\}$ *as the longest path in the deterministic temporal network. This path has a task-wise worst-case duration of* $\mathbb{I}_{P_1}^\top \delta = 18$ *and a path-wise worst-case duration of* $\phi(x^1; P_1) = 16.5$*. This path therefore satisfies condition (ii) of Step 3(b), and we set* $\mathcal{P}_2 = \{\{1, 2, 4, 6\}\}$ *and* $t = 2$*.*

We are back in Step 2 of Algorithm 5.1. The new lower bound $\mathcal{LARTN}_2$ *is obtained from the following optimisation problem.*

$$\begin{aligned}
\underset{\tau,x,\lambda,\gamma}{\text{minimise}} \quad & \tau \\
\text{subject to} \quad & \tau \in \mathbb{R}_+, \quad x \in \mathbb{R}_+^6, \quad \lambda \in \mathbb{R}_+^4, \quad \gamma \in \mathbb{R}_+ \\
& \tau \geq 2(1 - x_1) + \lambda_1^1/2 + 5(1 - x_2) + \lambda_2^1/2 + 4(1 - x_4) + \lambda_4^1/2 + 1(1 - x_6) + \lambda_6^1/2 + \gamma^1, \\
& \lambda_1^1 + \gamma^1 \geq 2(1 - x_1), \quad \lambda_2^1 + \gamma^1 \geq 5(1 - x_2), \quad \lambda_4^1 + \gamma^1 \geq 4(1 - x_4), \quad \lambda_6^1 + \gamma^1 \geq 1(1 - x_6), \\
& x_i \leq 1/2 \;\; \forall\, i \in \{1, \ldots, 6\}, \quad \sum_{i=1}^{6} x_i \leq 1.
\end{aligned}$$

The optimal allocation to this problem is $x^2 = (0, 0.5, 0, 0.5, 0, 0)^\top$ *and leads to an estimated worst-case makespan of* $\tau^2 = 9.75$*.*

We enter Step 3(a) of Algorithm 5.2 again. Figure 5.4 (upper right) illustrates the deterministic temporal network with worst-case task durations $\delta = (3, 3.75, 1.5, 3, 4.5, 1.5)^\top$*. We set* $s = 1$*.*

In Step 3(b), we identify $P_1 = \{1, 2, 5, 6\}$ *as the longest path in the deterministic temporal network. This path has a task-wise worst-case duration of* $\mathbb{I}_{P_1}^\top \delta = 12.75$ *and a path-wise worst-case duration of* $\phi(x^2; P_1) = 11.25$*. This path therefore satisfies condition (ii) of Step 3(b), and we set* $\mathcal{P}_3 = \{\{1, 2, 4, 6\}, \{1, 2, 5, 6\}\}$ *and* $t = 3$*.*

We are back in Step 2 of Algorithm 5.1. The new lower bound $\mathcal{LARTN}_3$ is obtained from the following optimisation problem.

$$\begin{aligned}
\underset{\tau,x,\lambda,\gamma}{\text{minimise}} \quad & \tau \\
\text{subject to} \quad & \tau \in \mathbb{R}_+, \quad x \in \mathbb{R}^6_+, \quad \lambda \in \mathbb{R}^8_+, \quad \gamma \in \mathbb{R}^2_+ \\
& \tau \geq 2(1-x_1) + \lambda^1_1/2 + 5(1-x_2) + \lambda^1_2/2 + 4(1-x_4) + \lambda^1_4/2 + 1(1-x_6) + \lambda^1_6/2 + \gamma^1, \\
& \lambda^1_1 + \gamma^1 \geq 2(1-x_1), \quad \lambda^1_2 + \gamma^1 \geq 5(1-x_2), \quad \lambda^1_4 + \gamma^1 \geq 4(1-x_4), \quad \lambda^1_6 + \gamma^1 \geq 1(1-x_6), \\
& \tau \geq 2(1-x_1) + \lambda^2_1/2 + 5(1-x_2) + \lambda^2_2/2 + 3(1-x_5) + \lambda^2_5/2 + 1(1-x_6) + \lambda^2_6/2 + \gamma^2, \\
& \lambda^2_1 + \gamma^2 \geq 2(1-x_1), \quad \lambda^2_2 + \gamma^2 \geq 5(1-x_2), \quad \lambda^2_5 + \gamma^2 \geq 3(1-x_5), \quad \lambda^2_6 + \gamma^2 \geq 1(1-x_6), \\
& x_i \leq 1/2 \;\; \forall i \in \{1,\ldots,6\}, \quad \sum_{i=1}^{6} x_i \leq 1.
\end{aligned}$$

The optimal allocation to this problem is $x^3 = (0, 0.5, 0, 0.36, 0.14, 0)^\top$ and leads to an estimated worst-case makespan of $\tau^3 = 10.61$.

We enter Step 3(a) of Algorithm 5.2 again. Figure 5.4 (bottom) illustrates the deterministic temporal network with worst-case task durations $\delta = (3, 3.75, 1.5, 3.86, 3.86, 1.5)^\top$. We set $s = 1$.

In Step 3(b), we identify $P_1 = \{1, 2, 4, 6\}$ as the longest path in the deterministic temporal network. This path has a task-wise worst-case duration of $\mathbb{I}^\top_{P_1}\delta = 12.11$ and a path-wise worst-case duration of $\phi(x^3; P_1) = 10.61$. This path therefore satisfies condition (iii) of Step 3(b), and we set $s = 2$.

In Step 3(b), we identify $P_2 = \{1, 2, 5, 6\}$ as the second-longest path in the deterministic temporal network. Like the previous path, this path has a task-wise worst-case duration of $\mathbb{I}^\top_{P_2}\delta = 12.11$ and a path-wise worst-case duration of $\phi(x^3; P_2) = 10.61$. This path therefore also satisfies condition (iii) of Step 3(b), and we set $s = 3$.

In Step 3(b), we identify $P_3 = \{1, 3, 5, 6\}$ as the third-longest path in the deterministic temporal network. This path has a task-wise worst-case duration of $\mathbb{I}^\top_{P_3}\delta = 9.86$. This path *therefore satisfies condition (i) of Step 3(b), and we terminate with the optimal allocation* $x^* = (0, 0.50, 0, 0.36, 0.14, 0)^\top$ *and its worst-case makespan* $\tau^* = 10.61$.

5.5 Upper Bounds

Consider a task path $P \in \overline{\mathcal{P}}$, together with a partition $\{P_r\}_{r=1}^R$ that satisfies $\bigcup_{r=1}^R P_r = P$ and $P_r \cap P_q = \emptyset$ for all $r \neq q$. According to (A2), we can bound P's worst-case duration $\phi(x; P)$ from above by $\sum_{r=1}^R \phi(x; P_r)$. Intuitively, this is the case because $\sum_{r=1}^R \phi(x; P_r)$ predicts different worst-case realisations of ξ for each block P_r, whereas $\phi(x; P)$ considers the same worst-case realisation for all tasks in P. If we partition all paths $P \in \overline{\mathcal{P}}$ in this way, we obtain an upper bound on the optimal value of $\mathcal{ARTN}$. The granularity of the path partitions trades off the quality of the bound with the size of the associated bounding problem. If we use singleton partitions $\{\{i\}\}_{i \in P}$ for each path $P \in \overline{\mathcal{P}}$, for example, the associated optimisation problem can be solved efficiently as a deterministic resource allocation problem with task durations $\phi(x; \{i\})$, $i \in V$. However, this approximation is very crude since it allows each task to attain its worst-case duration individually. At the other extreme, we recover $\mathcal{ARTN}$ if we employ single-block partitions $\{P\}$ for each path $P \in \overline{\mathcal{P}}$. In the following, we develop an algorithm that iteratively advances from singleton partitions to single-block partitions. We illustrate this idea with an example.

Example 5.5.1 *Consider the temporal network in Figure 5.5(a). Assume that* $\phi(x; \{5\}) = 0$, *that is, task 5 has duration zero, and fix a resource allocation* $x \in X$. *Due to (A1) and (A2), the objective value of* $\mathcal{ARTN}$ *is the maximum of* $\phi(x; \{1, 2, 3\})$ *and* $\phi(x; \{1, 2, 4\})$. *We can bound this value from above if we replace the worst-case duration* $\phi(x; P)$ *of both paths* $P \in \{\{1, 2, 3\}, \{1, 2, 4\}\}$ *with* $\sum_{i \in P} \phi(x; \{i\})$. *To calculate this bound, let* $y \in \mathbb{R}^5_+$ *denote the vector of task start times. We minimise* y_5 *subject to*

$$y_2 \geq y_1 + \phi(x; \{1\}), \qquad y_3 \geq y_2 + \phi(x; \{2\}), \qquad y_4 \geq y_1 + \phi(x; \{1\}),$$
$$y_4 \geq y_2 + \phi(x; \{2\}), \qquad y_5 \geq y_3 + \phi(x; \{3\}), \qquad y_5 \geq y_4 + \phi(x; \{4\}).$$

This problem contains one constraint for each precedence in Figure 5.5(a). By construction, y_5 *exceeds* $\phi(x;\{1\})+\phi(x;\{2\})+\phi(x;\{3\})$ *and* $\phi(x;\{1\})+\phi(x;\{2\})+\phi(x;\{4\})$*. We thus conclude that* y_5 *bounds* $\mathcal{ARTN}$ *from above.*

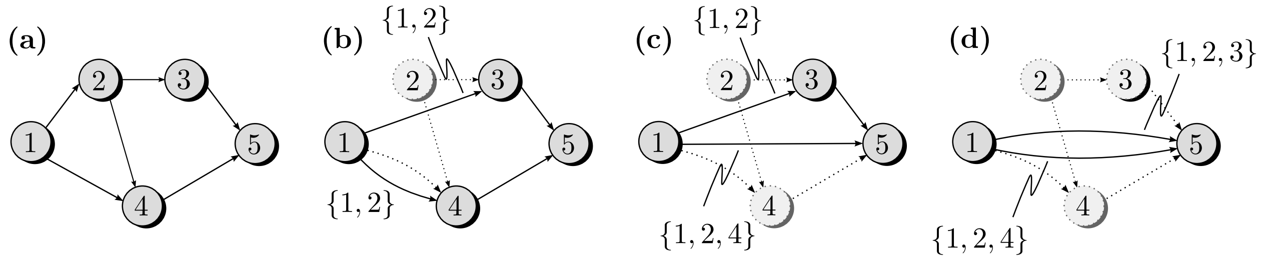


Figure 5.5: Bounding graphs for the temporal network in (a). Dotted nodes (arcs) represent redundant variables (constraints) in the bounding problem.

This upper bound relies on the assumption that different tasks can attain different worst-case durations. To obtain a tighter bound, we coarsen our path partitions. We can achieve this by replacing the precedence $(1, 2)$ *in Figure 5.5(a) with the two new precedences shown in Figure 5.5(b). The labels attached to these precedences list the tasks that need to be processed between the corresponding components of* y*. To calculate our new upper bound, we minimise* y_5 *subject to*

$$y_3 \geq y_1 + \phi(x;\{1,2\}), \qquad y_4 \geq y_1 + \phi(x;\{1,2\}),$$
$$y_5 \geq y_3 + \phi(x;\{3\}), \qquad y_5 \geq y_4 + \phi(x;\{4\})$$

and the constraints corresponding to the dotted arcs in Figure 5.5(b). In the figure, dotted arcs lie on paths that are not inclusion-maximal, and (A1) allows us to ignore the associated precedences. By construction, y_5 *exceeds* $\phi(x;\{1,2\}) + \phi(x;\{3\})$ *and* $\phi(x;\{1,2\}) + \phi(x;\{4\})$*. Hence,* y_5 *still bounds* $\mathcal{ARTN}$ *from above. Our new bound is at least as tight as the old one since* $\phi(x;\{1,2\}) \leq \phi(x;\{1\}) + \phi(x;\{2\})$*. Note that the components of* y *cannot be interpreted as task start times anymore.*

We now replace the labelled arc $(1, 4)$ *in Figure 5.5(b) with the new labelled arc in Figure 5.5(c). To obtain our new upper bound, we minimise* y_5 *subject to*

$$y_3 \geq y_1 + \phi(x;\{1,2\}), \qquad y_5 \geq y_1 + \phi(x;\{1,2,4\}), \qquad y_5 \geq y_3 + \phi(x;\{3\}).$$

Since y_5 exceeds $\phi(x;\{1,2\})+\phi(x;\{3\})$ and $\phi(x;\{1,2,4\})$, it bounds $\mathcal{ARTN}$ from above. Again, our new upper bound is at least as tight as the previous one since $\phi(x;\{1,2,4\}) \leq \phi(x;\{1,2\}) + \phi(x;\{4\})$.

If we replace the labelled arc $(1,3)$ in Figure 5.5(c), then we obtain the graph in Figure 5.5(d). The associated bounding problem minimises y_5 subject to

$$y_5 \geq y_1 + \phi(x;\{1,2,3\}), \qquad y_5 \geq y_1 + \phi(x;\{1,2,4\}).$$

This problem is equivalent to $\mathcal{ARTN}$. Note that for the path $\{1,2,3,5\}$, we iteratively generated the partitions $\{\{1\},\{2\},\{3\}\}$ in Figure 5.5(a), $\{\{1,2\},\{3\}\}$ in Figure 5.5(b)+(c) and $\{\{1,2,3\}\}$ in Figure 5.5(c).

We now formalise our approach. To simplify the exposition, we assume that $\phi(x;\{n\}) = 0$ for all $x \in X$, that is, the sink node of the network has duration zero. This can always be achieved by introducing a dummy task.

For a temporal network $G = (V,E)$, we define a sequence of *bounding graphs* $G_1, G_2, \ldots$ as follows. Each bounding graph $G_t = (V, E_t)$ is directed and acyclic with nodes V and labelled arcs E_t. The arcs are of the form (j,k,P_{jk}), where $j,k \in V$ and the label P_{jk} satisfies $P_{jk} \subseteq V \setminus \{n\}$. There can be multiple arcs between j and k as long as they have different labels. The networks in Figure 5.5 constitute bounding graphs if we attach the label $\{j\}$ to the each unlabelled arc from j to k.

We associate with G_t the following *bounding problem.*

$$\min_{\substack{x\in X,\\ y\in\mathbb{R}^n_+}} \{y_n \,:\, y_k - y_j \geq \phi(x;P_{jk}) \;\; \forall\,(j,k,P_{jk}) \in E_t\} \qquad (\mathcal{UARTN}_t)$$

$\mathcal{UARTN}_t$ assigns a variable y_j to every node $j \in V$. The constraints ensure that y_k exceeds y_j by at least $\phi(x;P_{jk})$ time units if $(j,k,P_{jk}) \in E_t$. In Example 5.5.1 we formulated $\mathcal{UARTN}_t$ for the four bounding graphs in Figure 5.5.

For a bounding graph G_t, we say that $P \in \overline{\mathcal{P}}$ is an *induced path* if every feasible solution (x, y) to G_t's bounding problem satisfies $y_n \geq \phi(x; P)$. To obtain an upper bound on $\mathcal{ARTN}$, we are interested in bounding graphs that induce all paths $P \in \overline{\mathcal{P}}$. Formally, we define the set of induced paths as

$$\mathcal{P}(G_t) = \Big\{ P \in \overline{\mathcal{P}} \, : \, \exists \, \{(i_r, i_{r+1}, P_r)\}_{r=1}^{R} \subseteq E_t \\ \text{such that } \; i_{R+1} = n \; \text{ and } \; (P \setminus \{n\}) = \bigcup_{r=1}^{R} P_r \Big\}.$$

Hence, $P \in \mathcal{P}(G_t)$ if the tasks in $P \setminus \{n\}$ are contained in the union of arc labels on a path in G_t that ends at the sink node n. Intuitively, y_n exceeds $\phi(x; P)$ because there is a partition $\{P_r\}_{r=1}^{R}$ of $P \setminus \{n\}$ such that $y_n \geq \sum_{r=1}^{R} \phi(x; P_r)$. Note that we can ignore the sink node n in this consideration since its duration is zero. The following lemma makes this argument explicit.

Lemma 5.5.1 (Induced Paths) *If $P \in \mathcal{P}(G_t)$, then any feasible solution (x, y) to $\mathcal{UARTN}_t$ satisfies $y_n \geq \phi(x; P)$.*

Proof By definition of $\mathcal{P}(G_t)$, there is $\{(i_r, i_{r+1}, P_r)\}_{r=1}^{R} \subseteq E_t$ with $i_{R+1} = n$ and $(P \setminus \{n\}) = \bigcup_{r=1}^{R} P_r$. We thus have

$$y_n \overset{\text{(a)}}{\geq} y_n - y_{i_1} = \sum_{r=1}^{R} (y_{i_{r+1}} - y_{i_r}) \overset{\text{(b)}}{\geq} \sum_{r=1}^{R} \phi(x; P_r) \overset{\text{(c)}}{\geq} \phi(x; P \setminus \{n\}) \overset{\text{(d)}}{=} \phi(x; P),$$

where (a) follows from non-negativity of y, (b) from the fact that (x, y) is feasible in $\mathcal{UARTN}_t$, and (c) and (d) from (A1), (A2) and $\phi(x; \{n\}) = 0$. ∎

As an illustration of induced paths, consider the path $\{1, 2, 4, 5\}$ in Example 5.5.1. It is induced by G_1 via $\{(1, 2, \{1\}), (2, 4, \{2\}), (4, 5, \{4\})\}$, by G_2 via $\{(1, 4, \{1, 2\}), (4, 5, \{4\})\}$, and by G_3 and G_4 via $\{(1, 5, \{1, 2, 4\})\}$, see Figure 5.5. Lemma 5.5.1 implies that the objective value of any feasible solution (x, y) to $\mathcal{UARTN}_t$ provides an upper bound on the worst-case makespan of x with respect to all induced task paths. We conclude that $\mathcal{UARTN}_t$ bounds $\mathcal{ARTN}$ from above if $\overline{\mathcal{P}} \subseteq \mathcal{P}(G_t)$.

An initial upper bound on $\mathcal{ARTN}$ is obtained from $\mathcal{UARTN}_1$ where

$$G_1 = (V, E_1) \quad \text{with} \quad E_1 = \{(j, k, \{j\}) \, : \, (j, k) \in E\} \, . \tag{5.8}$$

$\mathcal{UARTN}_1$ comprises one constraint for every arc $(j, k, P_{jk}) \in E_1$. Since E_1 contains $|E|$ arcs, $\mathcal{UARTN}_1$ is a tractable optimisation problem. The following lemma shows that $\mathcal{UARTN}_1$ bounds $\mathcal{ARTN}$ from above.

Lemma 5.5.2 (Initial Bound) $\overline{\mathcal{P}} \subseteq \mathcal{P}(G_1)$ *for* G_1 *defined in (5.8).*

Proof Consider any path $P = \{i_1 = 1, i_2, \ldots, i_{R+1} = n\} \in \overline{\mathcal{P}}$ with $(i_r, i_{r+1}) \in E$ for $r = 1, \ldots, R$. For $P_r = \{i_r\}$, $r = 1, \ldots, R$, we have $\{(i_r, i_{r+1}, P_r)\}_{r=1}^R \subseteq E_1$ and $(P \setminus \{n\}) = \bigcup_{r=1}^R P_r$, so that $P \in \mathcal{P}(G_1)$. ∎

Figure 5.5(a) visualises G_1 for the temporal network in Example 5.5.1. The initial bounding graph approximates the worst-case duration $\phi(x; P)$ of every path $P \in \overline{\mathcal{P}}$ by the duration $\sum_{i \in P} \phi(x; \{i\})$ of the singleton partition $\{\{i\}\}_{i \in P}$. If this approximation is tight, then $\mathcal{UARTN}_1$ and $\mathcal{ARTN}$ are equivalent. This is the case, for example, if all task durations depend on disjoint parts of ξ that are not related to each other through Ξ. In general, however, $\phi(x; P) < \sum_{i \in P} \phi(x; \{i\})$, and the optimal value of $\mathcal{UARTN}_1$ constitutes a strict upper bound on the optimal value of $\mathcal{ARTN}$.

By suitably transforming the graph G_1, we can coarsen the path partitions to tighten the upper bound provided by $\mathcal{UARTN}_1$.

Definition 5.5.1 (Replacements) *For a bounding graph* $G_t = (V, E_t)$ *we construct* $G_{t+1} = (V, E_{t+1})$ *via the following two types of replacements.*

1. ***Predecessor Replacement.*** G_{t+1} *results from a* predecessor replacement *of* $(j, k, P_{jk}) \in E_t$ *if* $j \neq 1$ *and*

$$E_{t+1} = E_t \setminus \{(j, k, P_{jk})\} \cup \bigcup_{\substack{i \in V, P_{ij} \in \mathcal{P}: \\ (i,j,P_{ij}) \in E_t}} \{(i, k, P_{ij} \cup P_{jk})\} \, .$$

2. ***Successor Replacement.*** *G_{t+1} results from a* successor replacement *of $(j, k, P_{jk}) \in E_t$ if $k \neq n$ and*

$$E_{t+1} = E_t \setminus \{(j, k, P_{jk})\} \cup \bigcup_{\substack{l \in V, P_{kl} \in \mathcal{P}: \\ (k,l,P_{kl}) \in E_t}} \{(j, l, P_{jk} \cup P_{kl})\}\,.$$

The two replacements are illustrated in Figures 5.6 and 5.7. We call $(j, k, P_{jk}) \in E_t$ *replaceable* if it qualifies for either of the two replacements. The application of a replacement to $(j, k, P_{jk}) \in E_t$ reduces the approximation error for every path $P \in \mathcal{P}(G_t)$ whose partition $\{P_r\}_{r=1}^R$ contains the block P_{jk}. At the same time, however, the number of arcs in the resulting bounding graph (and hence the size of the bounding problem) typically increases. In Example 5.5.1 we applied successor replacements to $(1, 2, \{1\}) \in E_1$, $(1, 4, \{1, 2\}) \in E_2$ and $(1, 3, \{1, 2\}) \in E_3$. As the result of a replacement, some nodes and/or arcs in the bounding graph may become redundant, see Figure 5.5. We will identify such redundancies at the end of this section.

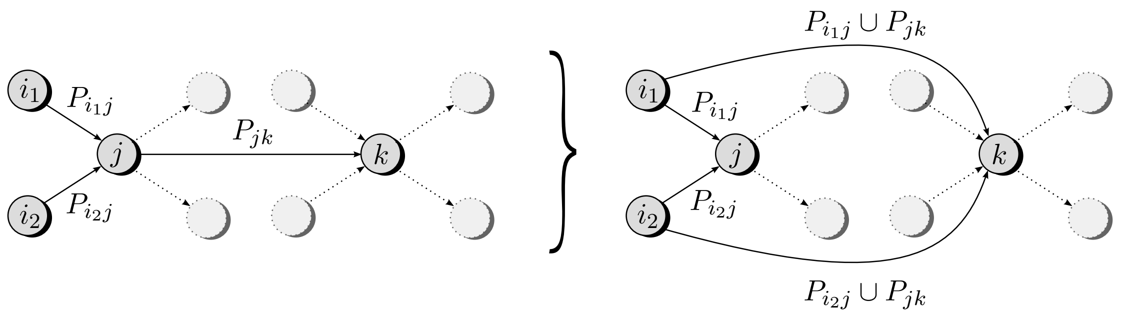


Figure 5.6: Predecessor replacement of (j, k, P_{jk}) with two predecessor nodes.

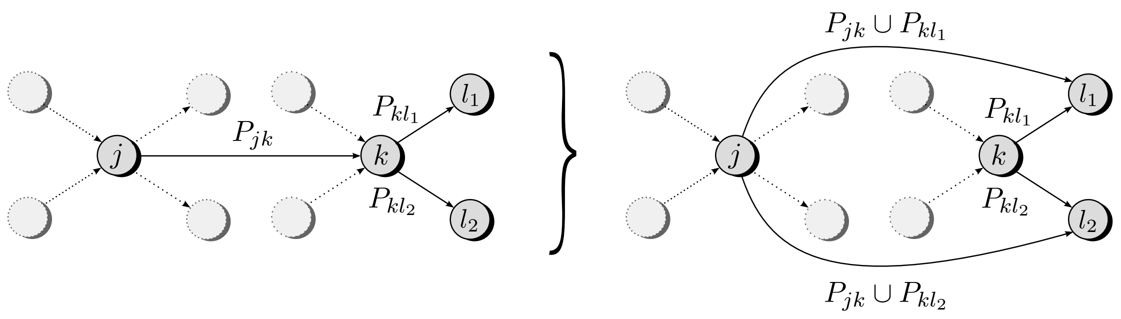


Figure 5.7: Successor replacement of (j, k, P_{jk}) with two successor nodes.

From now on, we assume that $(G_t)_t$ is a sequence of bounding graphs where G_1 is defined in (5.8) and $G_2, G_3, \ldots$ result from an iterated application of replacements in the sense of

Definition 5.5.1. In this case, the label P_{jk} of an arc $(j, k, P_{jk}) \in E_t$ contains precisely the tasks on a path from j to k (excluding k) in the temporal network G.

Lemma 5.5.3 *For each arc $(j, k, P_{jk}) \in E_t$ the temporal network G contains a directed path $\{(l_r, l_{r+1})\}_{r=1}^{R} \subseteq E$ with $R \geq 1$, $(l_1, l_{R+1}) = (j, k)$ and $P_{jk} = \{l_1, \ldots, l_R\}$.*

Proof We prove the assertion by induction on t. By construction of G_1, the assertion holds for $t = 1$. Assume now that the assertion holds for G_t and that G_{t+1} results from a predecessor replacement of $(j, k, P_{jk}) \in E_t$ (an analogous argument can be made for successor replacements). According to Definition 5.5.1, any new arc in $E_{t+1} \backslash E_t$ must be of the form (i, k, P_{ik}), and E_t must contain an arc $(i, j, P_{ij}) \in E_t$ with $P_{ij} \cup P_{jk} = P_{ik}$. Since the assertion holds for G_t, G contains directed paths $\{(l_r, l_{r+1})\}_{r=1}^{R}, \left\{(l'_r, l'_{r+1})\right\}_{r=1}^{R'} \subseteq E$ with $(l_1, l_{R+1}) = (i, j)$, $(l'_1, l'_{R'+1}) = (j, k)$, $P_{ij} = \{l_1, \ldots, l_R\}$ and $P_{jk} = \{l'_1, \ldots, l'_{R'}\}$. Since $l_{R+1} = l'_1$, we can connect both paths to prove the assertion for (i, k, P_{ik}). Since the arc $(j, k, P_{jk}) \in E_t$ was chosen arbitrarily, the assertion of the lemma follows. ∎

The next lemma shows that replacements preserve the upper bound property.

Lemma 5.5.4 (Bound Preservation) *If $\overline{\mathcal{P}} \subseteq \mathcal{P}(G_t)$, then $\overline{\mathcal{P}} \subseteq \mathcal{P}(G_{t+1})$.*

Proof Choose any path $P \in \overline{\mathcal{P}}$. By assumption, $P \in \mathcal{P}(G_t)$, that is, there exists a set of arcs $\{(i_r, i_{r+1}, P_r)\}_{r=1}^{R} \subseteq E_t$ with $i_{R+1} = n$ and $(P \setminus \{n\}) = \bigcup_{r=1}^{R} P_r$. We show that $P \in \mathcal{P}(G_{t+1})$. Assume that G_{t+1} results from a predecessor replacement of $(j, k, P_{jk}) \in E_t$; the proof is widely parallel for successor replacements.

If $(j, k) \neq (i_r, i_{r+1})$ for all $r \in \{1, \ldots, R\}$, then $P \in \mathcal{P}(G_{t+1})$ is vacuously satisfied. Hence, assume that $(j, k) = (i_s, i_{s+1})$ for some $s \in \{1, \ldots, R\}$. Since 1 is the unique source of G (see Section 1.1) and $P \in \overline{\mathcal{P}}$, we have $1 \in P$. Lemma 5.5.3 then implies that $i_1 = 1$. Hence, $s \neq 1$ since (i_1, i_2, P_1) does not qualify for a predecessor replacement. Let $i'_r = i_r$ for $r = 1, \ldots, s-1$ and $i'_r = i_{r+1}$ for $r = s, \ldots, R$. Similarly, let $P'_r = P_r$ for $r = 1, \ldots, s-2$ (if $s > 2$), $P'_{s-1} = P_{s-1} \cup P_s$ and $P'_r = P_{r+1}$ for $r = s, \ldots, R-1$. We have that $\left\{(i'_r, i'_{r+1}, P'_r)\right\}_{r=1}^{R-1} \subseteq E_{t+1}$,

$i'_R = n$ and $(P \setminus \{n\}) = \bigcup_{r=1}^{R-1} P'_r$, which ensures that $P \in \mathcal{P}(G_{t+1})$. Since P was chosen arbitrarily, the assertion follows. ∎

We can now prove that the proposed replacements result in a monotonically non-increasing, convergent sequence of upper bounds on $\mathcal{ARTN}$.

Proposition 5.5.1 *Let (x^t, y^t) denote an optimal solution to $\mathcal{UARTN}_t$. Then:*

(a) For every t, x^t is a feasible allocation in $\mathcal{ARTN}$ and y^t_n is an upper bound on the worst-case makespan of x^t in $\mathcal{ARTN}$.

(b) There is $T \in \mathbb{N}$ such that there are no replaceable arcs in G_T. For this T, x^T is an optimal allocation in $\mathcal{ARTN}$ and y^T_n is the worst-case makespan of x^T in $\mathcal{ARTN}$.

(c) The sequence $\{y^t_n\}_{t=1}^{T}$ is monotonically non-increasing.

Proof By construction, x^t constitutes a feasible allocation for every t. Due to Lemma 5.5.1, assertion (a) is therefore satisfied if $\overline{\mathcal{P}} \subseteq \mathcal{P}(G_t)$ for every t. Employing Lemmas 5.5.2 and 5.5.4, this follows by induction on t.

As for (b), we recall that G_1 is acyclic. Hence, we can relabel the nodes of G_1 such that all $(j, k, P_{jk}) \in E_1$ satisfy $j < k$. Every replacement removes one arc $(j, k, P_{jk}) \in E_t$, $t = 1, 2, \ldots$, and adds less than $|E_t|$ arcs (i, l, P_{il}) with $i \leq j$ and $l \geq k$, where one of these inequalities is strict. Since all $(j, k, P_{jk}) \in E_t$ satisfy $1 \leq j, k \leq n$, there is $T \in \mathbb{N}$ such that there are no replaceable arcs in G_T.

All arcs in E_T are of the form $(1, n, P_{1n})$ for some $P_{1n} \subseteq V \setminus \{n\}$ since otherwise, further replacements would be possible. Hence, $\mathcal{UARTN}_T$ is equivalent to

$$\min_{x \in X} \max_{(1,n,P_{1n}) \in E_T} \phi(x; P_{1n}).$$

We have $\overline{\mathcal{P}} \subseteq \{P_{1n} \in \mathcal{P} \,:\, (1, n, P_{1n}) \in E_T\} \subseteq \mathcal{P}$ due to Lemma 5.5.3 and part (a) of this proof. Hence, $\mathcal{UARTN}_T$ is equivalent to $\mathcal{ARTN}$, and claim (b) follows.

To prove (c), we first show that if (x, y) is feasible in $\mathcal{UARTN}_t$, $t \in \{1, \ldots, T-1\}$, then it is also feasible in $\mathcal{UARTN}_{t+1}$. Assume that G_{t+1} is obtained from a predecessor replacement of $(j, k, P_{jk}) \in E_t$. The argument is widely parallel for successor replacements. $\mathcal{UARTN}_{t+1}$ results from $\mathcal{UARTN}_t$ by replacing the constraint $y_k - y_j \geq \phi(x; P_{jk})$ with new constraints of the form $y_k - y_i \geq \phi(x; P_{ij} \cup P_{jk})$ for $i \in V$ and $P_{ij} \subseteq V \setminus \{n\}$ with $(i, j, P_{ij}) \in E_t$. These new constraints are less restrictive, however, because

$$y_k - y_i = (y_k - y_j) + (y_j - y_i) \overset{\text{(i)}}{\geq} \phi(x; P_{ij}) + \phi(x; P_{jk}) \overset{\text{(ii)}}{\geq} \phi(x; P_{ij} \cup P_{jk}).$$

Here, (i) follows from the fact that (x, y) is feasible in $\mathcal{UARTN}_t$, while (ii) is due to (A2). Hence, (x, y) is feasible in $\mathcal{UARTN}_{t+1}$, too. Since $\mathcal{UARTN}_t$ and $\mathcal{UARTN}_{t+1}$ share the same objective function, assertion (c) follows. ∎

Proposition 5.5.1 provides the justification for the following algorithm.

Algorithm 5.3 Convergent upper bounds on $\mathcal{ARTN}$.

1. **Initialisation.** Construct G_1 as defined in (5.8). Set $t = 1$.
2. **Bounding Problem.** Find an optimal solution (x^t, y^t) to $\mathcal{UARTN}_t$.
3. **Replacement.** Choose a replaceable arc $(j, k, P_{jk}) \in E_t$.
   (a) <u>If there is no such arc</u>, terminate: $x^* = x^t$ is an optimal allocation in $\mathcal{ARTN}$ and $y_n^* = y_n^t$ is the worst-case makespan of x^* in $\mathcal{ARTN}$.
   (b) <u>Otherwise</u>, construct G_{t+1} by applying a replacement to arc (j, k, P_{jk}), set $t \to t+1$ and go to Step 2.

Algorithm 5.3 does not prescribe the choice of any specific replacement. We will discuss a selection scheme below. Before that, we summarise the following algorithm properties which are a direct consequence of Proposition 5.5.1.

Corollary 5.5.1 *Algorithm 5.3 terminates with an optimal resource allocation x^* in $\mathcal{ARTN}$ and its worst-case makespan y^*_n. Moreover, $\{x^t\}_{t=1}^T$ represents a sequence of feasible allocations in $\mathcal{ARTN}$ and $\{y^t_n\}_{t=1}^T$ a monotonically non-increasing sequence of upper bounds on their objective values in $\mathcal{ARTN}$.*

By combining Algorithms 5.1 and 5.3, we obtain monotonically convergent lower and upper bounds on the optimal value of $\mathcal{ARTN}$, together with feasible allocations $x^t \in X$ whose worst-case makespans are bracketed by these bounds. This provides us with feasible allocations that converge to the optimal allocation and whose suboptimality can be quantified at any iteration.

The tractability assumption (A3) allows us to reduce the set of meaningful replacement candidates in Step 3 of Algorithm 5.3 as follows.

Proposition 5.5.2 *Assume that (A3) holds, and let (x^t, y^t) denote any optimal solution to $\mathcal{UARTN}_t$. We have:*

(a) If $y^t_k - y^t_j > \phi(x^t; P_{jk})$ for some *replaceable arc $(j, k, P_{jk}) \in E_t$, then $\mathcal{UARTN}_{t+1}$ with G_{t+1} obtained from G_t by replacing (j, k, P_{jk}) has an optimal value of y^t_n, too.*

(b) If $y^t_k - y^t_j > \phi(x^t; P_{jk})$ for all *replaceable arcs $(j, k, P_{jk}) \in E_t$, then $\mathcal{UARTN}_s$ with $s > t$ and G_s obtained from G_t by any sequence of replacements has an optimal value of y^t_n, too.*

Remark 5.5.1 *According to assertion (a), replacing any arc $(j, k, P_{jk}) \in E_t$ that satisfies the described condition leads to the same upper bound as $\mathcal{UARTN}_t$. Since we intend to reduce this bound, we may disregard all such replacement candidates in Step 3 of Algorithm 5.3. Part (b) describes a condition under which x^t is the optimal allocation and y^t_n the optimal value of $\mathcal{ARTN}$.*

Proof of Proposition 5.5.2 Assume that (a) is false, that is, $y^t_k - y^t_j > \phi(x^t; P_{jk})$, but there is a feasible solution (x^{t+1}, y^{t+1}) to $\mathcal{UARTN}_{t+1}$ that has an objective value smaller than y^t_n.

From the argumentation in the proof of Proposition 5.5.1 (c) we know that (x^t, y^t) is feasible in $\mathcal{UARTN}_{t+1}$. Due to (A3),

$$(x^\lambda, y^\lambda) = \lambda(x^{t+1}, y^{t+1}) + (1-\lambda)(x^t, y^t) \quad \text{for } \lambda \in (0,1]$$

is also feasible for $\mathcal{UARTN}_{t+1}$ and has an objective value smaller than y^t_n. We show that for small λ, (x^λ, y^λ) is feasible in $\mathcal{UARTN}_t$, too. Since $E_t \setminus E_{t+1} = \{(j,k,P_{jk})\}$, we only need to show that $y^\lambda_k - y^\lambda_j \geq \phi(x^\lambda; P_{jk})$. For sufficiently small λ, this follows from continuity of $\phi(\cdot; P_{jk})$ in its first component, which is a consequence of (A3), and the fact that $y^t_k - y^t_j > \phi(x^t; P_{jk})$. Since $\mathcal{UARTN}_t$ and $\mathcal{UARTN}_{t+1}$ share the same objective function, this implies that (x^t, y^t) is not optimal for $\mathcal{UARTN}_t$. Thus, our assumption is false and (a) must be true.

As for (b), let us now assume that $y^t_k - y^t_j > \phi(x^t; P_{jk})$ for all replaceable arcs $(j,k,P_{jk}) \in E_t$. In this case, assertion (a) guarantees that (x^t, y^t) remains optimal for G_{t+1} if G_{t+1} results from applying one replacement to G_t. Assume that G_{t+1} results from a predecessor replacement of $(j,k,P_{jk}) \in E_t$ (the proof for successor replacements is analogous). We then have

$$\begin{aligned}(y^t_k - y^t_i) = (y^t_k - y^t_j) + (y^t_j - y^t_i) &\overset{\text{(i)}}{>} \phi(x^t; P_{ij}) + \phi(x^t; P_{jk}) \\ &\overset{\text{(ii)}}{\geq} \phi(x^t; P_{ij} \cup P_{jk}) \qquad\qquad \forall\, (i,j,P_{ij}) \in E_t,\end{aligned}$$

where (i) follows from the assumption and (ii) is due to (A2). Hence, the condition described in assertion (b) is satisfied for all new arcs $(i,k,P_{ij} \cup P_{jk}) \in E_{t+1}$ as well. An iterated application of this argument shows that assertion (b) remains valid for $\mathcal{UARTN}_s$ with G_s obtained from applying any sequence of predecessor and/or successor replacements to G_t. This implies that $\mathcal{UARTN}_s$ has an optimal value of y^t_n, and thus the claim follows. ∎

$\mathcal{UARTN}_t$ may have several optimal solutions, and the conditions in Proposition 5.5.2 may only be satisfied for some of them. If an optimal solution (x^t, y^t) to $\mathcal{UARTN}_t$ does not satisfy the condition in Proposition 5.5.2 (a) for $(j,k,P_{jk}) \in E_t$, then we can use y^t_n to check whether other optimal solutions (x', y') satisfy the condition. Indeed, this is the case if

$$\max_{\substack{x \in X, \\ y \in \mathbb{R}^n_+}} \left\{ (y_k - y_j) - \phi(x; P_{jk}) \; : \; y_n = y^t_n, \; y_q - y_p \geq \phi(x; P_{pq}) \;\; \forall \, (p, q, P_{pq}) \in E_t \right\} > 0. \tag{5.9}$$

Similarly, Proposition 5.5.2 (b) implies that x^t is an optimal allocation for $\mathcal{ARTN}$ if all replacement candidates $(j, k, P_{jk}) \in E_t$ satisfy (5.9). Unfortunately, evaluating the left-hand side of (5.9) is as difficult as solving $\mathcal{UARTN}_t$, and it is prohibitive to compute it for all $(j, k, P_{jk}) \in E_t$. If we fix x to x^t and optimise (5.9) only over y, however, the maximisation can be computed in time $\mathcal{O}(|E_t|)$ by a combined forward and backward calculation, see [DH02]. In this case, however, we might not identify all replacement candidates that satisfy the conditions of Proposition 5.5.2.

Although Proposition 5.5.2 reduces the set of potential replacement candidates, it provides no criterion for selecting specific arcs to be replaced. Ideally, one would choose a replacement that leads to the largest reduction of the upper bound. This approach is computationally prohibitive, however, since it requires the solution of bounding problems for all replacement candidates. Likewise, 'first fit' approaches are unsuited due to similar reasons as in Section 5.4. We propose to choose a replacement for G_t that leads to the largest reduction of the upper bound when x is fixed to the optimal allocation of $\mathcal{UARTN}_t$. Like the optimisation of (5.9) for fixed x, this evaluation requires time $\mathcal{O}(|E_t|)$ and can hence be implemented efficiently. At the same time, however, this selection scheme is likely to lead to better results than naive 'first fit' approaches.

We close this section with an investigation of redundant nodes and arcs in the bounding graphs G_t. We call an arc $(j, k, P_{jk}) \in E_t$ *redundant* if it can be removed from E_t without changing the set of induced paths $\mathcal{P}(G_t)$. The following proposition lists sufficient conditions for redundancy.

Proposition 5.5.3 *An arc $(j, k, P_{jk}) \in E_t$ is redundant if one of the following conditions is met:*

1. *There is another arc (j, k, P'_{jk}) with $P_{jk} \subseteq P'_{jk}$, $P_{jk} \neq P'_{jk}$.*

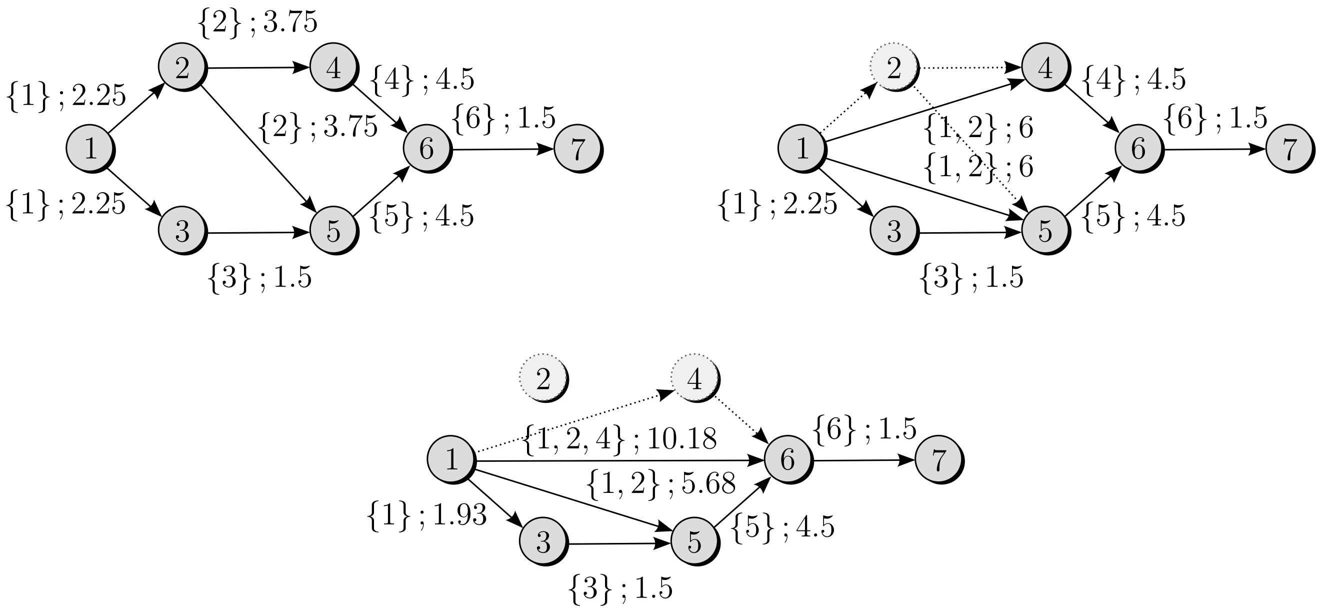


Figure 5.8: Bounding graphs generated by Algorithm 5.3. The upper left, upper right and bottom chart visualises the bounding graph in iteration $t = 1$, $t = 2$ and $t = 3$, respectively. Attached to each arc $(i, j, P_{ij}) \in E_t$ is its label P_{ij} and its worst-case duration $\phi(x^t; P_{ij})$.

2. Node j has no incoming arcs in G_t and $j \neq 1$.

3. Node k has no outgoing arcs in G_t and $k \neq n$.

The proof of this proposition is straightforward, and we omit it for the sake of brevity. Proposition 5.5.3 allows us to identify redundant nodes as well: node $i \in V$ is redundant in G_t if all of its incoming and outgoing arcs are redundant.

We close this section with an example that illustrates Algorithm 5.3.

Example 5.5.2 *Consider again the problem instance from Examples 5.3.2 and 5.4.1. We generate upper bounds on the optimal objective value of this problem with Algorithm 5.3.*

We start with Step 1, where we construct the bounding graph G_1 shown in Figure 5.8 (upper left). Note that we added a dummy sink node 7 and an artificial precedence between nodes 6 and 7 so that the last task (i.e., task 7) has duration zero. Since none of the arcs in the bounding graph G_1 satisfies the conditions of Proposition 5.5.3, we cannot identify any arc or node in G_1 as redundant. We set $t = 1$.

In Step 2 we solve the upper bound problem $\mathcal{UARTN}_1$*:*

$$
\begin{array}{ll}
\underset{x,y,\lambda,\gamma}{\text{minimise}} & y_7 \\[1em]
\text{subject to} & x \in \mathbb{R}^6_+, \quad y \in \mathbb{R}^7_+, \quad \lambda \in \mathbb{R}^8_+, \quad \gamma \in \mathbb{R}^8_+ \\[1em]
& y_2 \geq y_1 + 2(1-x_1) + \lambda^1_1/2 + \gamma^1, \quad \lambda^1_1 + \gamma^1 \geq 2(1-x_1), \\
& y_3 \geq y_1 + 2(1-x_1) + \lambda^2_1/2 + \gamma^2, \quad \lambda^2_1 + \gamma^2 \geq 2(1-x_1), \\
& y_4 \geq y_2 + 5(1-x_2) + \lambda^3_2/2 + \gamma^3, \quad \lambda^3_2 + \gamma^3 \geq 5(1-x_2), \\
& y_5 \geq y_2 + 5(1-x_2) + \lambda^4_2/2 + \gamma^4, \quad \lambda^4_2 + \gamma^4 \geq 5(1-x_2), \\
& y_5 \geq y_3 + 1(1-x_3) + \lambda^5_3/2 + \gamma^5, \quad \lambda^5_3 + \gamma^5 \geq 1(1-x_3), \\
& y_6 \geq y_4 + 4(1-x_4) + \lambda^6_4/2 + \gamma^6, \quad \lambda^6_4 + \gamma^6 \geq 4(1-x_4), \\
& y_6 \geq y_5 + 3(1-x_5) + \lambda^7_5/2 + \gamma^7, \quad \lambda^7_5 + \gamma^7 \geq 3(1-x_5), \\
& y_7 \geq y_6 + 1(1-x_6) + \lambda^8_6/2 + \gamma^8, \quad \lambda^8_6 + \gamma^8 \geq 1(1-x_6), \\[1em]
& x_i \leq 1/2 \ \ \forall\, i \in \{1,\dots,6\}, \quad \sum_{i=1}^{6} x_i \leq 1.
\end{array}
$$

The optimal allocation and task start schedule are given by $x^1 = (0.25, 0.5, 0, 0.25, 0, 0)^\top$ *and* $y^1 = (0, 2.25, 4.5, 6, 6, 10.5, 12)^\top$*, respectively. The estimated worst-case makespan is* 12.

According to Proposition 5.5.2, we should not replace $(1, 3, \{1\}) \in E_1$ *because* $\phi(x^1; \{1\}) = 2.25$ *but* $y^1_3 - y^1_1 = 4.5$*. If we apply the extended check described in (5.9), we see that we should not replace* $(3, 5, \{3\}) \in E_1$ *either. For ease of exposition, we use a 'first-fit' approach here and apply a forward replacement to the arc* $(1, 2, \{1\}) \in E_1$ *in Step 3. The new bounding graph* G_2 *is visualised in Figure 5.8 (upper right). Note that the arcs* $(2, 4, \{2\}), (2, 5, \{2\}) \in E_2$ *satisfy the second condition of Proposition 5.5.3 and are therefore redundant. As a result, node* 2 *is redundant as well. We set* $t = 2$*.*

Back in Step 2*, we solve the upper bound problem* $\mathcal{UARTN}_2$*:*

$$\begin{aligned}
\underset{x,y,\lambda,\gamma}{\text{minimise}} \quad & y_7 \\
\text{subject to} \quad & x \in \mathbb{R}^6_+, \quad y \in \mathbb{R}^6_+, \quad \lambda \in \mathbb{R}^9_+, \quad \gamma \in \mathbb{R}^7_+ \\
& y_4 \geq y_1 + 2(1 - x_1) + \lambda^1_1/2 + 5(1 - x_2) + \lambda^1_2/2 + \gamma^1, \\
& \lambda^1_1 + \gamma^1 \geq 2(1 - x_1), \quad \lambda^1_2 + \gamma^1 \geq 5(1 - x_2), \\
& y_5 \geq y_1 + 2(1 - x_1) + \lambda^2_1/2 + 5(1 - x_2) + \lambda^2_2/2 + \gamma^2, \\
& \lambda^2_1 + \gamma^2 \geq 2(1 - x_1), \quad \lambda^2_2 + \gamma^2 \geq 5(1 - x_2), \\
& y_3 \geq y_1 + 2(1 - x_1) + \lambda^3_1/2 + \gamma^3, \quad \lambda^3_1 + \gamma^3 \geq 2(1 - x_1), \\
& y_5 \geq y_3 + 1(1 - x_3) + \lambda^4_3/2 + \gamma^4, \quad \lambda^4_3 + \gamma^4 \geq 1(1 - x_3), \\
& y_6 \geq y_4 + 4(1 - x_4) + \lambda^5_4/2 + \gamma^5, \quad \lambda^5_4 + \gamma^5 \geq 4(1 - x_4), \\
& y_6 \geq y_5 + 3(1 - x_5) + \lambda^6_5/2 + \gamma^6, \quad \lambda^6_5 + \gamma^6 \geq 3(1 - x_5), \\
& y_7 \geq y_6 + 1(1 - x_6) + \lambda^7_6/2 + \gamma^7, \quad \lambda^7_6 + \gamma^7 \geq 1(1 - x_6), \\
& x_i \leq 1/2 \ \ \forall\, i \in \{1, \ldots, 6\}, \quad \sum_{i=1}^{6} x_i \leq 1.
\end{aligned}$$

The optimal allocation is $x^2 = (0.25, 0.5, 0, 0.25, 0, 0)^\top$*, and the optimal vector* y^2 *is given by* $y^2_1 = 0$*,* $y^2_3 = 4.5$*,* $y^2_4 = y^2_5 = 6$*,* $y^2_6 = 10.5$ *and* $y^2_7 = 12$*. The estimated worst-case makespan is* 12*. Note that neither the optimal allocation nor the upper bound changed. This is due to the fact that our uncertainty set allows two tasks to attain their worst-case durations simultaneously.*

As before, Proposition 5.5.2 indicates that we should not replace the arcs $(1, 3, \{1\}), (3, 5, \{3\}) \in E_2$*. We apply a forward replacement to the arc* $(1, 4, \{1, 2\}) \in E_2$ *in* Step 3*. The new bounding graph* G_3 *is visualised in Figure 5.8 (bottom). Due to the replacement, the arc* $(4, 6, \{4\}) \in E_3$ *and node* 4 *have become redundant. We set* $t = 3$*.*

Back in Step 2, we solve the upper bound problem $\mathcal{UARTN}_3$*:*

$$\begin{aligned}
\underset{x,y,\lambda,\gamma}{\text{minimise}} \quad & y_7 \\
\text{subject to} \quad & x \in \mathbb{R}^6_+, \quad y \in \mathbb{R}^5_+, \quad \lambda \in \mathbb{R}^9_+, \quad \gamma \in \mathbb{R}^6_+ \\
& y_6 \geq y_1 + 2(1-x_1) + \lambda^1_1/2 + 5(1-x_2) + \lambda^1_2/2 + 4(1-x_4) + \lambda^1_4 + \gamma^1, \\
& \lambda^1_1 + \gamma^1 \geq 2(1-x_1), \quad \lambda^1_2 + \gamma^1 \geq 5(1-x_2), \quad \lambda^1_4 + \gamma^1 \geq 4(1-x_4), \\
& y_5 \geq y_1 + 2(1-x_1) + \lambda^2_1/2 + 5(1-x_2) + \lambda^2_2/2 + \gamma^2, \\
& \lambda^2_1 + \gamma^2 \geq 2(1-x_1), \quad \lambda^2_2 + \gamma^2 \geq 5(1-x_2), \\
& y_3 \geq y_1 + 2(1-x_1) + \lambda^3_1/2 + \gamma^3, \quad \lambda^3_1 + \gamma^3 \geq 2(1-x_1), \\
& y_5 \geq y_3 + 1(1-x_3) + \lambda^4_3/2 + \gamma^4, \quad \lambda^4_3 + \gamma^4 \geq 1(1-x_3), \\
& y_6 \geq y_5 + 3(1-x_5) + \lambda^5_5/2 + \gamma^5, \quad \lambda^5_5 + \gamma^5 \geq 3(1-x_5), \\
& y_7 \geq y_6 + 1(1-x_6) + \lambda^6_6/2 + \gamma^6, \quad \lambda^6_6 + \gamma^6 \geq 1(1-x_6), \\
& x_i \leq 1/2 \;\; \forall\, i \in \{1,\ldots,6\}, \quad \sum_{i=1}^{6} x_i \leq 1.
\end{aligned}$$

The optimal allocation is $x^3 \approx (0.36, 0.5, 0, 0.14, 0, 0)^\top$*, and the optimal vector* y^3 *is given by* $y^3_1 = 0$*,* $y^3_3 \approx 1.94$*,* $y^3_5 \approx 5.68$*,* $y^3_6 \approx 10.18$ *and* $y^3_7 \approx 11.68$*. The estimated worst-case makespan is approximately* 11.68*.*

Proposition 5.5.2 indicates that we should not replace the arcs $(1, 3, \{1\}), (3, 5, \{3\}) \in E_3$*. We proceed by applying a forward replacement to the arc* $(1, 6, \{1, 2, 4\})$*. For the sake of brevity, however, we omit the remaining steps of our bounding approach.*

5.6 Numerical Results for Random Test Instances

We investigate the performance of our bounding technique and compare it with the decision rule approximations reviewed in Section 5.2.2. To this end, we use the RanGen algorithm described in [DVH03] to generate 100 random instances of problem $\mathcal{RTN}$ of size $n \in \{100, 200, 300\}$ and order strength 0.25, 0.5 and 0.75. The *order strength* of a network $G = (V, E)$ denotes the fraction of all $n(n-1)/2$ theoretically possible precedences between the nodes in V that are enforced through the arcs in E (either directly or via transitivity), see [DH02]. Table 5.1 summarises the numbers of inclusion-maximal paths for each instance class. Note that this number increases with the instance size and the order strength. We expect instances with a larger number of paths to be more challenging to solve with our bounding approach.

n	0.25	0.50	0.75
100	1,158	14,940	1,929,456
200	7,275	390,715	3,134,873,127
300	22,893	3,477,994	608,740,179,463

Table 5.1: Numbers of inclusion-maximal paths for the generated instance classes. Each class is described by its network size (row) and its order strength (column). Shown are the median values over 100 test instances.

We solve the resource allocation problem outlined in Example 5.3.2, that is, we assume a single resource and task durations

$$d_i(x; \xi) := d_i^0 \,(1 - x_i)\,(1 + \xi_i) \qquad \text{for } i \in V,$$

where d_i^0 denotes the nominal task duration, x_i the amount of the resource that is assigned to task i, and ξ_i the uncertainty inherent to the task duration. We sample d_i^0 uniformly from the interval $[1, 10]$ and set

$$\begin{aligned} X &:= \left\{x \in \mathbb{R}^n_+ \;:\; x \leq (1/2)\mathrm{e},\; \mathrm{e}^\top x \leq \beta\right\} \\ \text{and} \quad \Xi &:= \left\{\xi \in \mathbb{R}^n_+ \;:\; \xi \leq (1/2)\mathrm{e},\; \mathrm{e}^\top \xi \leq \gamma\right\}. \end{aligned}$$

Thus, the duration of task i can fall below or exceed its nominal duration d_i^0 by 50%, depending on the resource allocation and the realisation of the uncertain parameter vector ξ. We choose the resource budget β such that 10% of all tasks can be sped up to their minimal durations. Likewise, we select the uncertainty budget γ such that on average 10% of the tasks on each inclusion-maximal path can attain their worst-case durations.

Even though they constitute linear programs, the resulting instances of $\mathcal{ARTN}$ are difficult to solve with a standard optimiser. Indeed, for instances with 100 tasks and an order strength of 0.5, $\mathcal{ARTN}$ already contains more than 345,000 variables and 235,000 constraints on average. To bound the optimal value of $\mathcal{ARTN}$, we run the algorithms from Sections 5.4 and 5.5 in parallel for one hour. We solve all intermediate optimisation problems with IBM ILOG CPLEX 12.1 on a 2.53 GHz Intel Core 2 Duo computer. Figure 5.9 visualises the resulting optimality gaps as functions of the computation time. As expected, instances with a large number of tasks and a high order strength are more difficult to solve. Apart from instances with 300 tasks and an order strength of 0.75, however, the optimality gaps after one hour are all below 10%. Moreover, more than 90% of the instances of three classes (100 tasks with an order strength of 0.25; 100 tasks with an order strength of 0.5; 200 tasks with an order strength of 0.25) are solved within the time limit.

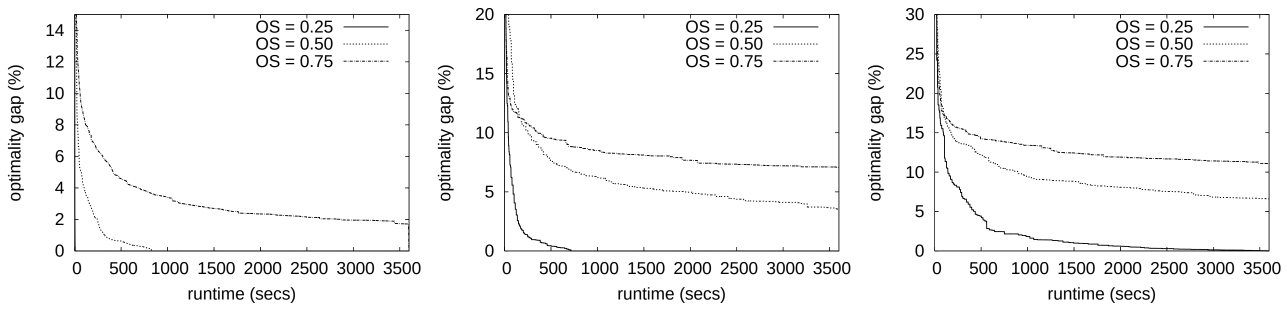


Figure 5.9: Median optimality gaps of our bounding approach as functions of the runtime. From left to right, the graphs show the results for instances of size 100, 200 and 300. For each instance class, we display the optimality gaps for three different order strengths (OS). In the first graph, the optimality gap for OS = 0.25 vanishes so quickly that the curve cannot be seen.

We now investigate the individual contributions of the upper and lower bounds to the optimality gaps in Figure 5.9. To this end, Figure 5.10 presents the upper and lower bounds as functions of the runtime for instances with an order strength of 0.5. For instances with 200 and 300 tasks, the lower bound improves rapidly in the beginning but fails to prove optimality within the time limit. Indeed, the graphs reveal that the upper and lower bounds improve throughout the computation, although the progress slows down after some time.

We now compare the results of our bounding approach with the decision rule approximations outlined in Section 5.2.2. We were unable to solve the affine decision rule approximations for

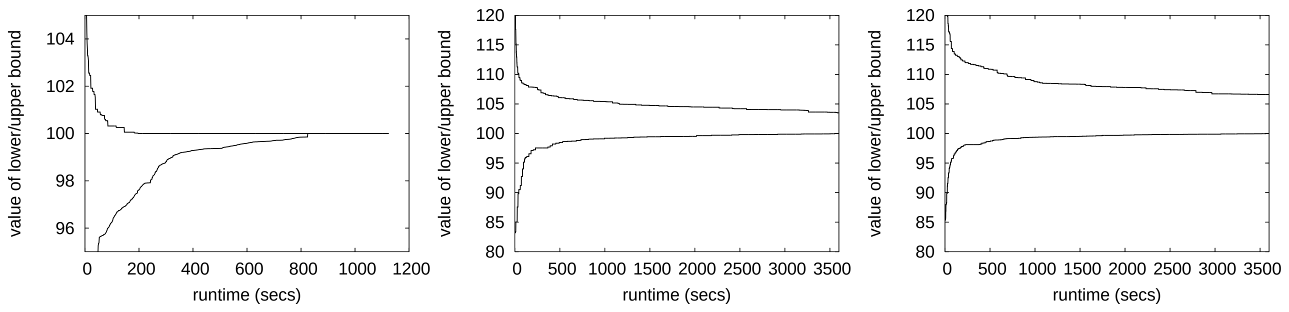


Figure 5.10: Median lower and upper bounds of our bounding approach as functions of the runtime. From left to right, the graphs visualise the results for instances of size 100, 200 and 300 and an order strength of 0.5. The objective values are normalised so that the lower bound after one hour evaluates to 100.

any of the test instances within the time limit of one hour. Indeed, the optimisation models for instances with 100 tasks and an order strength of 0.25 already contain more than 140,000 variables and 130,000 constraints on average. We therefore restrict each affine decision rule $y_j(\xi)$, $j \in V$, to depend on a small number of random variables ξ_i associated with the task durations d_i of predecessor tasks i of j. The results are presented in Table 5.2. As expected, affine decision rules perform better than constant decision rules, and the approximation quality of the affine decision rules improves with the number of considered random variables. However, the results are consistently dominated by our bounding approach. The results in Table 5.2 could be improved by using piecewise affine decision rules, but in this case the allowed computation time would have to be increased considerably.

	CDR			ADR-5			ADR-10		
n	0.25	0.50	0.75	0.25	0.50	0.75	0.25	0.50	0.75
100	32.16%	36.91%	36.77%	19.57%	26.47%	28.00%	15.85%	22.11%	23.17%
	0.62	0.98	1.28	1.92	8.14	11.32	6.98	39.11	42.82
200	30.10%	31.09%	33.30%	22.14%	22.35%	25.47%	18.74%	19.16%	22.10%
	4.36	7.16	9.47	18.63	270.45	443.74	91.8	1,100.53	1,562.63
300	26.65%	27.40%	30.95%	19.64%	21.65%	22.60%	17.56%	n/a	n/a
	12.53	23.69	37.75	181.71	2,062.24	2,612.82	717.77	n/a	n/a

Table 5.2: Computational results for constant decision rules (CDR) and affine decision rules over 5 (ADR-5) and 10 (ADR-10) random variables. For each approximation scheme, the results are grouped as in Table 5.1. The first value of each entry shows the median percentage by which the decision rule approximation exceeds the final upper bound of our approach, while the second value presents the median runtime of the decision rule approximation in seconds. Experiments in which less than 50% of the instances could be solved within one hour do not allow the calculation of median values and are therefore labelled ‘n/a’.

We close with an analysis of the impact of the resource budget β and the uncertainty budget γ on our bounding approach. As Table 5.3 shows, our bounding scheme works best if β is large and γ is small. An empirical inspection revealed that in this case, the intermediate resource allocations change less between consecutive iterations of the lower and upper bounds. We suspect that this 'allocation stability' allows our bounding algorithm to progress faster.

	budget β		budget γ	
nominal	20%	30%	20%	30%
5.85%	0.54%	0.00%	8.04%	10.21%
3600.0	3600.0	74.22	3600.0	3600.0

Table 5.3: Median optimality gaps and median runtimes of our bounding approach for different values of the resource budget β and the uncertainty budget γ. The nominal test set comprises instances of size 200 and an order strength of 0.5 in which β and γ are chosen as described in the beginning of the section. The remaining test sets increase one of the budgets by a factor of 2 (20%) or 3 (30%).

5.7 Case Study: VLSI Design

We now apply our bounding technique to a circuit sizing problem with process variations. For a survey of optimisation problems in circuit design, see [BKPH05].

An important problem in circuit design is to select the gate sizes in a circuit with the goal to optimally balance three conflicting objectives: operating speed, circuit size and power consumption. Loosely speaking, larger gate sizes increase the circuit size and power consumption, but they reduce the gate delays. We can model a circuit as a temporal network with gates as tasks and interconnections between gates as precedences. The duration of task $i \in V$ refers to the delay of gate i. The makespan of the network corresponds to the delay of the overall circuit, which in turn is inversely proportional to the circuit's operating speed. A resource allocation assigns sizes to all gates in the circuit.

The maximisation of circuit speed, subject to constraints on power consumption and circuit size, can be cast as a deterministic resource allocation problem that is defined on a temporal network. In practise, however, a circuit represents only one component of a larger system, and its eventual operating speed depends on adjacent circuits (that are outside the model). Hence, one commonly imposes a lower bound on the circuit speed and minimises the circuit size instead. For the sake of simplicity, we ignore power consumption here. The deterministic problem then becomes

$$\inf_{\substack{x \in [\underline{x}, \overline{x}], \\ y \in Y(x)}} \left\{ \sum_{i \in V} A_i x_i \,:\, y_n + d_n(x) \leq T \right\}, \tag{5.10a}$$

where

$$Y(x) = \left\{ y \in \mathbb{R}^n_+ \,:\, y_j \geq y_i + d_i(x) \; \forall \, (i,j) \in E \right\}. \tag{5.10b}$$

Here, x_i represents the size of gate i (with positive lower and upper bounds $\underline{x}_i$ and $\overline{x}_i$, respectively) and $A_i x_i$ the area occupied by gate i. Assuming that the circuit has a unique sink n (see Section 1.1), $y_n + d_n(x)$ denotes the delay of the overall circuit. We require that this quantity must not exceed some target value T. Note that for some values of T, the problem may be infeasible, which necessitates the use of the infimum operator instead of a minimum.

In the following, we employ a resistor-capacitor model for the gate delays:

$$d_i(x) = 0.69 \frac{R_i}{x_i} \Big(C_i^{\text{int}} x_i + \sum_{j:(i,j) \in E} C_j^{\text{in}} x_j \Big) \quad \text{for } i \in V,\, x \in X, \tag{5.11}$$

where R_i, C_i^{int} and C_i^{in} denote the driving resistance, intrinsic capacitance and input capacitance of gate i, respectively [BKPH05].

Variations in the manufacturing process entail that the factual gate sizes deviate from the selected target sizes x by some random, zero-mean noise $\xi \in \mathbb{R}^n$. If this noise is small compared to x, then we can express the resulting gate delays $d_i(x + \xi)$, $i \in V$, by a first-order Taylor approximation:

$$d_i(x; \xi) = d_i(x) + \big[\nabla d_i(x) \big]^\top \xi \quad \text{for } i \in V$$

Process variations exhibit non-negative correlations [SNLS05]. We can account for such correlations by using an ellipsoidal uncertainty set:

$$\Xi = \left\{ \xi \in \mathbb{R}^n \,:\, \exists \, u \in \mathbb{R}^l \,.\, \xi = \Sigma u,\, \|u\|_2 \leq 1 \right\} \quad \text{with } \Sigma \in \mathbb{R}^{n \times l}_+ \tag{5.12}$$

We thus seek to optimise the following variant of $\mathcal{RTN}$:

$$\inf_{x \in [\underline{x}, \overline{x}]} \sup_{\xi \in \Xi} \inf_{y \in Y(x,\xi)} \left\{ \sum_{i \in V} A_i x_i \; : \; y_n + d_n(x;\xi) \leq T \right\} \tag{5.13}$$

For a suitable ϕ (see Section 5.3), this results in the following variant of $\mathcal{ARTN}$:

$$\inf_{x \in [\underline{x}, \overline{x}]} \left\{ \sum_{i \in V} A_i x_i \; : \; \phi(x;P) \leq T \;\; \forall \, P \in \mathcal{P} \right\} \tag{5.14}$$

Again, problem (5.14) may be infeasible if T is chosen too small. An inspection of Sections 5.4 and 5.5 reveals that we can apply our bounding approach to problem (5.14) if we allow the bounds to attain values on the extended real line $\mathbb{R} \cup \{\infty\}$. A lower bound of ∞ signalises that problem (5.14) is infeasible, while an upper bound of ∞ indicates that the determined gate sizes x may violate the target value T for the overall circuit delay. The following result provides us with a conservative reformulation of (5.13):

Proposition 5.7.1 *For d and Ξ defined in (5.11)–(5.12), let*

$$\phi(x;P) = \mathbb{I}_P^\top d(x) + \left\| \Sigma^\top \Big(\sum_{i \in P} \left[\nabla d_i(x) \right]^+ \Big) \right\|_2 + \left\| \Sigma^\top \Big(\sum_{i \in P} \left[\nabla d_i(x) \right]^- \Big) \right\|_2, \tag{5.15}$$

where

$$\left[f(x) \right]^+ = \sum_{i : \alpha_i > 0} \alpha_i \prod_j (x_j)^{\beta_{ij}} \quad \text{for} \quad f(x) = \sum_i \alpha_i \prod_j (x_j)^{\beta_{ij}}$$

and $\left[f(x) \right]^-$ defined analogously for i with $\alpha_i < 0$. If X has a tractable representation, then (5.14)–(5.15) constitutes a conservative reformulation of (5.13) that satisfies (A1)–(A3).

Proof It follows from [SNLS05] that ϕ as defined in (5.15) satisfies condition (5.5) on page 123 and (A3). It remains to be shown that ϕ satisfies (A1) and (A2). For $x \in X$ and $P \subseteq V$, we introduce the following notation:

$$\varphi^+(x,P) = \left\| \Sigma^\top \Big(\sum_{i \in P} \left[\nabla d_i(x) \right]^+ \Big) \right\|_2 \text{ and } \varphi^-(x,P) = \left\| \Sigma^\top \Big(\sum_{i \in P} \left[\nabla d_i(x) \right]^- \Big) \right\|_2.$$

As for (A1), we need to show that

$$\begin{aligned}\phi(x;P') \;&=\; \mathbb{I}_{P'}^\top d(x) + \varphi^+(x,P') + \varphi^-(x;P') \\ &\geq\; \mathbb{I}_P^\top d(x) + \varphi^+(x;P) + \varphi^-(x;P) \;=\; \phi(x;P)\end{aligned}$$

for all $x \in X$ and $P \subset P' \subseteq V$. Note that $\mathbb{I}_{P'}^\top d(x) \geq \mathbb{I}_P^\top d(x)$ since $\mathbb{I}_{P'} \geq \mathbb{I}_P$ and $d(x) \geq 0$ for all $x \in X$. We show that $\varphi^+(x;P') \geq \varphi^+(x;P)$ and $\varphi^-(x;P') \geq \varphi^-(x;P)$. The first inequality follows from the fact that Σ is element-wise non-negative and $\left[\nabla d_i(x)\right]^+ \geq 0$ for all $i \in V$. The second inequality follows from the positive homogeneity of norms and the fact that $\left[\nabla d_i(x)\right]^- \leq 0$ for all $i \in V$.

(A2) is satisfied if

$$\begin{aligned}\phi(x;P) + \phi(x;P' \setminus P) \;&= \mathbb{I}_P^\top d(x) + \varphi^+(x;P) + \varphi^-(x;P) + \\ &\qquad \mathbb{I}_{[P'\setminus P]}^\top d(x) + \varphi^+(x;P' \setminus P) + \varphi^-(x;P' \setminus P) \\ &\geq \mathbb{I}_{P'}^\top d(x) + \varphi^+(x,P') + \varphi^-(x;P') \;=\; \phi(x;P')\end{aligned}$$

for all $x \in X$ and $P \subset P' \subseteq V$. Note that $\mathbb{I}_P^\top d(x) + \mathbb{I}_{[P'\setminus P]}^\top d(x) = \mathbb{I}_{P'}^\top d(x)$. Also, we have

$$\varphi^+(x;P) + \varphi^-(x;P) + \varphi^+(x;P' \setminus P) + \varphi^-(x;P' \setminus P) \geq \varphi^+(x,P') + \varphi^-(x;P')$$

by the triangle inequality. ∎

We use Proposition 5.7.1 to determine robust gate sizes for the ISCAS 85 benchmark circuits.[2] To this end, we set $(\underline{x}_i, \overline{x}_i) = (1, 16)$ and select the circuit parameters A_i, R_i, C_i^{int} and C_i^{in} according to the Logical Effort model [BKPH05, SSH99]. We set the target delay T to 130% of the minimal circuit delay in absence of process variations. For ease of exposition, we assume independent process variations, that is, Σ is a diagonal matrix. We set the diagonal elements of Σ to 25% of the gate sizes determined by the deterministic model (5.10).

The data in Table 5.4 specifies the temporal networks corresponding to the ISCAS 85 benchmark

[2]ISCAS 85 benchmark circuits: `http://www.cbl.ncsu.edu/benchmarks`.

circuit	# tasks	# precedences	# task paths
C432	196	336	83,926
C499	243	408	9,440
C880	443	729	8,642
C1355	587	1,064	4,173,216
C1908	913	1,498	729,056
C2670	1,426	2,076	679,954
C3540	1,719	2,939	28,265,874
C5315	2,485	4,386	1,341,305
C6288	2,448	4,800	1,101,055,638
C7552	3,719	6,144	726,494

Table 5.4: ISCAS 85 benchmark circuits.

circuits. For a circuit with $|V|$ tasks and $|\overline{\mathcal{P}}|$ inclusion-maximal task paths, the path-wise model (5.14) can be reformulated as a geometric program with $1 + |V| + 2\,|\overline{\mathcal{P}}|$ variables and $3\,|\overline{\mathcal{P}}|$ constraints, see [BKPH05, SNLS05]. Due to the choice of ϕ in (5.15), the Jacobian of the constraints is dense. In view of the cardinality of $\overline{\mathcal{P}}$ in the benchmark circuits (see Table 5.4), a direct solution of (5.14) is prohibitive.

We now use our bounding approach to solve problem (5.14) for the benchmark circuits. We terminate our algorithm after 50 iterations of the lower and upper bound procedures. Since the lower bound requires the investigation of a potentially large number of task paths (see Step 3(b) of Algorithm 5.2), we limit its computation time per iteration to the time required by the upper bound. All results are generated with CONOPT 3 on an Intel Xeon architecture with 2.83GHz.[3] We employ warm starts for the calculation of both lower and upper bounds, which significantly reduces the computational effort.

Table 5.5 presents the optimality gaps after 1, 25 and 50 iterations. It also documents the reduction in overall circuit size when we use our bounding approach (for 50 iterations) instead of a model with constant decision rules (see Section 5.2.2). We remark that the choice of Ξ and ϕ in (5.12) and (5.15) implies that constant and affine decision rules result in the same solutions. Although the initial optimality gaps can be large, our bounding approach reduces them to reasonable values after a few iterations. Moreover, the computational effort remains modest for all considered problem instances. Finally, we see that our bounding approach can

[3]CONOPT homepage: `http://www.conopt.com`.

lead to drastic reductions in overall circuit size.

circuit	first it.	after 25 its.	after 50 its.	reduction
C432	34.13%	*solved after 11 its.*		*24.48%*
	0:03	1:03		
C499	148.82%	12.31%	8.96%	*42.89%*
	0:12	27:35	128:30	
C880	16.78%	2.31%	0.70%	*11.16%*
	0:11	2:44	8:39	
C1355	113.16%	*solved after 24 its.*		*52.95%*
	0:17	17:31		
C1908	37.05%	11.37%	6.90%	*18.13%*
	1:17	6:58	21:06	
C2670	14.62%	1.61%	1.02%	*11.09%*
	0:51	24:03	99:35	
C3540	37.66%	9.19%	7.40%	*20.50%*
	4:22	16:31	56:06	
C5315	15.23%	4.30%	2.29%	*10.33%*
	6:56	30:39	52:37	
C6288	68.24%	3.40%	2.52%	*39.07%*
	6:33	45:09	69:08	
C7552	11.03%	*solved after 12 its.*		*5.01%*
	5:54	15:08		

Table 5.5: Results for the circuits from Table 5.4. Columns 2, 3 and 4 present the optimality gaps and computation times (mins:secs) after 1, 25 and 50 iterations of our bounding approach, respectively. The last column quantifies the reduction in overall circuit size if we employ our bounding approach instead of a model with constant decision rules (see Section 5.2.2).

5.8 Conclusion

This chapter studied robust resource allocations in temporal networks. Our problem formulation assumes that the task durations are uncertain and that resource allocations are evaluated in view of their worst-case makespan. We showed that the resulting optimisation problem is $\mathcal{NP}$-hard. We developed convergent bounds on its optimal objective value, as well as feasible resource allocations whose objective values are bracketed by these bounds.

It would be interesting to extend our solution procedure to renewable and doubly-constrained resources. Indeed, Section 2.1 lists some application domains (*e.g.*, scheduling of production processes and microprocessors) that impose additional restrictions on the consumption rate of resources. Such constraints result in nonconvex problems that render our bounding approach computationally prohibitive. Instead, one could design a branch-and-bound algorithm that branches upon violations of the additional constraints. For every node in the resulting branch-and-bound tree, the incurred worst-case makespan can be bounded with our method.

Chapter 6

Multi-Stage Net Present Value Maximisation

In addition to the notation introduced in Section 1.3, this chapter uses the following notation. For a finite set $\mathcal{X} = \{1, \ldots, X\}$, $\mathcal{M}(\mathcal{X})$ denotes the probability simplex in $\mathbb{R}^X$. An $\mathcal{X}$-valued random variable χ has distribution $m \in \mathcal{M}(\mathcal{X})$, denoted by $\chi \sim m$, if $\mathbb{P}(\chi = x) = m_x$ for all $x \in \mathcal{X}$. For square matrices A and B, the relation $A \succeq B$ indicates that the matrix $A - B$ is positive semidefinite. We denote the space of symmetric $n \times n$ matrices by $\mathbb{S}^n$. The declaration $f : X \overset{\text{c}}{\mapsto} Y$ ($f : X \overset{\text{a}}{\mapsto} Y$) implies that f is a continuous (affine) function from X to Y. For a matrix A, we denote its ith row by $A_{i\cdot}^\top$ (a row vector) and its jth column by $A_{\cdot j}$.

6.1 Introduction

Markov decision processes (MDPs) provide a versatile model for sequential decision-making under uncertainty, which accounts for both the immediate effects and future ramifications of decisions. In the past sixty years, MDPs have been successfully applied to numerous areas, ranging from inventory control and investment planning to studies in economics and behavioural ecology [Ber07, Put94]. Our interest in MDPs arises from the fact that multi-stage NPV maximisation problems in temporal networks can be modelled as MDPs. We will discuss this link between MDPs and temporal networks in Section 6.7.

In this chapter, we study MDPs with a finite state space $\mathcal{S} = \{1, \ldots, S\}$, a finite action space $\mathcal{A} = \{1, \ldots, A\}$, and a discrete but infinite planning horizon $\mathcal{T} = \{0, 1, 2, \ldots\}$. We briefly review the definitions introduced in Section 2.2.3. We assume that every action is admissible in every state. The initial state is random and follows the probability distribution $p_0 \in \mathcal{M}(\mathcal{S})$. If action $a \in \mathcal{A}$ is chosen in state $s \in \mathcal{S}$, the subsequent state is determined by the conditional probability distribution $p(\cdot|s, a) \in \mathcal{M}(\mathcal{S})$. We condense these conditional distributions to the transition kernel $P \in [\mathcal{M}(\mathcal{S})]^{S \times A}$, where $P_{sa} := p(\cdot|s, a)$ for $(s, a) \in \mathcal{S} \times \mathcal{A}$. The decision maker receives an expected reward of $r(s, a, s') \in \mathbb{R}_+$ if action $a \in \mathcal{A}$ is chosen in state $s \in \mathcal{S}$ and the subsequent state is $s' \in \mathcal{S}$. Without loss of generality, we assume that all rewards are non-negative. The MDP is controlled through a policy $\pi = (\pi_t)_{t \in \mathcal{T}}$, where $\pi_t : (\mathcal{S} \times \mathcal{A})^{t-1} \times \mathcal{S} \mapsto \mathcal{M}(\mathcal{A})$. $\pi_t(\cdot|s_0, a_0, \ldots, s_{t-1}, a_{t-1}; s_t)$ represents the probability distribution over $\mathcal{A}$ according to which the next action is chosen if the current state is s_t and the state-action history is given by $(s_0, a_0, \ldots, s_{t-1}, a_{t-1})$. Together with the transition kernel P, π induces a stochastic process $(s_t, a_t)_{t \in \mathcal{T}}$ on the space $(\mathcal{S} \times \mathcal{A})^\infty$ of sample paths. We use the notation $\mathbb{E}^{P,\pi}$ to denote expectations with respect to this process. Throughout this chapter, we evaluate policies in view of their expected total reward under the discount factor $\lambda \in (0, 1)$:

$$\mathbb{E}^{P,\pi} \left[\sum_{t=0}^{\infty} \lambda^t r(s_t, a_t, s_{t+1}) \;\middle|\; s_0 \sim p_0 \right] \tag{6.1}$$

For a fixed policy π, the *policy evaluation problem* asks for the value of expression (6.1). The *policy improvement problem*, on the other hand, asks for a policy π that maximises (6.1).

Most of the literature on MDPs assumes that the expected rewards r and the transition kernel P are known, with a tacit understanding that they have to be estimated in practise. However, it is well-known that the expected total reward (6.1) can be very sensitive to small changes in r and P [MSST07]. Thus, decision makers are confronted with two different sources of uncertainty. On one hand, they face *internal variation* due to the stochastic nature of MDPs. On the other hand, they need to cope with *external variation* because the estimates for r and P deviate from their true values. We assume that the decision maker is risk-neutral to internal variation but risk-averse to external variation. This is justified if the MDP runs for a long time, or if many instances of the same MDP run in parallel [MSST07]. We focus on external variation in P and assume r to be known. Indeed, the expected total reward (6.1) is typically more sensitive to P, and the inclusion of reward variation is straightforward [DM10, MSST07].

Let P^0 be the unknown true transition kernel of the MDP. Since the expected total reward of a policy depends on P^0, we cannot evaluate expression (6.1) under external variation. It is suggested in [Iye05, NG05] to find a policy that guarantees the highest expected total reward at a given confidence level. To this end, a policy π is determined that maximises the worst-case expected total reward

$$z^* = \inf_{P \in \mathcal{P}} \mathbb{E}^{P,\pi} \left[\sum_{t=0}^{\infty} \lambda^t r(s_t, a_t, s_{t+1}) \,\middle|\, s_0 \sim p_0 \right], \tag{6.2}$$

where the uncertainty set $\mathcal{P}$ is the Cartesian product of independent marginal sets $\mathcal{P}_{sa} \subseteq \mathcal{M}(\mathcal{S})$ for each $(s, a) \in \mathcal{S} \times \mathcal{A}$. In the following, we call such uncertainty sets *rectangular*. Problem (6.2) determines the worst-case expected total reward of π if the transition kernel can vary freely within $\mathcal{P}$. In analogy to our earlier definitions, the *robust policy evaluation problem* evaluates expression (6.2) for a fixed policy π, while the *robust policy improvement problem* asks for a policy that maximises (6.2). The optimal value z^* in (6.2) provides a lower bound on the expected total reward of π if the true transition kernel P^0 is contained in the uncertainty set $\mathcal{P}$. Hence, if $\mathcal{P}$ is a confidence region that contains P^0 with probability $1 - \beta$, then the policy π guarantees an expected total reward of at least z^* at a confidence level $1 - \beta$. To construct an uncertainty set $\mathcal{P}$ with this property, [Iye05] and [NG05] assume that independent transition samples are available for each state-action pair $(s, a) \in \mathcal{S} \times \mathcal{A}$. Under this assumption, the samples for each state-action pair follow independent multinomial distributions whose (unknown) parameters coincide with the entries of P^0. One can then employ standard statistical techniques to derive a confidence region for P^0. If we project this confidence region onto the marginal sets $\mathcal{P}_{sa}$, then z^* provides the desired probabilistic lower bound on the expected total reward of π.

In this chapter, we alter two key assumptions of the outlined procedure. Firstly, we assume that the decision maker cannot obtain independent transition samples for the state-action pairs. Instead, she has merely access to an observation history $(s_1, a_1, \ldots, s_n, a_n) \in (\mathcal{S} \times \mathcal{A})^n$ generated by the MDP under some known policy. Secondly, we relax the assumption of rectangular uncertainty sets. In the following, we briefly motivate these changes and give an outlook on their consequences.

Although transition sampling has theoretical appeal, it is often prohibitively costly or even infeasible in practise. To obtain independent samples for each state-action pair, one needs to repeatedly direct the MDP into any of its states and record the transitions resulting from different actions. In particular, one cannot use the transition frequencies of an observation history because those frequencies violate the independence assumption stated above. The availability of an observation history, on the other hand, seems much more realistic in practise. Observation histories introduce a number of theoretical challenges, such as the lack of observations for some transitions and stochastic dependencies between the transition frequencies. We will apply results from statistical inference on Markov chains to address these issues.

The restriction to rectangular uncertainty sets has been introduced in [Iye05] and [NG05] to facilitate computational tractability. Under the assumption of rectangularity, the robust policy evaluation and improvement problems can be solved efficiently with a modified value or policy iteration. This implies, however, that non-rectangular uncertainty sets have to be projected onto the marginal sets $\mathcal{P}_{sa}$. Not only does this 'rectangularisation' unduly increase the level of conservatism, but it also creates a number of undesirable side-effects that we discuss in Section 6.2. In this chapter, we show that the robust policy evaluation and improvement problems remain tractable for uncertainty sets that exhibit a milder form of rectangularity, and we develop a polynomial time solution method. On the other hand, we prove that the robust policy evaluation and improvement problems are intractable for non-rectangular uncertainty sets. For this setting, we formulate conservative approximations of the policy evaluation and improvement problems. We bound the optimality gap incurred from solving those approximations, and we outline how our approach can be generalised to a hierarchy of increasingly accurate approximations.

The contributions of this chapter can be summarised as follows.

1. We analyse a new class of uncertainty sets, which contains the above defined rectangular uncertainty sets as a special case. We show that the optimal policies for this class are randomised but memoryless. We develop algorithms that solve the robust policy evaluation and improvement problems over these uncertainty sets in polynomial time.

2. It is stated in [NG05] that the robust policy evaluation and improvement problems "seem to be hard to solve" for non-rectangular uncertainty sets. We prove that both problems are indeed strongly $\mathcal{NP}$-hard. We develop a hierarchy of increasingly accurate conservative approximations, together with bounds on the incurred optimality gap.

3. We present a method to construct uncertainty sets from observation histories. In contrast, existing approaches rely on transition sampling, which is often too costly or infeasible in practise. Our approach allows to account for different types of a priori information about the transition kernel, which helps to reduce the size of the uncertainty set. We also investigate the convergence behaviour of our uncertainty set when the length of the observation history increases.

The study of robust MDPs with rectangular uncertainty sets dates back to the seventies, see [BNS01, GLD00, SL73, WE94] and the surveys in [Iye05, NG05]. However, most of the early contributions do not address the construction of suitable uncertainty sets. In [MSST07], the authors approximate the bias and variance of the expected total reward (6.1) if the unknown model parameters are replaced with estimates. These approximations are used in [DM10] to solve a chance-constrained policy improvement problem in a Bayesian setting. Recently, alternative performance criteria have been suggested to address external variation, such as the worst-case expected utility and regret measures. We refer to [PK08, XM06] and the references cited therein. Note that we could address external variation by encoding the unknown model parameters into the states of a partially observable MDP (POMDP) [Mon82]. However, the optimisation of POMDPs becomes challenging even for small state spaces. In our case, the augmented state space would become very large, which renders optimisation of the resulting POMDPs prohibitively expensive.

The remainder of this chapter is organised as follows. Section 6.2 defines and analyses the classes of robust MDPs that we consider. Sections 6.3 and 6.4 study the robust policy evaluation and improvement problems, respectively. Section 6.5 constructs uncertainty sets from observation histories. We illustrate our method in Section 6.6, where we apply it to the machine replacement problem. Section 6.7 establishes the link between MDPs and temporal networks. We conclude in Section 6.8.

Remark 6.1.1 (Finite Horizon MDPs) *Throughout the chapter, we outline how our results extend to finite horizon MDPs. In this case, we assume that $\mathcal{T} = \{0, 1, 2, \ldots, T\}$ with $T < \infty$ and that $\mathcal{S}$ can be partitioned into nonempty disjoint sets $\{\mathcal{S}_t\}_{t \in \mathcal{T}}$ such that at period t the system is in one of the states in $\mathcal{S}_t$. We do not discount rewards in finite horizon MDPs. If the MDP reaches a terminal state $s \in \mathcal{S}_T$, an expected reward of $\mathfrak{r}_s \in \mathbb{R}_+$ is received. We assume that $p_0(s) = 0$ for $s \notin \mathcal{S}_1$.*

6.2 Robust Markov Decision Processes

This section studies properties of the robust policy evaluation and improvement problems. Both problems are concerned with *robust MDPs*, for which the transition kernel is only known to be an element of an uncertainty set $\mathcal{P} \subseteq [\mathcal{M}(\mathcal{S})]^{S \times A}$. Without loss of generality, we assume that the initial state distribution p_0 is known.

We start with the robust policy evaluation problem. We define the structure of the uncertainty sets that we consider, as well as different types of rectangularity that can be imposed to facilitate computational tractability. Afterwards, we discuss the robust policy improvement problem. We define several policy classes that are commonly used in MDPs, and we investigate the structure of optimal policies for different types of rectangularity. We close with a complexity result for the robust policy evaluation problem. Since the remainder of this chapter almost exclusively deals with the robust versions of the policy evaluation and improvement problems, we may suppress the attribute 'robust' in the following.

6.2.1 The Robust Policy Evaluation Problem

Consider the policy evaluation problem (6.2), where we replace the uncertainty set $\mathcal{P}$ with

$$\mathcal{P} := \left\{ P \in [\mathcal{M}(\mathcal{S})]^{S \times A} \, : \, \exists \, \xi \in \Xi \text{ such that } P_{sa} = p^{\xi}(\cdot|s,a) \;\; \forall \, (s,a) \in \mathcal{S} \times \mathcal{A} \right\}. \tag{6.3a}$$

Here, we assume that Ξ is a subset of $\mathbb{R}^q$ and that $p^{\xi}(\cdot|s,a)$, $(s,a) \in \mathcal{S} \times \mathcal{A}$, is an affine function from Ξ to $\mathcal{M}(\mathcal{S})$ that satisfies $p^{\xi}(\cdot|s,a) := k_{sa} + K_{sa}\xi$ for some $k_{sa} \in \mathbb{R}^S$ and $K_{sa} \in \mathbb{R}^{S \times q}$. We also stipulate that

$$\Xi := \left\{ \xi \in \mathbb{R}^q \, : \, \xi^\top O_l \, \xi + o_l^\top \xi + \omega \geq 0 \;\; \forall \, l = 1, \ldots, L \right\}, \tag{6.3b}$$

where $O_l \in \mathbb{S}^q$ satisfies $O_l \preceq 0$. We assume that Ξ is bounded and that it contains a Slater point $\overline{\xi} \in \mathbb{R}^q$ which satisfies $\overline{\xi}^\top O_l \, \overline{\xi} + o_l^\top \overline{\xi} + \omega > 0$ for all l. Our definition of Ξ encompasses all compact subsets of $\mathbb{R}^q$ that have a nonempty interior and that result from finite intersections of closed halfspaces and ellipsoids.

Example 6.2.1 *Consider a robust infinite horizon MDP with three states and one action. The transition probabilities are defined through*

$$p^{\xi}(1|s,1) = \frac{1}{3} + \frac{\xi_1}{3}, \quad p^{\xi}(2|s,1) = \frac{1}{3} + \frac{\xi_2}{3} \quad \textit{and} \quad p^{\xi}(3|s,1) = \frac{1}{3} - \frac{\xi_1}{3} - \frac{\xi_2}{3} \quad \textit{for } s \in \{1,2,3\},$$

where $\xi = (\xi_1, \xi_2)$ is only known to satisfy $\xi_1^2 + \xi_2^2 \leq 1$ and $\xi_1 \leq \xi_2$. We can model this MDP through

$$\Xi = \left\{ \xi \in \mathbb{R}^2 \, : \, \xi_1^2 + \xi_2^2 \leq 1, \, \xi_1 \leq \xi_2 \right\}, \quad k_{s1} = \frac{1}{3}\mathrm{e} \quad \textit{and} \quad K_{s1} = \frac{1}{3} \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -1 & -1 \end{pmatrix} \quad \textit{for } s \in \{1,2,3\}.$$

Note that the mapping K cannot be absorbed in the definition of Ξ without violating the Slater condition.

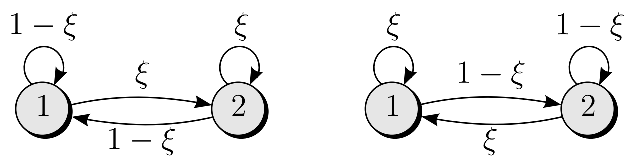


Figure 6.1: MDP with two states and two actions. The left and right charts present the transition probabilities for actions 1 and 2, respectively. In both diagrams, nodes correspond to states and arcs to transitions. We label each arc with the probability of the associated transition. We suppress p_0 and the expected rewards.

We say that an uncertainty set $\mathcal{P}$ is (s,a)*-rectangular* if

$$\mathcal{P} = \bigtimes_{(s,a)\in\mathcal{S}\times\mathcal{A}} \mathcal{P}_{sa}, \qquad \text{where} \qquad \mathcal{P}_{sa} := \{P_{sa} \,:\, P \in \mathcal{P}\} \text{ for } (s,a) \in \mathcal{S}\times\mathcal{A}.$$

Likewise, we say that an uncertainty set $\mathcal{P}$ is s*-rectangular* if

$$\mathcal{P} = \bigtimes_{s\in\mathcal{S}} \mathcal{P}_s, \qquad \text{where} \qquad \mathcal{P}_s := \{(P_{s1},\ldots,P_{sA}) \,:\, P \in \mathcal{P}\} \text{ for } s \in \mathcal{S}.$$

For any uncertainty set $\mathcal{P}$, we call $\mathcal{P}_{sa}$ and $\mathcal{P}_s$ the *marginal uncertainty sets* (or simply marginals). For our definition (6.3) of $\mathcal{P}$, we have $\mathcal{P}_{sa} = \{p^\xi(\cdot|s,a) \,:\, \xi \in \Xi\}$ and $\mathcal{P}_s = \{(p^\xi(\cdot|s,1),\ldots,p^\xi(\cdot|s,A)) \,:\, \xi \in \Xi\}$, respectively. Note that all transition probabilities $p^\xi(\cdot|s,a)$ can vary freely within their marginals $\mathcal{P}_{sa}$ if the uncertainty set is (s,a)-rectangular. In contrast, the transition probabilities $\{p^\xi(\cdot|s,a) \,:\, a \in \mathcal{A}\}$ for different actions in the same state may be dependent in an s-rectangular uncertainty set. By definition, (s,a)-rectangularity implies s-rectangularity. (s,a)-rectangular uncertainty sets have been introduced in [Iye05, NG05], whereas the notion of s-rectangularity seems to be new. Note that our definition (6.3) of $\mathcal{P}$ does not impose any kind of rectangularity. Indeed, the uncertainty set in Example 6.2.1 is not s-rectangular. The following example shows that rectangular uncertainty sets can result in crude approximations of the decision maker's knowledge about the true transition kernel P^0.

Example 6.2.2 (Rectangularity) *Consider the robust infinite horizon MDP that is shown in Figure 6.1. The uncertainty set $\mathcal{P}$ encompasses all transition kernels that correspond to parameter realisations $\xi \in [0,1]$. This MDP can be assigned an uncertainty set of the form (6.3). Figure 6.2 visualises $\mathcal{P}$ and the smallest s-rectangular and (s,a)-rectangular uncertainty sets*

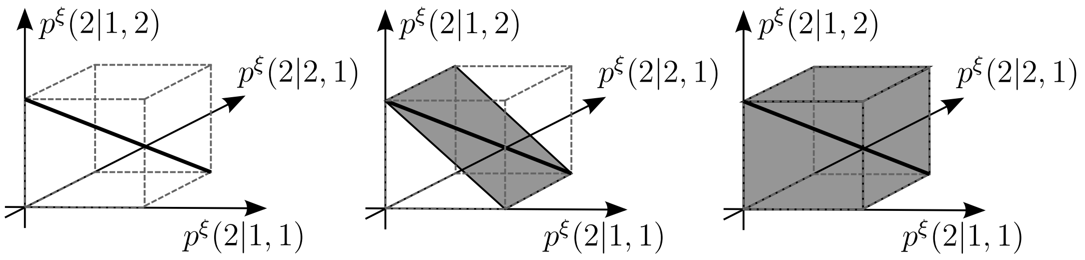


Figure 6.2: Illustration of $\mathcal{P}$ (left chart) and the smallest s-rectangular (middle chart) and (s, a)-rectangular (right chart) uncertainty sets that contain $\mathcal{P}$. The charts show three-dimensional projections of $\mathcal{P} \subset \mathbb{R}^8$. The thick line represents $\mathcal{P}$, while the shaded areas visualise the corresponding rectangular uncertainty sets. Figure 6.1 implies that $p^{\xi}(2|1,1) = \xi$, $p^{\xi}(2|1,2) = 1 - \xi$ and $p^{\xi}(2|2,1) = \xi$. The dashed lines correspond to the unit cube in $\mathbb{R}^3$.

that contain $\mathcal{P}$.

From now on, we always consider uncertainty sets of the form (6.3). We may sometimes call a generic uncertainty set *non-rectangular* to emphasise that it is neither s- nor (s, a)-rectangular.

6.2.2 The Robust Policy Improvement Problem

We now consider the policy improvement problem, which asks for a policy that maximises the worst-case expected total reward (6.2) over an uncertainty set of the form (6.3). Remember that a policy π represents a sequence of functions $(\pi_t)_{t \in \mathcal{T}}$ that map state-action histories to probability distributions over $\mathcal{A}$. In its most general form, such a policy is *history dependent*, that is, at any time period t the policy may assign a different probability distribution to each state-action history $(s_1, a_1, \ldots, s_{t-1}, a_{t-1}; s_t)$.

Due to the storage requirements of history dependent policies, one typically prefers more 'economical' policy classes. A policy π is called *Markovian* if π_t is determined by s_t and t for all $t \in \mathcal{T}$. A Markovian policy π is called *stationary* if π_t is solely determined by s_t for all $t \in \mathcal{T}$. In finite horizon MDPs, Markovian and stationary policies are equally expressive since the sets $\mathcal{S}_t$ are disjoint. In infinite horizon MDPs, however, stationary policies form a strict subset of the class of Markovian policies. A policy π is called *deterministic* if π_t places all probability mass on one action for each $t \in \mathcal{T}$; otherwise, π is called *randomised.* In the following, we will focus on stationary policies due to their favourable storage requirements. We denote by Π the set of all randomised stationary policies for a given MDP instance.

It is well-known that non-robust finite and infinite horizon MDPs always allow for a deterministic stationary policy that maximises the expected total reward (6.1). Optimal policies can be determined via value or policy iteration, or via linear programming. Finding an optimal policy, as well as evaluating (6.1) for a given stationary policy, can be done in polynomial time. For a detailed discussion, see [Ber07, Put94, Tsi07].

To date, the literature on robust MDPs has focused on (s, a)-rectangular uncertainty sets. For this class of uncertainty sets, it is shown in [Iye05, NG05] that the worst-case expected total reward (6.2) is maximised by a deterministic stationary policy π for finite and infinite horizon MDPs. Optimal policies can be determined via extensions of the value and policy iteration. For some uncertainty sets, finding an optimal policy, as well as evaluating (6.2) for a given stationary policy, can be achieved in polynomial time. Moreover, the policy improvement problem satisfies the following saddle point condition:

$$\sup_{\pi \in \Pi} \inf_{P \in \mathcal{P}} \mathbb{E}^{P,\pi} \left[\sum_{t=0}^{\infty} \lambda^t r(s_t, a_t, s_{t+1}) \;\middle|\; s_0 \sim p_0 \right] \;=\; \inf_{P \in \mathcal{P}} \sup_{\pi \in \Pi} \mathbb{E}^{P,\pi} \left[\sum_{t=0}^{\infty} \lambda^t r(s_t, a_t, s_{t+1}) \;\middle|\; s_0 \sim p_0 \right] \tag{6.4}$$

A similar result for robust finite horizon MDPs is discussed in [NG05].

We now show that the benign structure of optimal policies over (s, a)-rectangular uncertainty sets partially extends to the broader class of s-rectangular uncertainty sets.

Proposition 6.2.1 (s-Rectangular Uncertainty Sets) *Consider the robust policy improvement problem for a finite or infinite horizon MDP over an s-rectangular uncertainty set of the form (6.3).*

(a) There is always an optimal policy that is stationary.

(b) It is possible that all optimal stationary policies are randomised.

Proof As for claim (a), consider a finite horizon MDP with an s-rectangular uncertainty set. By construction, the probabilities associated with transitions emanating from state $s \in \mathcal{S}$ are independent from those emanating from any other state $s' \in \mathcal{S}$, $s' \neq s$. Moreover, each state s is visited at most once since the sets $\mathcal{S}_t$ are disjoint. Hence, any knowledge about past transition probabilities cannot contribute to better decisions in future time periods, which implies that stationary policies are optimal.

Consider now an infinite horizon MDP with an s-rectangular uncertainty set. Appendix B shows that the saddle point condition (6.4) extends to s-rectangular uncertainty sets. For any fixed transition kernel $P \in \mathcal{P}$, the supremum over all stationary policies on the right-hand side of (6.4) is equivalent to the supremum over all history dependent policies. By weak duality, the right-hand side of (6.4) thus represents an upper bound on the worst-case expected total reward of any history dependent policy. Since there is a stationary policy whose worst-case expected total reward on the left-hand side of (6.4) attains this upper bound, claim (a) follows.

As for claim (b), consider the robust infinite horizon MDP that is visualised in Figure 6.3. The uncertainty set $\mathcal{P}$ encompasses all transition kernels that correspond to parameter realisations $\xi \in [0, 1]$. This MDP can be assigned an s-rectangular uncertainty set of the form (6.3). Since the transitions are independent of the chosen actions from time 1 onwards, a policy is completely determined by the decision $\beta = \pi_0(1|1)$ at time 0. The worst-case expected total reward is

$$\min_{\xi \in [0,1]} \left[\beta\xi + (1-\beta)(1-\xi)\right] \frac{\lambda}{1-\lambda} = \min\left\{\beta, 1-\beta\right\} \frac{\lambda}{1-\lambda}.$$

Over $\beta \in [0, 1]$, this expression has its unique maximum at $\beta^* = 1/2$, that is, the optimal policy is randomised. If we replace the self-loops with expected terminal rewards of $\mathbf{r}_2 := 1$ and $\mathbf{r}_3 := 0$, then we obtain an example of a robust *finite* horizon MDP whose optimal policy is randomised. ∎

Figure 6.3 illustrates the counterintuitive result that randomisation is superfluous for (s, a)-rectangular uncertainty sets. If we project the uncertainty set $\mathcal{P}$ associated with Figure 6.3 onto its marginals $\mathcal{P}_{sa}$, then the transition probabilities in the left chart become independent

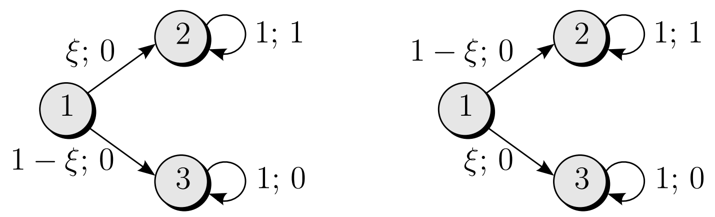


Figure 6.3: MDP with three states and two actions. The left and right figures present the transition probabilities and expected rewards for actions 1 and 2, respectively. The first and second expressions in the arc labels correspond to the probabilities and expected rewards of the associated transitions, respectively. Apart from that, the same drawing conventions as in Figure 6.1 are used. The initial state distribution p_0 places unit mass on state 1.

of those in the right chart. In this case, any policy results in an expected total reward of zero, and randomisation becomes ineffective.

We now show that in addition to randomisation, the optimal policy may require history dependence if the uncertainty set lacks s-rectangularity.

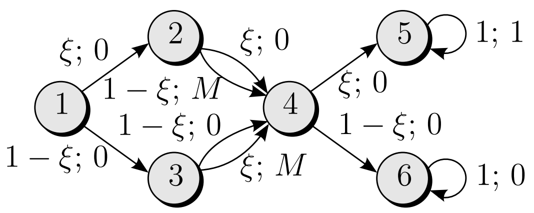
Proposition 6.2.2 (General Uncertainty Sets) *For finite and infinite horizon MDPs, the policy improvement problem over non-rectangular uncertainty sets is in general solved by non-Markovian policies.*

Proof Consider the robust infinite horizon MDP with six states and two actions that is visualised in Figure 6.4. The uncertainty set $\mathcal{P}$ encompasses all transition kernels that correspond to parameter realisations $\xi \in [0,1]$. This MDP can be assigned an uncertainty set of the form (6.3). Since the transitions do not depend on the chosen actions except for π_2, a policy is completely determined by the decision $\beta = (\beta_1, \beta_2)$, where $\beta_1 = \pi_2(1|1, a_0, 2, a_1; 4)$ and $\beta_2 = \pi_2(1|1, a_0, 3, a_1; 4)$.

The conditional probability to reach state 5 is $\varphi_1(\xi) := \beta_1\xi + (1-\beta_1)(1-\xi)$ if state 2 is visited and $\varphi_2(\xi) := \beta_2\xi + (1-\beta_2)(1-\xi)$ if state 3 is visited, respectively. Thus, the expected total reward amounts to

$$2\lambda\xi(1-\xi)M + \frac{\lambda^3}{1-\lambda}\left[\xi\,\varphi_1(\xi) + (1-\xi)\,\varphi_2(\xi)\right],$$

which is concave in ξ for all $\beta \in [0,1]^2$ if $M \geq \lambda^2/(1-\lambda)$. Thus, the worst (minimal) expected total reward is incurred for $\xi^* \in \{0,1\}$, independently of $\beta \in [0,1]^2$. Hence, the worst-case



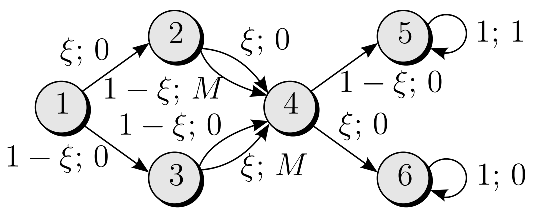


Figure 6.4: MDP with six states and two actions. The initial state distribution p_0 places unit mass on state 1. The same drawing conventions as in Figure 6.3 are used.

expected total reward is

$$\min_{\xi \in \{0,1\}} \frac{\lambda^3}{1-\lambda} \left[\xi\, \varphi_1(\xi) + (1-\xi)\, \varphi_2(\xi)\right] = \frac{\lambda^3}{1-\lambda} \min\left\{\beta_1, 1-\beta_2\right\},$$

and the unique maximiser of this expression is $\beta = (1, 0)$. We conclude that in state 4, the optimal policy chooses action 1 if state 2 has been visited and action 2 otherwise. Hence, the optimal policy is history dependent. If we replace the self-loops with expected terminal rewards of $\mathbf{r}_5 := \lambda^3/(1-\lambda)$ and $\mathbf{r}_6 := 0$, then we can extend the result to robust finite horizon MDPs. ∎

Although the policy improvement problem over non-rectangular uncertainty sets is in general solved by non-Markovian policies, we will restrict ourselves to stationary policies in the remainder. Thus, we will be interested in the best deterministic or randomised stationary policies for robust MDPs.

6.2.3 Complexity of the Robust Policy Evaluation Problem

We show that the policy evaluation problem over non-rectangular uncertainty sets is strongly $\mathcal{NP}$-hard. To this end, we will reduce the evaluation of (6.2) to the 0/1 Integer Programming (IP) problem [GJ79]:

0/1 Integer Programming.

Instance. Given are $F \in \mathbb{Z}^{m \times n}$, $g \in \mathbb{Z}^m$, $c \in \mathbb{Z}^n$, $\zeta \in \mathbb{Z}$.

Question. Is there a vector $x \in \{0,1\}^n$ such that $Fx \leq g$ and $c^\top x \leq \zeta$?

Assume that $x \in [0,1]^n$ constitutes a fractional vector that satisfies $Fx \leq g$ and $c^\top x \leq \zeta$. The following lemma shows that we can obtain an integral vector $y \in \{0,1\}^n$ that satisfies $Fy \leq g$ and $c^\top y \leq \zeta$ by rounding x if its components are 'close enough' to zero or one.

Lemma 6.2.1 *Let $0 < \epsilon \leq \min\{\epsilon_F, \epsilon_c\}$, where $0 < \epsilon_F < \min_i \left\{ \left(\sum_j |F_{ij}| \right)^{-1} \right\}$ and $0 < \epsilon_c < \left(\sum_j |c_j| \right)^{-1}$. Assume that $x \in ([0,\epsilon] \cup [1-\epsilon, 1])^n$ satisfies $Fx \leq g$ and $c^\top x \leq \zeta$. Then $Fy \leq g$ and $c^\top y \leq \zeta$ for $y \in \{0,1\}^n$, where $y_j := 1$ if $x_j \geq 1 - \epsilon$ and $y_j := 0$ otherwise.*

Proof By construction, $F_{i\cdot}^\top y \leq F_{i\cdot}^\top x + \sum_j |F_{ij}| \, \epsilon_F < F_{i\cdot}^\top x + 1 \leq g_i + 1$ for all $i \in \{1, \ldots, m\}$. Similarly, we have that $c^\top y \leq c^\top x + \sum_j |c_j| \, \epsilon_c < c^\top x + 1 \leq \zeta + 1$. Due to the integrality of F, g, c, ζ and y, we therefore conclude that $Fy \leq g$ and $c^\top y \leq \zeta$. ∎

We can now prove strong $\mathcal{NP}$-hardness of the policy evaluation problem.

Theorem 6.2.1 *Deciding whether the worst-case expected total reward (6.2) over an uncertainty set of the form (6.3) exceeds a given value γ is strongly $\mathcal{NP}$-hard for deterministic as well as randomised stationary policies and for finite as well as infinite horizon MDPs.*

Proof Let us fix an IP instance specified through F, g, c and ζ. Without loss of generality, we can assume that $\zeta \leq \sum_j [c_j]^+$ because all feasible IP solutions are binary. We construct a reduction to a robust infinite horizon MDP as follows. The states are $\mathcal{S} = \left\{ b_j, b_j^1, b_j^0 \, : \, j = 1, \ldots, n \right\} \cup \{c_0, \tau\}$, there is only one action, and the discount factor $\lambda \in (0,1)$ can be chosen freely. The state transitions and expected rewards are illustrated in Figure 6.5. The uncertainty set $\mathcal{P}$ contains all transition kernels associated with $\xi \in [0,1]^n$ that satisfy $F\xi \leq g$. We choose $M > \left(\lambda n \sum_j |c_j| \right) / \left(2\epsilon^2 \right)$, where ϵ is chosen as in Lemma 6.2.1, and set $\gamma := \lambda^2 \zeta$. Following our discussion in Section 6.2.1, the described MDP instance can be constructed in polynomial time with respect to the size of the IP instance (which we abbreviate as 'in polynomial time' in the remainder of this proof).[1]

[1]Note that the set Ξ associated with the MDP instance might not contain a Slater point. However, one can decide in polynomial time whether the system of linear equations $Fx \leq g$, $x \in [0,1]^n$ is strictly feasible. If this is not the case, one can furthermore reduce the system to a strictly feasible one in polynomial time.

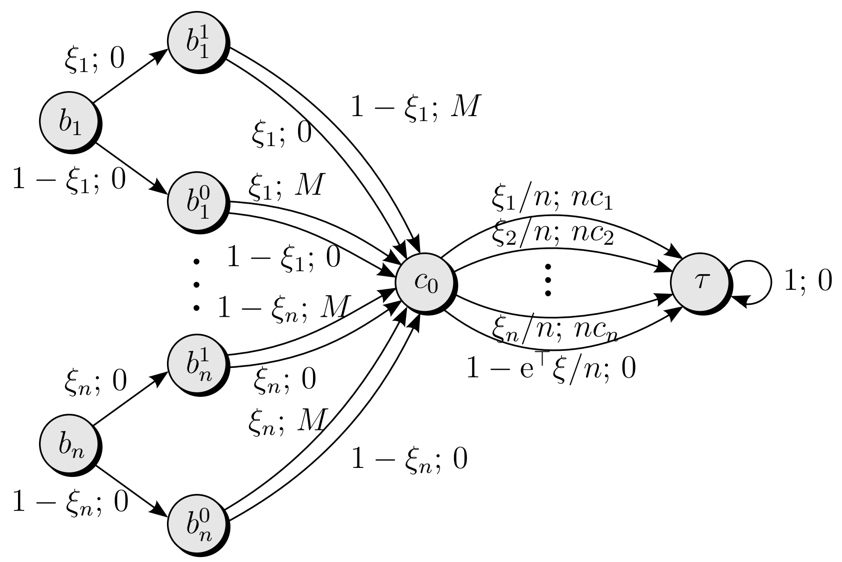


Figure 6.5: MDP with $3n+2$ states and one action. The distribution p_0 places a probability mass of $1/n$ on each state b_j, $j=1,\ldots,n$. The drawing conventions from Figure 6.3 are used.

We show that the answer to the IP instance is affirmative if and only if the worst-case expected total reward (6.2) does not exceed γ. Indeed, assume that the answer to the IP instance is affirmative, that is, there is a vector $x \in \{0,1\}^n$ that satisfies $Fx \leq g$ and $c^\top x \leq \zeta$. The transition kernel associated with $\xi = x$ is contained in $\mathcal{P}$ and leads to an expected total reward of $\lambda^2 c^\top \xi \leq \lambda^2 \zeta = \gamma$. This implies that the worst-case expected total reward (6.2) does not exceed γ either. Conversely, assume that (6.2) does not exceed γ. For the constructed MDP, the expected total reward (6.1) is continuous in ξ. Since $\mathcal{P}$ is compact, we can therefore assume that the value of (6.2) is attained by a transition kernel associated with some $\xi^* \in \Xi$. By construction of Ξ, ξ^* satisfies $\xi^* \in [0,1]^n$ and $F\xi^* \leq g$. Assume that $\xi^*_q \notin ([0,\epsilon] \cup [1-\epsilon,1])$ for some $q \in \{1,\ldots,n\}$. In this case, the expected total reward under ξ^* is greater than or equal to $2\lambda\xi^*_q(1-\xi^*_q)M/n - \lambda^2 \sum_j [-c_j]^+ > \lambda^2 \sum_j [c_j]^+ \geq \gamma$, which contradicts our assumption. We have thus established that $\xi^* \in ([0,\epsilon] \cup [1-\epsilon,1])^n$. Under the transition kernel associated with ξ^*, the expected reward in periods 0 and 1 is guaranteed to be non-negative, while the expected reward from period 2 onward amounts to $\lambda^2 c^\top \xi^*$. Since the expected total reward under ξ^* does not exceed γ, we therefore have that $\lambda^2 c^\top \xi^* \leq \gamma = \lambda^2 \zeta$, which implies that $c^\top \xi^* \leq \zeta$. Hence, we can apply Lemma 6.2.1 to obtain a vector $\xi' \in \{0,1\}^n$ that also satisfies $F\xi' \leq g$ and $c^\top \xi' \leq \zeta$. We have thus shown that the answer to the IP instance is affirmative if and only if the worst-case expected total reward (6.2) does not exceed γ.

uncertainty set $\mathcal{P}$	optimal policy	complexity
(s, a)-rectangular, convex	deterministic, stationary	polynomial
(s, a)-rectangular, nonconvex	deterministic, stationary	strongly $\mathcal{NP}$-hard
s-rectangular, convex	randomised, stationary	polynomial
s-rectangular, nonconvex	randomised, history dependent	strongly $\mathcal{NP}$-hard
non-rectangular, convex	randomised, history dependent	strongly $\mathcal{NP}$-hard

Table 6.1: Properties of infinite horizon MDPs with different uncertainty sets. From left to right, the columns describe the structure of the uncertainty set, the structure of the optimal policy, and the complexity of the policy evaluation and improvement problems over randomised stationary policies. Each uncertainty set is of the form (6.3). For nonconvex uncertainty sets, we do not require the matrices O_l in (6.3b) to be negative semidefinite. The properties of finite horizon MDPs are similar, the only difference being that MDPs with s-rectangular nonconvex uncertainty sets are optimised by randomised stationary policies.

If we could decide in polynomial time whether the worst-case expected total reward of the constructed MDP exceeds γ, we could also decide IP in polynomial time. Since IP is strongly $\mathcal{NP}$-hard [GJ79], we conclude that the policy evaluation problem (6.2) is strongly $\mathcal{NP}$-hard for MDPs with a single action and uncertainty sets of the form (6.3). Since the policy space of the constructed MDP reduces to a singleton, our proof applies to robust MDPs with deterministic and randomised stationary policies. If we remove the self-loop emanating from state τ, introduce a terminal reward $\mathbf{r}_\tau := 0$ and multiply the rewards in period t with λ^{-t}, our proof furthermore applies to robust finite horizon MDPs. ∎

Remark 6.2.1 *Theorem 6.2.1 remains valid if definition (6.3) is altered to require that $O_l = 0$ and $o_l \in \{0, 1\}^q$. This follows from the fact that IP remains strongly $\mathcal{NP}$-hard if F and g are binary, see [GJ79].*

Remark 6.2.2 *Throughout this section we assumed that $\mathcal{P}$ is a convex set of the form (6.3). If we extend our analysis to nonconvex uncertainty sets, then we obtain the results in Table 6.1. Note that the complexity of some of the policy evaluation and improvement problems will be discussed in Sections 6.3 and 6.4.*

6.3 Robust Policy Evaluation

It is shown in [Iye05, NG05] that the worst-case expected total reward (6.2) can be calculated in polynomial time for certain types of (s, a)-rectangular uncertainty sets. We extend this result to the broader class of s-rectangular uncertainty sets in Section 6.3.1. On the other hand, Theorem 6.2.1 shows that the evaluation of (6.2) is strongly $\mathcal{NP}$-hard for non-rectangular uncertainty sets. We therefore develop conservative approximations for the policy evaluation problem over general uncertainty sets in Section 6.3.2. We bound the optimality gap that is incurred by solving these approximations, and we outline how these approximations can be refined. Although this section primarily sets the stage for the policy improvement problem, we stress that policy evaluation is an important problem in its own right. For example, it finds frequent use in labour economics, industrial organisation and marketing [MSST07].

Our solution approaches for s-rectangular and non-rectangular uncertainty sets rely on the reward to-go function. For a stationary policy π, we define the *reward to-go* function $v : \Pi \times \Xi \mapsto \mathbb{R}^S$ through

$$v_s(\pi; \xi) = \mathbb{E}^{p^\xi, \pi} \left[\sum_{t=0}^{\infty} \lambda^t r(s_t, a_t, s_{t+1}) \,\middle|\, s_0 = s \right] \qquad \text{for } s \in \mathcal{S}. \tag{6.5}$$

$v_s(\pi; \xi)$ represents the expected total reward under the transition kernel p^ξ and the policy π if the initial state is $s \in \mathcal{S}$. The reward to-go function allows us to express the worst-case expected total reward as

$$\inf_{\xi \in \Xi} \mathbb{E}^{p^\xi, \pi} \left[\sum_{t=0}^{\infty} \lambda^t r(s_t, a_t, s_{t+1}) \,\middle|\, s_0 \sim p_0 \right] = \inf_{\xi \in \Xi} \left\{ p_0^\top v(\pi; \xi) \right\}. \tag{6.6}$$

We simplify our notation by defining the Markov reward process (MRP) induced by p^ξ and π. MRPs are Markov chains which pay a state-dependent reward at each time period. In our case, the MRP is given by the transition kernel $\widehat{P} : \Pi \times \Xi \overset{\text{a}}{\mapsto} \mathbb{R}^{S \times S}$ and the expected state rewards

$\widehat{r} : \Pi \times \Xi \overset{\text{a}}{\mapsto} \mathbb{R}^S$ defined through

$$\widehat{P}_{ss'}(\pi;\xi) := \sum_{a\in\mathcal{A}} \pi(a|s)\, p^{\xi}(s'|s,a) \tag{6.7a}$$

and
$$\widehat{r}_s(\pi;\xi) := \sum_{a\in\mathcal{A}} \pi(a|s) \sum_{s'\in\mathcal{S}} p^{\xi}(s'|s,a)\, r(s,a,s'). \tag{6.7b}$$

Note that $\widehat{r}(\pi;\xi) \geq 0$ for each $\pi \in \Pi$ and $\xi \in \Xi$ since all expected rewards $r(s,a,s')$ were assumed to be non-negative. For $s, s' \in \mathcal{S}$, $\widehat{P}_{ss'}(\pi;\xi)$ denotes the probability that the next state of the MRP is s', given that the MRP is currently in state s. Likewise, $\widehat{r}_s(\pi;\xi)$ denotes the expected reward that is received in state s. By taking the expectation with respect to the sample paths of the MRP and reordering terms, we can reformulate (6.5) as

$$v(\pi;\xi) = \sum_{t=0}^{\infty} \left[\lambda\, \widehat{P}(\pi;\xi)\right]^t \widehat{r}(\pi;\xi), \tag{6.8}$$

see [Put94]. The following proposition brings together results about v that we will use later on.

Proposition 6.3.1 *The reward to-go function v has the following properties.*

(a) v is Lipschitz continuous on $\Pi \times \Xi$.

(b) For given $\pi \in \Pi$ and $\xi \in \Xi$, $w \in \mathbb{R}^S$ satisfies $w = \widehat{r}(\pi;\xi) + \lambda\, \widehat{P}(\pi;\xi)\, w$ if and only if $w = v(\pi;\xi)$.

(c) For given $\pi \in \Pi$ and $\xi \in \Xi$, if $w \in \mathbb{R}^S$ satisfies $w \leq \widehat{r}(\pi;\xi) + \lambda\, \widehat{P}(\pi;\xi)\, w$, then $w \leq v(\pi;\xi)$.

Proof For a square matrix $A \in \mathbb{R}^{n\times n}$, let $\mathrm{Adj}(A)$ and $\det(A)$ denote the adjugate matrix and the determinant of A, respectively. From equation (6.8), we see that

$$v(\pi;\xi) \;=\; \left[I - \lambda\, \widehat{P}(\pi;\xi)\right]^{-1} \widehat{r}(\pi;\xi) \;=\; \frac{\mathrm{Adj}\left(I - \lambda\, \widehat{P}(\pi;\xi)\right) \widehat{r}(\pi;\xi)}{\det\left(I - \lambda\, \widehat{P}(\pi;\xi)\right)} \qquad \forall\, \xi \in \Xi. \tag{6.9}$$

Here, the first identity follows from the matrix inversion lemma, see Theorem C.2 in [Put94], while the second equality is due to Cramer's rule. The adjugate matrix and the determinant in (6.9) constitute polynomials in π and ξ, and the matrix inversion lemma guarantees that the determinant is nonzero throughout Ξ. Hence, the fraction on the right hand-side of (6.9) has bounded first derivative on $\Pi \times \Xi$, which implies that it is Lipschitz continuous on $\Pi \times \Xi$. We have thus proven assertion (a).

Assertions (b) and (c) follow directly from Theorems 6.1.1 and 6.2.2 in [Put94], respectively. ∎

Proposition 6.3.1 allows us to reformulate the worst-case expected total reward (6.6) as follows.

$$\begin{aligned}
\inf_{\xi \in \Xi} \left\{ p_0^\top v(\pi;\xi) \right\} &= \inf_{\xi \in \Xi} \sup_{w \in \mathbb{R}^S} \left\{ p_0^\top w \, : \, w \leq \widehat{r}(\pi;\xi) + \lambda \, \widehat{P}(\pi;\xi) \, w \right\} \\
&= \sup_{\vartheta : \Xi \mapsto \mathbb{R}^S} \left\{ \inf_{\xi \in \Xi} \left\{ p_0^\top \vartheta(\xi) \right\} \, : \, \vartheta(\xi) \leq \widehat{r}(\pi;\xi) + \lambda \, \widehat{P}(\pi;\xi) \, \vartheta(\xi) \ \ \forall \, \xi \in \Xi \right\} \\
&= \sup_{\vartheta : \Xi \overset{c}{\mapsto} \mathbb{R}^S} \left\{ \inf_{\xi \in \Xi} \left\{ p_0^\top \vartheta(\xi) \right\} \, : \, \vartheta(\xi) \leq \widehat{r}(\pi;\xi) + \lambda \, \widehat{P}(\pi;\xi) \, \vartheta(\xi) \ \ \forall \, \xi \in \Xi \right\} \qquad (6.10)
\end{aligned}$$

Here, the first equality follows from Proposition 6.3.1 (b)–(c) and non-negativity of p_0, while the last equality follows from Proposition 6.3.1 (a). The second equality follows from the identity (2.5a) in Section 2.2.2. Theorem 6.2.1 implies that (6.10) is intractable for general uncertainty sets. In the following, we approximate (6.10) by replacing the space of continuous functions in the outer supremum with the subspaces of constant, affine and piecewise affine functions. Since the policy π is fixed in this section, we may omit the dependence of v, $\widehat{P}$ and $\widehat{r}$ on π in the following.

6.3.1 Robust Policy Evaluation over s-Rectangular Uncertainty Sets

We show that the policy evaluation problem (6.10) is optimised by a constant reward to-go function if the uncertainty set $\mathcal{P}$ is s-rectangular. The result also points out an efficient method to solve problem (6.10).

Theorem 6.3.1 *For an s-rectangular uncertainty set $\mathcal{P}$, the policy evaluation problem (6.10) is optimised by the constant reward to-go function $\vartheta^*(\xi) := w^*$, $\xi \in \Xi$, where $w^* \in \mathbb{R}^S$ is the* *unique fixed point of the contraction mapping* $\phi(\pi;\cdot) : \mathbb{R}^S \mapsto \mathbb{R}^S$ *defined through*

$$\phi_s(\pi; w) := \min_{\xi^s \in \Xi} \left\{ \widehat{r}_s(\pi; \xi^s) + \lambda \widehat{P}_{s\cdot}^\top(\pi; \xi^s)\, w \right\} \qquad \forall\, s \in \mathcal{S}. \tag{6.11}$$

Remark 6.3.1 *A function* $\varphi : \mathbb{R}^S \mapsto \mathbb{R}^S$ *is called* contraction mapping *if there is some* $\gamma \in [0,1)$ *such that* $\|\varphi(w) - \varphi(w')\| \leq \gamma\, \|w - w'\|$ *for all* $w, w' \in \mathbb{R}^S$. *The iterated application of* φ *to any* $w \in \mathbb{R}^S$ *converges to the unique fixed point* w^* *that satisfies* $w^* = \varphi(w^*)$, *see [Put94].*

Proof of Theorem 6.3.1 We prove the assertion in two steps. We first show that w^* solves the restriction of the policy evaluation problem (6.10) to constant reward to-go functions:

$$\sup_{w \in \mathbb{R}^S} \left\{ p_0^\top w \;:\; w \leq \widehat{r}(\xi) + \lambda \widehat{P}(\xi)\, w \;\; \forall\, \xi \in \Xi \right\} \tag{6.12}$$

Afterwards, we prove that the optimal values of (6.10) and (6.12) coincide for s-rectangular uncertainty sets.

In view of the first step, we note that the objective function of (6.12) is linear in w. Moreover, the feasible region of (6.12) is closed because it results from the intersection of closed halfspaces parametrised by $\xi \in \Xi$. Since $w = 0$ is feasible in (6.12), we can append the constraint $w \geq 0$ without changing the optimal value of (6.12). Hence, the feasible region is also bounded, and we can apply Weierstrass' extreme value theorem to replace the supremum in (6.12) with a maximum. Since each of the S one-dimensional inequality constraints in (6.12) has to be satisfied for all $\xi \in \Xi$, (6.12) is equivalent to

$$\max_{w \in \mathbb{R}^S} \left\{ p_0^\top w \;:\; w_s \leq \widehat{r}_s(\xi^s) + \lambda \widehat{P}_{s\cdot}^\top(\xi^s)\, w \;\; \forall\, s \in \mathcal{S},\; \xi^1, \ldots, \xi^S \in \Xi \right\}.$$

We can reformulate the semi-infinite constraints in this problem to obtain

$$\max_{w \in \mathbb{R}^S} \left\{ p_0^\top w \;:\; w_s \leq \min_{\xi^s \in \Xi} \left\{ \widehat{r}_s(\xi^s) + \lambda \widehat{P}_{s\cdot}^\top(\xi^s)\, w \right\} \;\; \forall\, s \in \mathcal{S} \right\}. \tag{6.13}$$

Note that the constraints in (6.13) are equivalent to $w \leq \phi(\pi; w)$, where ϕ is defined in (6.11). One can adapt the results in [Iye05, NG05] to show that $\phi(\pi;\cdot)$ is a contraction mapping.

Hence, the Banach fixed point theorem guarantees existence and uniqueness of $w^* \in \mathbb{R}^S$. This vector w^* is feasible in (6.13), and any feasible solution $w \in \mathbb{R}^S$ to (6.13) satisfies $w \leq \phi(\pi; w)$. According to Theorem 6.2.2 in [Put94], this implies that $w^* \geq w$ for every feasible solution w to (6.13). By non-negativity of p_0, w^* must therefore maximise (6.13). Since (6.12) and (6.13) are equivalent, we have thus shown that w^* maximises (6.12).

We now prove that the optimal values of (6.10) and (6.13) coincide if $\mathcal{P}$ is s-rectangular. Since (6.13) is maximised by the unique fixed point w^* of $\phi(\pi; \cdot)$, we can reexpress (6.13) as

$$\min_{w \in \mathbb{R}^S} \left\{ p_0^\top w \,:\, w_s = \min_{\xi^s \in \Xi} \left\{ \widehat{r}_s(\xi^s) + \lambda \widehat{P}_{s\cdot}^\top(\xi^s)\, w \right\} \;\; \forall\, s \in \mathcal{S} \right\}.$$

Since p_0 is non-negative, this problem is equivalent to

$$\min_{w \in \mathbb{R}^S} \min_{\substack{\xi^s \in \Xi: \\ s \in \mathcal{S}}} \left\{ p_0^\top w \,:\, w_s = \widehat{r}_s(\xi^s) + \lambda \widehat{P}_{s\cdot}^\top(\xi^s)\, w \;\; \forall\, s \in \mathcal{S} \right\}. \tag{6.14}$$

The s-rectangularity of the uncertainty set $\mathcal{P}$ implies that (6.14) can be reformulated as

$$\min_{w \in \mathbb{R}^S} \min_{\xi \in \Xi} \left\{ p_0^\top w \,:\, w_s = \widehat{r}_s(\xi) + \lambda \widehat{P}_{s\cdot}^\top(\xi)\, w \;\; \forall\, s \in \mathcal{S} \right\}. \tag{6.15}$$

For a fixed $\xi \in \Xi$, $w = v(\xi)$ is the unique feasible solution to (6.15), see Proposition 6.3.1 (b). By Weierstrass' extreme value theorem, (6.15) is therefore equivalent to the policy evaluation problem (6.10). ∎

The fixed point w^* of the contraction mapping $\phi(\pi; \cdot)$ defined in (6.11) can be found by applying the following *robust value iteration*. We start with an initial estimate $w^1 := 0$. In the ith iteration, $i = 1, 2, \ldots$, we determine the updated estimate w^{i+1} via $w^{i+1} := \phi(\pi; w^i)$. Since $\phi(\pi; \cdot)$ is a contraction mapping, the Banach fixed point theorem guarantees that the sequence w^i converges to w^* at a geometric rate. The following corollary investigates the computational complexity of this approach.

Corollary 6.3.1 *If the uncertainty set $\mathcal{P}$ is s-rectangular, then problem (6.10) can be solved to any accuracy ϵ in polynomial time $\mathcal{O}\left(q^3 L^{3/2} S \log^2 \epsilon^{-1} + qAS^2 \log \epsilon^{-1}\right)$.*

Proof Assume that at each iteration i of the robust value iteration, we evaluate $\phi(\pi; w^i)$ to the accuracy $\delta := \epsilon(1-\lambda)^2/(4+4\lambda)$. We stop the algorithm as soon as $\left\|w^{N+1} - w^N\right\|_\infty \leq \epsilon(1-\lambda)/(1+\lambda)$ at some iteration N. This is guaranteed to happen within $\mathcal{O}\left(\log \epsilon^{-1}\right)$ iterations [Put94]. By construction, w^{N+1} is feasible for the policy evaluation problem (6.10), see [Put94]. We can adapt Theorem 5 from [NG05] to show that w^{N+1} satisfies $\left\|w^{N+1} - w^*\right\|_\infty \leq \epsilon$. Hence, w^{N+1} is also an ϵ-optimal solution to (6.10).

We now investigate the complexity of evaluating ϕ to the accuracy δ. Under mild assumptions, interior point methods can solve second-order cone programs of the form

$$\min_{x \in \mathbb{R}^n} \left\{ f^\top x \; : \; \|A_j x + b_j\|_2 \leq c_j^\top x + d_j \quad \forall\, j = 1, \ldots, m \right\},$$

where $A_j \in \mathbb{R}^{n_j \times n}$, $b_j \in \mathbb{R}^{n_j}$, $c_j \in \mathbb{R}^n$ and $d_j \in \mathbb{R}$, $j = 1, \ldots, m$, to any accuracy δ in polynomial time $\mathcal{O}\left(\sqrt{m}\left[n^3 + n^2 \sum_j n_j\right] \log \delta^{-1}\right)$, see [LVBL98]. For $w \in \mathbb{R}^S$, we can evaluate $\phi(\pi; w)$ by solving the following second-order cone program:

$$\begin{aligned}
&\underset{\xi}{\text{minimise}} && \sum_{a \in \mathcal{A}} \pi(a|s)\left(k_{sa} + K_{sa}\xi\right)^\top \left(r_{sa} + \lambda w\right) && && \text{(6.16a)}\\
&\text{subject to} && \xi \in \mathbb{R}^q && &&\\
& && \left\| \begin{bmatrix} \Omega_l \\ -o_l^\top \end{bmatrix} \xi + \begin{bmatrix} 0 \\ \frac{1-\omega_l}{2} \end{bmatrix} \right\|_2 \leq o_l^\top \xi + \frac{\omega_l + 1}{2} && \forall\, l = 1, \ldots, L, && \text{(6.16b)}
\end{aligned}$$

where $(r_{sa})_{s'} := r(s, a, s')$ for $(s, a, s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ and Ω_l satisfies $\Omega_l^\top \Omega_l = -O_l$. We can determine each matrix Ω_l in time $\mathcal{O}\left(q^3\right)$ by a Cholesky decomposition, we can construct (6.16) in time $\mathcal{O}\left(qAS + q^2 L\right)$, and we can solve (6.16) to accuracy δ in time $\mathcal{O}\left(q^3 L^{3/2} \log \delta^{-1}\right)$. Each step of the robust value iteration requires the construction and solution of S such problems. Since the constraints of (6.16) only need to be generated once, this results in an iteration complexity of $\mathcal{O}\left(q^3 L^{3/2} S \log \delta^{-1} + qAS^2\right)$. The assertion now follows from the fact that the robust value iteration terminates within $\mathcal{O}\left(\log \epsilon^{-1}\right)$ iterations. ∎

Depending on the properties of Ξ defined in (6.3b), we can evaluate the mapping ϕ more efficiently. We refer to [Iye05, NG05] for a discussion of different numerical schemes.

Remark 6.3.2 (Finite Horizon MDPs) *For a finite horizon MDP, we can solve the policy evaluation problem (6.10) over an s-rectangular uncertainty set $\mathcal{P}$ via* robust backward induction *as follows. We start with $w^T \in \mathbb{R}^S$ defined through $w^T_s := \mathbf{r}_s$ if $s \in \mathcal{S}_T$; $:= 0$ otherwise. At iteration $i = T-1, T-2, \ldots, 1$, we determine w^i through $w^i_s := \widehat{\phi}_s(\pi; w^{i+1})$ if $s \in \mathcal{S}_i$; $:= w^{i+1}_s$ otherwise. The operator $\widehat{\phi}$ is defined as*

$$\widehat{\phi}_s(\pi; w) := \min_{\xi^s \in \Xi} \left\{ \widehat{r}_s(\pi; \xi^s) + \widehat{P}_{s\cdot}^\top(\pi; \xi^s)\, w \right\} \qquad \forall\, s \in \mathcal{S}.$$

An adaptation of Corollary 6.3.1 shows that we obtain an ϵ-optimal solution to the policy evaluation problem (6.10) in polynomial time $\mathcal{O}\left(q^3 L^{3/2} S \log \epsilon^{-1} + qAS^2\right)$ if we evaluate $\widehat{\phi}$ to the accuracy $\epsilon/(T-1)$.

We close with an example that illustrates the solution of the policy evaluation problem (6.10) for s-rectangular uncertainty sets.

Example 6.3.1 *Consider again the robust infinite horizon MDP defined in Proposition 6.2.1 and visualised in Figure 6.3. The state set of the MDP is $\mathcal{S} = \{1,2,3\}$, the set of admissible actions is $\mathcal{A} = \{1,2\}$, and the initial state distribution is given by $p_0 = \mathrm{e}_1$. The uncertainty set Ξ is specified by*

$$\Xi = \{\xi \in \mathbb{R} \,:\, 0 \leq \xi \leq 1\} \;=\; \left\{\xi \in \mathbb{R} \,:\, (\xi - 1/2)^2 \leq (1/2)^2\right\} \;=\; \left\{\xi \in \mathbb{R} \,:\, -\xi^2 + \xi \geq 0\right\},$$

that is, we have $L = 1$, $O_1 = -1$, $o_1 = 1$ and $\omega_1 = 0$. The transition probabilities are described by $p(\cdot|1,1) = (0,\ \xi,\ 1-\xi)^\top$, $p(\cdot|1,2) = (0,\ 1-\xi,\ \xi)^\top$ and $p(\cdot|s,a) = \mathrm{e}_s$ for $s \in \{2,3\}$, $a \in \mathcal{A}$. In the notation of Section 6.2.1, we therefore have

$$\begin{aligned} & (k_{11}, K_{11}) = (\mathrm{e}_3,\ \mathrm{e}_2 - \mathrm{e}_3)\,, && (k_{12}, K_{12}) = (\mathrm{e}_2,\ \mathrm{e}_3 - \mathrm{e}_2) \\ \textit{and} \quad & (k_{sa}, K_{sa}) = (\mathrm{e}_s,\ 0) && \textit{for } s \in \{2,3\}\,,\ a \in \mathcal{A}. \end{aligned}$$

The reward is given by $r_{1,a} = r_{3,a} = 0$ and $r_{2,a} = \mathrm{e}_2$, $a \in \mathcal{A}$. For the discount factor $\lambda = 0.9$ and the policy $\pi(a|s) = 1/2$, $(s,a) \in \mathcal{S} \times \mathcal{A}$, the first component of $\phi(\pi; w)$ is identical to the optimal value of the following optimisation problem, see (6.16).

$$\begin{array}{ll} \underset{\xi}{\text{minimise}} & \frac{1}{2}\left(\begin{bmatrix}0\\0\\1\end{bmatrix} + \begin{bmatrix}0\\1\\-1\end{bmatrix}\xi\right)^{\top}\left(\begin{bmatrix}0\\0\\0\end{bmatrix} + 0.9\begin{bmatrix}w_1\\w_2\\w_3\end{bmatrix}\right) + \frac{1}{2}\left(\begin{bmatrix}0\\1\\0\end{bmatrix} + \begin{bmatrix}0\\-1\\1\end{bmatrix}\xi\right)^{\top}\left(\begin{bmatrix}0\\0\\0\end{bmatrix} + 0.9\begin{bmatrix}w_1\\w_2\\w_3\end{bmatrix}\right) \\ \text{subject to} & \xi \in \mathbb{R} \\ & \left\|\begin{bmatrix}1\\-1\end{bmatrix}\xi + \begin{bmatrix}0\\\frac{1}{2}\end{bmatrix}\right\|_2 \leq \xi + \frac{1}{2}. \end{array}$$

Likewise, $\phi_2(\pi; w)$ is equal to the optimal value of the following problem.

$$\begin{array}{ll} \underset{\xi}{\text{minimise}} & \frac{1}{2}\left(\begin{bmatrix}0\\1\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\end{bmatrix}\xi\right)^{\top}\left(\begin{bmatrix}0\\1\\0\end{bmatrix} + 0.9\begin{bmatrix}w_1\\w_2\\w_3\end{bmatrix}\right) + \frac{1}{2}\left(\begin{bmatrix}0\\1\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\end{bmatrix}\xi\right)^{\top}\left(\begin{bmatrix}0\\1\\0\end{bmatrix} + 0.9\begin{bmatrix}w_1\\w_2\\w_3\end{bmatrix}\right) \\ \text{subject to} & \xi \in \mathbb{R} \\ & \left\|\begin{bmatrix}1\\-1\end{bmatrix}\xi + \begin{bmatrix}0\\\frac{1}{2}\end{bmatrix}\right\|_2 \leq \xi + \frac{1}{2}. \end{array}$$

The third component of $\phi(\pi; w)$, finally, is identical to the optimal value of the following problem.

$$\begin{array}{ll} \underset{\xi}{\text{minimise}} & \frac{1}{2}\left(\begin{bmatrix}0\\0\\1\end{bmatrix} + \begin{bmatrix}0\\0\\0\end{bmatrix}\xi\right)^{\top}\left(\begin{bmatrix}0\\0\\0\end{bmatrix} + 0.9\begin{bmatrix}w_1\\w_2\\w_3\end{bmatrix}\right) + \frac{1}{2}\left(\begin{bmatrix}0\\0\\1\end{bmatrix} + \begin{bmatrix}0\\0\\0\end{bmatrix}\xi\right)^{\top}\left(\begin{bmatrix}0\\0\\0\end{bmatrix} + 0.9\begin{bmatrix}w_1\\w_2\\w_3\end{bmatrix}\right) \\ \text{subject to} & \xi \in \mathbb{R} \\ & \left\|\begin{bmatrix}1\\-1\end{bmatrix}\xi + \begin{bmatrix}0\\\frac{1}{2}\end{bmatrix}\right\|_2 \leq \xi + \frac{1}{2}. \end{array}$$

If we start with the initial estimate $w^1 := 0$*, we obtain* $w^2 := \phi(\pi; w^1) = (0, 1, 0)^\top$*,* $w^3 := \phi(\pi; w^2) = (0.45, 1.9, 0)^\top$*,* $w^4 := \phi(\pi; w^3) \approx (0.86, 2.71, 0)^\top$ *and so on. If we want to solve the policy evaluation problem (6.10) to the accuracy* $\epsilon := 10^{-3}$*, then we have to execute the robust value iteration until* $\left\| w^{N+1} - w^N \right\|_\infty \leq 10^{-3} \cdot 0.1/1.1 \approx 9.1 \cdot 10^{-5}$ *at some iteration* N*. For our example, the robust value iteration takes* $N = 90$ *iterations to determine the fixed point* $w^* = (4.5, 10, 0)^\top$*. Since the initial state distribution is* $p_0 = \mathrm{e}_1$*, the optimal value of the robust policy evaluation problem (6.10) is* 4.5.

6.3.2 Robust Policy Evaluation over Non-Rectangular Uncertainty Sets

If the uncertainty set $\mathcal{P}$ is non-rectangular, then Theorem 6.2.1 implies that constant reward to-go functions are no longer guaranteed to optimise the policy evaluation problem (6.10). Nevertheless, we can still use the robust value iteration to obtain a lower bound on the optimal value of (6.10).

Proposition 6.3.2 *Let* $\mathcal{P}$ *be a non-rectangular uncertainty set, and define* $\overline{\mathcal{P}} := \times_{s \in \mathcal{S}} \mathcal{P}_s$ *as the smallest s-rectangular uncertainty set that contains* $\mathcal{P}$*. The function* $\vartheta^*(\xi) = w^*$ *defined in Theorem 6.3.1 has the following properties.*

1. *The vector* w^* *solves the restriction (6.12) of the policy evaluation problem (6.10) that approximates the reward to-go function by a constant.*

2. *The function* ϑ^* *solves the exact policy evaluation problem (6.10) over* $\overline{\mathcal{P}}$*.*

Proof The first property follows from the fact that the first part of the proof of Theorem 6.3.1 does not depend on the structure of the uncertainty set $\mathcal{P}$. As for the second property, the proof of Theorem 6.3.1 shows that w^* minimises (6.14), irrespective of the structure of $\mathcal{P}$. The proof also shows that (6.14) is equivalent to the policy evaluation problem (6.10) if we replace $\mathcal{P}$ with $\overline{\mathcal{P}}$. ∎

Proposition 6.3.2 provides a dual characterisation of the robust value iteration. On one hand, the robust value iteration determines the exact worst-case expected total reward over the rectangularised uncertainty set $\overline{\mathcal{P}}$. On the other hand, the robust value iteration calculates a lower bound on the worst-case expected total reward over the original uncertainty set $\mathcal{P}$. Hence, rectangularising the uncertainty set is equivalent to replacing the space of continuous reward to-go functions in the policy evaluation problem (6.10) with the subspace of constant functions.

We obtain a tighter lower bound on the worst-case expected total reward (6.10) if we replace the space of continuous reward to-go functions with the subspaces of affine or piecewise affine functions. We use the following result to formulate these approximations as tractable semidefinite optimisation problems.

Proposition 6.3.3 *For Ξ defined in (6.3b) and any fixed $S \in \mathbb{S}^q$, $s \in \mathbb{R}^q$ and $\sigma \in \mathbb{R}$, we have*

$$\exists\, \gamma \in \mathbb{R}^L_+ \;:\; \begin{bmatrix} \sigma & \frac{1}{2}s^\top \\ \frac{1}{2}s & S \end{bmatrix} - \sum_{l=1}^{L} \gamma_l \begin{bmatrix} \omega_l & \frac{1}{2}o_l^\top \\ \frac{1}{2}o_l & O_l \end{bmatrix} \succeq 0 \qquad \Longrightarrow \qquad \xi^\top S\,\xi + s^\top \xi + \sigma \geq 0 \quad \forall\, \xi \in \Xi. \tag{6.17}$$

Furthermore, the reversed implication holds if (C1) $L = 1$ or (C2) $S \succeq 0$.

Proof Implication (6.17) and the reversed implication under condition (C1) follow from the approximate and exact versions of the $\mathcal{S}$-Lemma, respectively (see for example Proposition 3.4 in [KWG09]).

Assume now that (C2) holds. We define $f(\xi) := \xi^\top S\,\xi + s^\top \xi + \sigma$ and $g_l(\xi) := -\xi^\top O_l\,\xi - o_l^\top \xi - \omega_l$, $l = 1, \ldots, L$. Since f and $g := (g_1, \ldots, g_L)$ are convex, Farkas' Theorem [Roc70] ensures that the system of inequalities

$$f(\xi) < 0, \quad g(\xi) < 0, \quad \xi \in \mathbb{R}^q \tag{6.18a}$$

has no solution if and only if there is a nonzero vector $(\kappa, \gamma) \in \mathbb{R}_+ \times \mathbb{R}^L_+$ such that

$$\kappa f(\xi) + \gamma^\top g(\xi) \geq 0 \quad \forall\, \xi \in \mathbb{R}^q. \tag{6.18b}$$

Since Ξ contains a Slater point $\overline{\xi}$ that satisfies $\overline{\xi}^\top O_l\,\overline{\xi} + o_l^\top \overline{\xi} + \omega = -g_l(\overline{\xi}) > 0$, $l = 1, \ldots, L$, convexity of g and continuity of f allows us to replace the second strict inequality in (6.18a) with a less or equal constraint. Hence, (6.18a) has no solution if and only if f is non-negative on $\Xi = \{\xi \in \mathbb{R}^q : g(\xi) \leq 0\}$, that is, if the right-hand side of (6.17) is satisfied. We now show that (6.18b) is equivalent to the left-hand side of (6.17). Assume that there is a nonzero vector $(\kappa, \gamma) \geq 0$ that satisfies (6.18b). Note that $\kappa \neq 0$ since otherwise, (6.18b) would not be satisfied by the Slater point $\overline{\xi}$. Hence, a suitable scaling of γ allows us to set $\kappa := 1$. For our choice of f and g, this implies that (6.18b) is equivalent to

$$\begin{bmatrix} 1 \\ \xi \end{bmatrix}^\top \left(\begin{bmatrix} \sigma & \frac{1}{2}s^\top \\ \frac{1}{2}s & S \end{bmatrix} - \sum_{l=1}^{L} \gamma_l \begin{bmatrix} \omega_l & \frac{1}{2}o_l^\top \\ \frac{1}{2}o_l & O_l \end{bmatrix} \right) \begin{bmatrix} 1 \\ \xi \end{bmatrix} \geq 0 \qquad \forall \xi \in \mathbb{R}^q. \tag{6.18b'}$$

Since the above inequality is homogeneous of degree 2 in $\left[1,\ \xi^\top\right]^\top$, it extends to the whole of $\mathbb{R}^{q+1}$. Hence, (6.18b') is equivalent to the left-hand side of (6.17). ∎

Proposition 6.3.3 allows us to bound the worst-case expected total reward (6.10) from below by the solution of a tractable semidefinite program.

Theorem 6.3.2 *Consider the following variant of the policy evaluation problem (6.10), which approximates the reward to-go function by an affine function,*

$$\sup_{\vartheta:\Xi \overset{a}{\mapsto} \mathbb{R}^S} \left\{ \inf_{\xi \in \Xi} \left\{ p_0^\top \vartheta(\xi) \right\} \, : \, \vartheta(\xi) \leq \widehat{r}(\xi) + \lambda \widehat{P}(\xi)\, \vartheta(\xi) \ \ \forall \xi \in \Xi \right\}, \tag{6.19}$$

as well as the semidefinite program

$$\begin{aligned}
&\underset{\tau, w, W, \gamma, \Gamma}{\text{maximise}} \quad \tau && \text{(6.20a)}\\
&\text{subject to} \quad \tau \in \mathbb{R}, \quad w \in \mathbb{R}^S, \quad W \in \mathbb{R}^{S \times q}, \quad \gamma \in \mathbb{R}^L_+, \quad \Gamma \in \mathbb{R}^{S \times L}_+\\
&\begin{bmatrix} p_0^\top w - \tau & \frac{1}{2} p_0^\top W \\ \frac{1}{2} W^\top p_0 & 0 \end{bmatrix} - \sum_{l=1}^L \gamma_l \begin{bmatrix} \omega_l & \frac{1}{2} o_l^\top \\ \frac{1}{2} o_l & O_l \end{bmatrix} \succeq 0, && \text{(6.20b)}\\
&\sum_{a \in \mathcal{A}} \pi(a|s) \begin{bmatrix} k_{sa}^\top (r_{sa} + \lambda w) & \frac{1}{2}\left(r_{sa}^\top K_{sa} + \lambda \left[k_{sa}^\top W + w^\top K_{sa}\right]\right) \\ \frac{1}{2}\left(K_{sa}^\top r_{sa} + \lambda \left[W^\top k_{sa} + K_{sa}^\top w\right]\right) & \lambda K_{sa}^\top W \end{bmatrix}\\
&\qquad - \begin{bmatrix} w_s & \frac{1}{2} W_{s\cdot}^\top \\ \frac{1}{2}\left(W_{s\cdot}^\top\right)^\top & 0 \end{bmatrix} - \sum_{l=1}^L \Gamma_{sl} \begin{bmatrix} \omega_l & \frac{1}{2} o_l^\top \\ \frac{1}{2} o_l & O_l \end{bmatrix} \succeq 0 \qquad \forall\, s \in \mathcal{S}, && \text{(6.20c)}
\end{aligned}$$

where $(r_{sa})_{s'} := r(s,a,s')$ for $(s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$. Let $(\tau^*, w^*, W^*, \gamma^*, \Gamma^*)$ denote an optimal solution to (6.20), and define $\vartheta^* : \Xi \overset{a}{\mapsto} \mathbb{R}^S$ through $\vartheta^*(\xi) := w^* + W^* \xi$. We have that:

(a) If $L = 1$, then (6.19) and (6.20) are equivalent *in the following sense: τ^* coincides with the supremum of (6.19), and ϑ^* is feasible and optimal in (6.19).*

(b) If $L > 1$, then (6.20) constitutes a conservative approximation *for (6.19): τ^* provides a lower bound on the supremum of (6.19), and ϑ^* is feasible in (6.19) and satisfies $\inf_{\xi \in \Xi} \left\{ p_0^\top \vartheta^*(\xi) \right\} = \tau^*$.*

Proof The approximate policy evaluation problem (6.19) can be written as

$$\sup_{\substack{w \in \mathbb{R}^S, \\ W \in \mathbb{R}^{S \times q}}} \left\{ \inf_{\xi \in \Xi} \left\{ p_0^\top (w + W\xi) \right\} \,:\, w + W\xi \leq \widehat{r}(\xi) + \lambda \widehat{P}(\xi)\,(w + W\xi) \quad \forall\, \xi \in \Xi \right\}. \qquad (6.21)$$

We first show that (6.21) is solvable. Since $p_0^\top (w + W\xi)$ is linear in (w, W) and continuous in ξ while Ξ is compact, $\inf_{\xi \in \Xi} \left\{ p_0^\top (w + W\xi) \right\}$ is a concave and therefore continuous function of (w, W). Likewise, the feasible region of (6.21) is closed because it results from the intersection of closed halfspaces parametrised by $\xi \in \Xi$. However, the feasible region of (6.21) is *not* bounded because any reward to-go function of the form (we, W) with $w \in \mathbb{R}_-$ and $W = 0$, constitutes a feasible solution. However, since $(w, W) = (0, 0)$ is feasible, we can append the constraint $w + W\xi \geq 0$ for all $\xi \in \Xi$ without changing the optimal value of (6.21). Moreover, all expected rewards $r(s, a, s')$ are bounded from above by $\overline{r} := \max_{s,a,s'} \{r(s, a, s')\}$. Therefore, Proposition 6.3.1 (c) implies that any feasible solution (w, W) for (6.21) satisfies $w + W\xi \leq \overline{r}\mathrm{e}/(1-\lambda)$ for all $\xi \in \Xi$.

Our results so far imply that any feasible solution (w, W) for (6.21) satisfies $0 \leq w + W\xi \leq \overline{r}\mathrm{e}/(1-\lambda)$ for all $\xi \in \Xi$. We now show that this implies boundedness of the feasible region for (w, W). The existence of a Slater point $\overline{\xi}$ with $\overline{\xi}^\top O_l \overline{\xi} + o_l^\top \xi + \omega_l > 0$ for all $l = 1, \ldots, L$ guarantees that there is an ϵ-neighbourhood of $\overline{\xi}$ that is contained in Ξ. Hence, W must be bounded because all points ξ in this neighbourhood satisfy $0 \leq w + W\xi \leq \overline{r}\mathrm{e}/(1-\lambda)$. As a consequence, w is bounded as well since $0 \leq w + W\overline{\xi} \leq \overline{r}\mathrm{e}/(1-\lambda)$. Thus, the feasible region of (6.21) is bounded, and Weierstrass' extreme value theorem is applicable. Therefore, (6.21) is solvable. If we furthermore replace $\widehat{P}$ and $\widehat{r}$ with their definitions from (6.7) and go over to an epigraph formulation, then we obtain

$$\begin{aligned}
&\underset{\tau,w,W}{\text{maximise}} && \tau && (6.22a)\\
&\text{subject to} && \tau \in \mathbb{R}, \quad w \in \mathbb{R}^S, \quad W \in \mathbb{R}^{S \times q} \\
& && \tau \leq p_0^\top (w + W\xi) \qquad \forall \xi \in \Xi && (6.22b)\\
& && w_s + W_{s\cdot}^\top \xi \leq \sum_{a \in \mathcal{A}} \pi(a|s) \left(k_{sa} + K_{sa}\xi\right)^\top \left(r_{sa} + \lambda \left[w + W\xi\right]\right) \qquad \forall \xi \in \Xi,\ s \in \mathcal{S}. && (6.22c)
\end{aligned}$$

Constraint (6.22b) is equivalent to constraint (6.20b) by Proposition 6.3.3 under condition (C2). Likewise, Proposition 6.3.3 guarantees that constraint (6.22c) is implied by constraint (6.20c). Moreover, if $L = 1$, condition (C1) of Proposition 6.3.3 is satisfied, and both constraints are equivalent. ∎

We can employ conic duality [AG03, LVBL98] to equivalently replace constraint (6.20b) with conic quadratic constraints. There does not seem to be a conic quadratic reformulation of constraint (6.20c), however.

Theorem 6.3.2 provides an exact (for $L = 1$) or conservative (for $L > 1$) reformulation for the approximate policy evaluation problem (6.19). Since (6.19) optimises only over affine approximations of the reward to-go function, Proposition 6.3.1 (c) implies that (6.19) provides a conservative approximation for the worst-case expected total reward (6.10). We will see below that both approximations are tight for s-rectangular uncertainty sets. First, however, we investigate the computational complexity of problem (6.20).

Corollary 6.3.2 *The semidefinite program (6.20) can be solved to any accuracy ϵ in polynomial time $\mathcal{O}\big((qS + LS)^{\frac{5}{2}}(q^2S + LS)\log\epsilon^{-1} + q^2AS^2\big)$.*

Proof The objective function and constraints of (6.20) can be constructed in time $\mathcal{O}\big(q^2AS^2 + q^2LS\big)$. Under mild assumptions, interior point methods can solve a semidefinite program

$$\min_{x\in\mathbb{R}^n}\left\{c^\top x \,:\, F_0 + \sum_{i=1}^{n} x_i F_i \succeq 0\right\},$$

where $F_i \in \mathbb{S}^m$ for $i = 0, \ldots, n$, to accuracy ϵ in time $\mathcal{O}\big(n^2 m^{\frac{5}{2}} \log\epsilon^{-1}\big)$, see [VB96]. Moreover, if all matrices F_i possess a block-diagonal structure with blocks $G_{ij} \in \mathbb{S}^{m_j}$, $j = 1, \ldots, J$ with $\sum_j m_j = m$, then the computational effort can be reduced to $\mathcal{O}\big(n^2 m^{\frac{1}{2}} \sum_j m_j^2\big)$. Problem (6.20) involves $\mathcal{O}(qS + LS)$ variables. By exploiting the block-diagonal structure of (6.20), constraint (6.20b) gives rise to a single block of dimension $(q + 1) \times (q + 1)$, constraint set (6.20c) leads to S blocks of dimension $(q + 1) \times (q + 1)$ each, and non-negativity of γ and Γ results in L and SL one-dimensional blocks, respectively. ∎

In Section 6.4 we discuss a method for constructing uncertainty sets from observation histories. Asymptotically, this method generates an uncertainty set Ξ that is described by a single quadratic inequality ($L = 1$), which means that problem (6.20) can be solved in time $\mathcal{O}\big(q^{\frac{9}{2}} S^{\frac{7}{2}} \log\epsilon^{-1} + q^2AS^2\big)$. Note that q does not exceed $S(S-1)A$, the affine dimension of the space $[\mathcal{M}(\mathcal{S})]^{S\times A}$, unless some components of ξ are perfectly correlated. If information about the structure of the transition kernel is available, however, q can be much smaller. Section 6.6

provides an example in which q remains constant as the problem size (measured in terms of S, the number of states) increases.

The semidefinite program (6.20) is based on two approximations. It is a conservative approximation for problem (6.19), which itself is a restriction of the policy evaluation problem (6.10) to affine reward to-go functions. We now show that both approximations are tight for s-rectangular uncertainty sets.

Proposition 6.3.4 *Let $(\tau^*, w^*, W^*, \gamma^*, \Gamma^*)$ denote an optimal solution to the semidefinite program (6.20), and define $\vartheta^* : \Xi \mapsto \mathbb{R}^S$ through $\vartheta^*(\xi) := w^* + W^*\xi$. If the uncertainty set $\mathcal{P}$ is s-rectangular, then the optimal value of the policy evaluation problem (6.10) is τ^*, and ϑ^* is feasible and optimal in (6.10).*

Proof We show that any constant reward to-go function that is feasible in the policy evaluation problem (6.10) can be extended to a feasible solution of the semidefinite program (6.20) with the same objective value. The assertion then follows from the optimality of constant reward to-go functions for s-rectangular uncertainty sets, see Theorem 6.3.1, and the fact that (6.20) bounds (6.10) from below, see Theorem 6.3.2.

Assume that $\vartheta : \Xi \mapsto \mathbb{R}^S$ with $\vartheta(\xi) = c$ for all $\xi \in \Xi$ satisfies the constraints of the policy evaluation problem (6.10). We show that there is a vector $\gamma \in \mathbb{R}^L_+$ and a matrix $\Gamma \in \mathbb{R}^{S\times L}_+$ such that $(\tau, w, W, \gamma, \Gamma)$ with $\tau := p_0^\top c$, $w := c$ and $W := 0$ satisfies the constraints of the semidefinite program (6.20). Since $\tau = \inf_{\xi\in\Xi} \left\{p_0^\top \vartheta(\xi)\right\}$, ϑ in (6.10) and $(\tau, w, W, \gamma, \Gamma)$ in (6.20) clearly attain equal objective values.

By the proof of Theorem 6.3.2, there is a vector $\gamma \in \mathbb{R}^L_+$ that satisfies constraint (6.20b) if and only if $\tau \leq p_0^\top (w + W\xi)$ for all $\xi \in \Xi$. Since $w + W\xi = c$ for all $\xi \in \Xi$ and $\tau = p_0^\top c$, such a vector γ indeed exists.

Let us now consider constraint set (6.20c). Since the constant reward to-go function $\vartheta(\xi) = c$ is feasible in the policy evaluation problem (6.10), we have for state $s \in \mathcal{S}$ that

$$c_s \leq \widehat{r}_s(\xi) + \lambda \widehat{P}_{s\cdot}^\top(\xi)\, c \qquad \forall\, \xi \in \Xi.$$

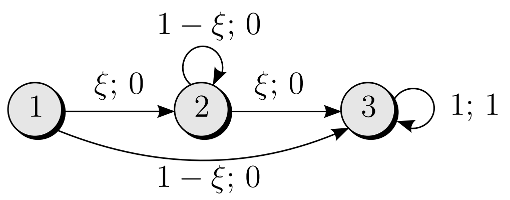


Figure 6.6: MDP with three states and one action. p_0 places unit probability mass on state 1. The same drawing conventions as in Figure 6.3 are used.

If we replace $\widehat{r}$ and $\widehat{P}$ with their definitions from (6.7), this is equivalent to

$$c_s \leq \sum_{a \in \mathcal{A}} \pi(a|s)(k_{sa} + K_{sa}\xi)^\top (r_{sa} + \lambda c) \qquad \forall\, \xi \in \Xi,$$

which is an instance of constraint (6.22c) where $w = c$ and $W = 0$. For this choice of (w, W), Proposition 6.3.3 under condition (C2) is applicable to constraint (6.22c). Hence, (6.22c) is satisfied if and only if there is $\Gamma_{s\cdot}^\top \in \mathbb{R}_+^{1\times L}$ that satisfies constraint (6.20c). Since (6.22c) is satisfied, we conclude that we can indeed find γ and Γ such that $(\tau, w, W, \gamma, \Gamma)$ satisfies the constraints of the semidefinite program (6.20). ∎

Propositions 6.3.2 and 6.3.4 show that the lower bound provided by the robust value iteration is dominated by the bound obtained from the semidefinite program (6.20). The following example highlights that the quality of these bounds can differ substantially.

Example 6.3.2 *Consider the robust infinite horizon MDP that is visualised in Figure 6.6. The uncertainty set $\mathcal{P}$ encompasses all transition kernels that correspond to parameter realisations $\xi \in [0, 1]$. This MDP can be assigned an uncertainty set of the form (6.3). For $\lambda := 0.9$, the worst-case expected total reward is $\lambda^2/(1-\lambda) = 8.1$ and is incurred under the transition kernel corresponding to $\xi = 1$. The solution of the semidefinite program (6.20) yields the (affine) approximate reward to-go function $\vartheta^*(\xi) = (6.5,\ 9\xi,\ 10)^\top$ and therefore provides a lower bound of 6.5. The unique solution to the fixed point equations $w^* = \phi(w^*)$, where ϕ is defined in (6.11), is $w^* = (0,\ 0,\ 1/[1-\lambda])$. Hence, the best constant reward to-go approximation yields a lower bound of zero. Since all expected rewards are non-negative, this is a trivial bound. Intuitively, the poor performance of the constant reward to-go function is due to the fact that it considers separate worst-case parameter realisations for states 1 ($\xi = 1$) and 2 ($\xi = 0$).*

Example 6.3.2 shows that the semidefinite program (6.20) generically provides a strict lower bound on the worst-case expected total reward if the uncertainty set is non-rectangular. In such cases, we would like to estimate the incurred approximation error. Note that we obtain an *upper* (*i.e.*, optimistic) bound on the worst-case expected total reward if we evaluate $p_0^\top v(\xi)$ for any single $\xi \in \Xi$. Let $\vartheta^*(\xi)$ denote an optimal affine approximation of the reward to-go function obtained from the semidefinite program (6.20). This ϑ^* can be used to obtain a suboptimal solution to $\arg\min \left\{ p_0^\top v(\xi) \,:\, \xi \in \Xi \right\}$ by solving $\arg\min \left\{ p_0^\top \vartheta^*(\xi) \,:\, \xi \in \Xi \right\}$, which is a convex optimisation problem. Let ξ^* denote an optimal solution to this problem. We obtain an upper bound on the worst-case expected total reward by evaluating

$$p_0^\top v(\xi^*) \;=\; p_0^\top \sum_{t=0}^{\infty} \left[\lambda \widehat{P}(\xi^*) \right]^t \widehat{r}(\xi^*) \;=\; p_0^\top \left[I - \lambda \widehat{P}(\xi^*) \right]^{-1} \widehat{r}(\xi^*), \tag{6.23}$$

where the last equality follows from the matrix inversion lemma, see Theorem C.2 in [Put94]. We can thus estimate the approximation error of the semidefinite program (6.20) by evaluating the difference between (6.23) and the optimal value of (6.20). If this difference is large, the affine approximation of the reward to-go function may be too crude. In this case, one could use modern decision rule techniques [BTGN09, GS09] to reduce the approximation error via piecewise affine approximations of the reward to-go function. Since the resulting generalisation requires no new ideas, we omit details for the sake of brevity.

Remark 6.3.3 (Finite Horizon MDPs) *Our results can be directly applied to finite horizon MDPs if we convert them to infinite horizon MDPs. To this end, we choose any discounting factor λ and multiply the rewards associated with transitions in period $t \in \mathcal{T}$ by λ^{-t}. Moreover, for every terminal state $s \in \mathcal{S}_T$, we introduce a deterministic transition to an auxiliary absorbing state and assign an action-independent expected reward of $\lambda^{-T} \mathbf{r}_s$. Note that in contrast to non-robust and rectangular MDPs, the approximate policy evaluation problem (6.20) does not decompose into separate subproblems for each time period $t \in \mathcal{T}$.*

We close with an example that illustrates the approximate policy evaluation problem (6.20).

Example 6.3.3 *Consider again the robust infinite horizon MDP defined in Example 6.3.2 and* *visualised in Figure 6.6. The state set of the MDP is $\mathcal{S} = \{1, 2, 3\}$, the set of admissible actions is $\mathcal{A} = \{1\}$, and the initial state distribution is given by $p_0 = \mathrm{e}_1$. The uncertainty set Ξ is specified by*

$$\Xi = \{\xi \in \mathbb{R} : 0 \leq \xi \leq 1\} \quad = \quad \left\{\xi \in \mathbb{R} : (\xi - 1/2)^2 \leq (1/2)^2\right\} \quad = \quad \left\{\xi \in \mathbb{R} : -\xi^2 + \xi \geq 0\right\},$$

that is, we have $L = 1$, $O_1 = -1$, $o_1 = 1$ and $\omega_1 = 0$. The transition probabilities are described by $p(\cdot|1, 1) = (0,\ \xi,\ 1 - \xi)^\top$, $p(\cdot|2, 1) = (0,\ 1 - \xi,\ \xi)^\top$ and $p(\cdot|3, 1) = \mathrm{e}_3$. In the notation of Section 6.2.1, we therefore have

$$(k_{11}, K_{11}) = (\mathrm{e}_3,\ \mathrm{e}_2 - \mathrm{e}_3), \quad (k_{21}, K_{21}) = (\mathrm{e}_2,\ \mathrm{e}_3 - \mathrm{e}_2) \quad \textit{and} \quad (k_{31}, K_{31}) = (\mathrm{e}_3,\ 0).$$

The reward is given by $r_{11} = r_{21} = 0$ and $r_{31} = \mathrm{e}_3$. For the discount factor $\lambda = 0.9$ and the policy $\pi(1|s) = 1$, $s \in \mathcal{S}$, the approximate policy evaluation problem (6.20) reads as follows.

$$\begin{array}{ll}
\underset{\tau, w, W, \gamma, \Gamma}{\text{minimise}} & \tau \\
\text{subject to} & \tau \in \mathbb{R}, \quad w \in \mathbb{R}^3, \quad W \in \mathbb{R}^3, \quad \gamma \in \mathbb{R}_+, \quad \Gamma \in \mathbb{R}^3_+ \\
& \begin{bmatrix} w_1 - \tau & \frac{1}{2}W_1 \\ \frac{1}{2}W_1 & 0 \end{bmatrix} - \gamma \begin{bmatrix} 0 & \frac{1}{2} \\ \frac{1}{2} & -1 \end{bmatrix} \succeq 0, \\
& \begin{bmatrix} \lambda w_3 & \frac{1}{2}\lambda(W_3 + w_2 - w_3) \\ \frac{1}{2}\lambda(W_3 + w_2 - w_3) & \lambda(W_2 - W_3) \end{bmatrix} - \begin{bmatrix} w_1 & \frac{1}{2}W_1 \\ \frac{1}{2}W_1 & 0 \end{bmatrix} - \Gamma_1 \begin{bmatrix} 0 & \frac{1}{2} \\ \frac{1}{2} & -1 \end{bmatrix} \succeq 0, \\
& \begin{bmatrix} \lambda w_2 & \frac{1}{2}\lambda(W_2 + w_3 - w_2) \\ \frac{1}{2}\lambda(W_2 + w_3 - w_2) & \lambda(W_3 - W_2) \end{bmatrix} - \begin{bmatrix} w_2 & \frac{1}{2}W_2 \\ \frac{1}{2}W_2 & 0 \end{bmatrix} - \Gamma_2 \begin{bmatrix} 0 & \frac{1}{2} \\ \frac{1}{2} & -1 \end{bmatrix} \succeq 0, \\
& \begin{bmatrix} 1 + \lambda w_3 & \frac{1}{2}\lambda W_3 \\ \frac{1}{2}\lambda W_3 & 0 \end{bmatrix} - \begin{bmatrix} w_3 & \frac{1}{2}W_3 \\ \frac{1}{2}W_3 & 0 \end{bmatrix} - \Gamma_3 \begin{bmatrix} 0 & \frac{1}{2} \\ \frac{1}{2} & -1 \end{bmatrix} \succeq 0.
\end{array}$$

The optimal solution to this problem satisfies $\tau^* = 6.5$, $w^* = (6.5, 0, 10)^\top$ and $W^* = (0, 9, 0)^\top$, see Example 6.3.2.

6.4 Robust Policy Improvement

In view of (6.10), we can formulate the policy improvement problem as

$$\sup_{\pi\in\Pi} \sup_{\vartheta:\Xi \overset{c}{\mapsto} \mathbb{R}^S} \left\{ \inf_{\xi\in\Xi} \left\{ p_0^\top \vartheta(\xi) \right\} \,:\, \vartheta(\xi) \le \widehat{r}(\pi;\xi) + \lambda\, \widehat{P}(\pi;\xi)\,\vartheta(\xi) \;\; \forall\, \xi \in \Xi \right\}. \qquad (6.24)$$

Since π is no longer fixed in this section, we make the dependence of v, $\widehat{P}$ and $\widehat{r}$ on π explicit. Section 6.3 shows that the policy evaluation problem can be solved efficiently if the uncertainty set $\mathcal{P}$ is s-rectangular. We now extend this result to the policy improvement problem.

Theorem 6.4.1 *For an s-rectangular uncertainty set $\mathcal{P}$, the policy improvement problem (6.24) is optimised by the policy $\pi^* \in \Pi$ and the constant reward to-go function $\vartheta^*(\xi) := w^*$, $\xi \in \Xi$, that are defined as follows. The vector $w^* \in \mathbb{R}^S$ is the unique fixed point of the contraction mapping φ defined through*

$$\varphi_s(w) := \max_{\pi\in\Pi} \left\{ \phi_s(\pi; w) \right\} \qquad \forall\, s \in \mathcal{S}, \qquad (6.25)$$

where ϕ is defined in (6.11). For each $s \in \mathcal{S}$, let $\pi^s \in \arg\max_{\pi\in\Pi} \left\{ \phi_s(\pi; w^*) \right\}$ denote a policy that attains the maximum on the right-hand side of (6.25) for $w = w^*$. Then $\pi^*(a|s) := \pi^s(a|s)$ for all $(s, a) \in \mathcal{S} \times \mathcal{A}$.

Proof In analogy to the proof of Theorem 6.3.1, we can rewrite the policy improvement problem (6.24) as

$$\max_{\pi\in\Pi} \max_{w\in\mathbb{R}^S} \left\{ p_0^\top w \,:\, w_s \le \widehat{r}_s(\pi;\xi^s) + \lambda\, \widehat{P}_{s\cdot}^\top(\pi;\xi^s)\, w \;\; \forall\, s \in \mathcal{S},\; \xi^1, \ldots, \xi^S \in \Xi \right\}.$$

By definition of ϕ, the S semi-infinite constraints in this problem are equivalent to the constraint $w \le \phi(\pi; w)$. If we interchange the order of the maximum operators, we can reexpress the problem as

$$\max_{w\in\mathbb{R}^S} \left\{ p_0^\top w \,:\, \exists\, \pi \in \Pi \;\text{ such that }\; w \le \phi(\pi; w) \right\}. \qquad (6.26)$$

Note that ϕ_s only depends on the components $\pi(\cdot|s)$ of π. Hence, we have $w^* = \phi(\pi^*; w^*)$, and π^* and w^* are feasible in (6.26). One can adapt the results in [Iye05, NG05] to show that φ is a contraction mapping. Since $w^* = \varphi(w^*)$ and every feasible solution w to (6.26) satisfies $w \leq \varphi(w)$, Theorem 6.2.2 in [Put94] therefore implies that $w^* \geq w$ for all feasible vectors w. By non-negativity of p_0, π^* and w^* must then be optimal in (6.26). The assertion now follows from the equivalence of (6.24) and (6.26). ∎

The fixed point w^* of the contraction mapping φ defined in (6.25) can be found via robust value iteration. Since the solution approach is essentially the same as in Section 6.3.1, we can keep ourselves brief in the following. The following result analyses the complexity of this method.

Corollary 6.4.1 *The fixed point w^* of the contraction mapping φ defined in (6.25) can be determined to any accuracy ϵ in time $\mathcal{O}\left((q + A + L)^{1/2}(qL + A)^3 S \log^2 \epsilon^{-1} + qAS^2 \log \epsilon^{-1}\right)$.*

Proof We apply the robust value iteration presented in Section 6.3.1 to the contraction mapping φ. To evaluate $\varphi_s(w)$, we solve the following semi-infinite optimisation problem:

$$\begin{aligned}
&\underset{\tau,\pi}{\text{maximise}} && \tau && \text{(6.27a)}\\
&\text{subject to} && \tau \in \mathbb{R}, \quad \pi \in \mathbb{R}^A \\
& && \tau \leq \sum_{a \in \mathcal{A}} \pi_a (k_{sa} + K_{sa}\xi)^\top (r_{sa} + \lambda w) \qquad \forall \xi \in \Xi, && \text{(6.27b)}\\
& && \pi \geq 0, \quad \mathrm{e}^\top \pi = 1. && \text{(6.27c)}
\end{aligned}$$

Second-order cone duality [AG03, LVBL98] allows us to replace the semi-infinite constraint (6.27b) with the following linear and conic quadratic constraints:

$$\exists\, Y \in \mathbb{R}^{q\times L},\, z \in \mathbb{R}^L,\, t \in \mathbb{R}^L \;:\qquad \tau - \sum_{a\in\mathcal{A}} \pi_a k_{sa}^\top (r_{sa} + \lambda w) \leq - \sum_{l=1}^{L} \left(\frac{1-\omega_l}{2} z_l + \frac{\omega_l + 1}{2} t_l \right) \tag{6.27b.1}$$

$$\sum_{l=1}^{L} \left(\Omega_l^\top Y_{\cdot l} - \frac{1}{2} o_l \left[t_l - z_l \right] \right) = \sum_{a\in\mathcal{A}} \pi_a K_{sa}^\top (r_{sa} + \lambda w) \tag{6.27b.2}$$

$$\left\| \begin{bmatrix} Y_{\cdot l} \\ z_l \end{bmatrix} \right\|_2 \leq t_l \quad \forall\, l = 1, \ldots, L. \tag{6.27b.3}$$

Here, Ω_l satisfies $\Omega_l^\top \Omega_l = -O_l$. The assertion follows if we evaluate $\varphi(w^i)$ at iteration i to an accuracy $\delta < \epsilon(1-\lambda)^2/8$ and stop when $\left\|w^{N+1} - w^N\right\|_\infty \leq \epsilon(1-\lambda)/4$ at some iteration N. ■

In analogy to Remark 6.3.2, we can solve the policy improvement problem for finite horizon MDPs via robust backward induction in time $\mathcal{O}\left((q+A+L)^{1/2}(qL+A)^3 S \log\epsilon^{-1} + qAS^2\right)$.

Since the policy improvement problem (6.24) contains the policy evaluation problem (6.10) as a special case, Theorem 6.2.1 implies that (6.24) is intractable for non-rectangular uncertainty sets. In analogy to Section 6.3, we can obtain a suboptimal solution to (6.24) by considering constant approximations of the reward to-go function. The following result is an immediate consequence of Proposition 6.3.2 and Theorem 6.4.1.

Corollary 6.4.2 *For a non-rectangular uncertainty set $\mathcal{P}$, consider the following variant of the policy improvement problem (6.24), which approximates the reward to-go function by a constant function.*

$$\sup_{\pi\in\Pi}\ \sup_{w\in\mathbb{R}^S} \left\{ p_0^\top w \;:\; w \leq \widehat{r}(\xi) + \lambda \widehat{P}(\xi)\, w \;\; \forall\, \xi \in \Xi \right\} \tag{6.28}$$

Problem (6.28) is optimised by the unique fixed point $w^* \in \mathbb{R}^S$ of the contraction mapping φ defined in (6.25).

In analogy to Proposition 6.3.2, the policy improvement problem (6.24) is equivalent to its approximation (6.28) if we replace $\mathcal{P}$ with $\times_s \mathcal{P}_s$. We can try to obtain better solutions to (6.24) over non-rectangular uncertainty sets by replacing the constant reward to-go approximations with affine or piecewise affine approximations. The associated optimisation problems are bilinear semidefinite programs and as such difficult to solve. Nevertheless, we can obtain a suboptimal solution with the following heuristic.

Algorithm 6.4.1. Sequential convex optimisation procedure.

1. *Initialisation.* Choose $\pi^1 \in \Pi$ (best policy found) and $i := 1$ (iteration counter).
2. *Policy Evaluation.* Solve the semidefinite program (6.20) for $\pi = \pi^i$ and store the τ-, w- and W-components of the solution in τ^i, w^i and W^i, respectively. Abort if $i > 1$ and $\tau^i = \tau^{i-1}$.
3. *Policy Improvement.* For each $s \in \mathcal{S}$, solve the semi-infinite optimisation problem

$$\begin{aligned}
&\underset{\sigma_s, \pi_s}{\text{maximise}} && \sigma_s && (6.29a)\\
&\text{subject to} && \sigma_s \in \mathbb{R}, \quad \pi_s \in \mathbb{R}^A \\
& && w_s + W_{s\cdot}^\top \xi + \sigma_s \leq \sum_{a \in \mathcal{A}} \pi_{sa} \left(k_{sa} + K_{sa}\xi\right)^\top \left(r_{sa} + \lambda\left[w + W\xi\right]\right) \qquad \forall \xi \in \Xi, && (6.29b)\\
& && \pi_s \geq 0, \quad \mathrm{e}^\top \pi_s = 1, && (6.29c)
\end{aligned}$$

where $(w, W) = (w^i, W^i)$. Set $\pi^{i+1}(a|s) := \pi^*_{sa}$ for all $(s, a) \in \mathcal{S} \times \mathcal{A}$, where π^*_s denotes the π_s- component of an optimal solution to (6.29) for state $s \in \mathcal{S}$. Set $i := i + 1$ and go back to Step 2.

Upon termination, the best policy found is stored in π^{i-1}, and τ^i is an estimate for the worst-case expected total reward of π^{i-1}. Depending on the number L of constraints that define Ξ, this estimate is exact (if $L = 1$) or a lower bound (if $L > 1$). We can equivalently reformulate (if $L = 1$) or conservatively approximate (if $L > 1$) the semi-infinite constraint (6.29b) with a semidefinite constraint. Since this reformulation parallels the proof of Theorem 6.3.2, we omit the details. Step 3 of the algorithm aims to increase the slack in the constraint (6.20c) of the policy evaluation problem solved in Step 2. One can show that if $\sigma_s > 0$ for some state $s \in \mathcal{S}$ that can be visited by the MDP, then Step 2 will lead to a better objective value in the next iteration. For $L = 1$, Algorithm 6.4.1 converges to a partial optimum of the policy improvement problem (6.24). We refer to Section 4.3 for a detailed convergence analysis and a numerical example of sequential convex optimisation.

6.5 Constructing Uncertainty Sets from Observation Histories

Assume that an observation history

$$(s_1, a_1, \ldots, s_n, a_n) \in (\mathcal{S} \times \mathcal{A})^n \tag{6.30}$$

of the MDP under some known stationary policy π^0 is available. We can use the observation (6.30) to construct an uncertainty set that contains the MDP's unknown true transition kernel P^0 with a probability of at least $1 - \beta$. The worst-case expected total reward of any policy π over this uncertainty set then provides a valid lower bound on the expected total reward of π under P^0 with a confidence of at least $1 - \beta$.

In the following, we first define the structural uncertainty set which incorporates all available a priori information about P^0. We then combine this structural information with the statistical information in the form of observation (6.30) to construct a confidence region for P^0. This confidence region will not be of the form (6.3). Section 6.5.3 therefore elaborates an approximate uncertainty set that is in line with the methods presented in Sections 6.3 and 6.4. We close with an asymptotic analysis of our approach.

6.5.1 Structural Uncertainty Set

Traditionally, uncertainty sets for the transition kernels of MDPs are constructed under the assumption that all transitions $(s, a, s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ are possible and that no a priori knowledge about the associated transition probabilities is available. In reality, however, one often has structural information about the MDP. For example, some transitions may be impossible, or certain functional relations between the transition probabilities may be known. We condense this kind of information into the *structural uncertainty set* $\mathcal{P}^0$, which captures all available a priori knowledge about the MDP. The use of structural information excludes irrelevant transition kernels and therefore leads to a smaller uncertainty set (and hence a tighter lower bound on the expected total reward). In Section 6.6, we will exemplify the benefits of this approach.

Formally, we assume that the structural uncertainty set $\mathcal{P}^0$ represents the affine image of a set Ξ^0, and that $\mathcal{P}^0$ and Ξ^0 satisfy our earlier definition (6.3) of $\mathcal{P}$ and Ξ. In the remainder of this chapter, we denote by ξ^0 the parameter vector associated with the unknown true transition kernel P^0 of the MDP, that is, $P^0_{sa} = p^{\xi^0}(\cdot|s, a)$ for all $(s, a) \in \mathcal{S} \times \mathcal{A}$. We require that

(A1) Ξ^0 contains the parameter vector ξ^0 in its interior: $\xi^0 \in \operatorname{int} \Xi^0$.

Assumption (A1) implies that all vanishing transition probabilities are known a priori. This requirement is standard in the literature on statistical inference for Markov chains [Bil61], and it is naturally satisfied if structural knowledge about the MDP is available. Otherwise, one may use the observation (6.30) to infer which transitions are possible. Indeed, it can be shown under mild assumptions that the probability to *not* observe a possible transition decreases exponentially with the length n of the observation [Bil61]. For a sufficiently long observation, we can therefore assign zero probability to unobserved transitions.

We illustrate the construction of the structural uncertainty set $\mathcal{P}^0$ in an important special case.

Example 6.5.1 *For every state-action pair $(s, a) \in \mathcal{S} \times \mathcal{A}$, let $\mathcal{S}_{sa} \subseteq \mathcal{S}$ denote the (nonempty) set of possible subsequent states if the MDP is in state s and action a is chosen. Assume* *that all sets $\mathcal{S}_{sa}$ are known, while no other structural information about the MDP's transition kernel is available. In the following, we define Ξ^0 and $p^\xi(\cdot|s,a)$ for this setting. For $(s,a) \in \mathcal{S} \times \mathcal{A}$, all but one of the probabilities corresponding to transitions (s,a,s'), $s' \in \mathcal{S}_{sa}$, can vary freely within the $(|\mathcal{S}_{sa}|-1)$-dimensional probability simplex, while the remaining transition probability is uniquely determined through the others. We therefore set the dimension of Ξ^0 to $q := \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}}(|\mathcal{S}_{sa}|-1)$. For each $(s,a) \in \mathcal{S} \times \mathcal{A}$, we define the set $\overline{\mathcal{S}}_{sa}$ of explicitly modelled transition probabilities through $\overline{\mathcal{S}}_{sa} := \mathcal{S}_{sa} \setminus \{\overline{s}_{sa}\}$, where $\overline{s}_{sa} \in \mathcal{S}_{sa}$ can be chosen freely. Let μ be a bijection that maps each triple (s,a,s'), $(s,a) \in \mathcal{S} \times \mathcal{A}$ and $s' \in \overline{\mathcal{S}}_{sa}$, to a component $\{1,\ldots,q\}$ of Ξ^0. We identify $\xi_{\mu(s,a,s')}$ with the probability of transition (s,a,s'). We define*

$$\Xi^0 := \left\{ \xi \in \mathbb{R}^q \,:\, \xi \geq 0,\; \sum_{s'\in\overline{\mathcal{S}}_{sa}} \xi_{\mu(s,a,s')} \leq 1 \;\; \forall\,(s,a) \in \mathcal{S}\times\mathcal{A} \right\} \tag{6.31}$$

and set $p^\xi(s'|s,a) := \xi_{\mu(s,a,s')}$ for $(s,a) \in \mathcal{S} \times \mathcal{A}$ and $s' \in \overline{\mathcal{S}}_{sa}$, as well as $p^\xi(\overline{s}_{sa}|s,a) := 1 - \sum_{s'\in\overline{\mathcal{S}}_{sa}} \xi_{\mu(s,a,s')}$ for $(s,a) \in \mathcal{S} \times \mathcal{A}$. The constraints in (6.31) ensure that all transition probabilities are non-negative.

6.5.2 Confidence Regions from Maximum Likelihood Estimation

In the following, we use the observation (6.30) to construct a confidence region for ξ^0. This confidence region will be centred around the maximum likelihood estimator associated with the observation (6.30), and its shape will be determined by the statistical properties of the likelihood difference between ξ^0 and its maximum likelihood estimator. To this end, we first calculate the log-likelihood function for the observation (6.30) and derive the corresponding maximum likelihood estimator. We then use existing statistical results for Markov chains (hereafter MCs) to construct a confidence region for ξ^0.

We remark that maximum likelihood estimation has recently been applied to construct confidence regions for the newsvendor problem [WGY09]. Our approach differs in two main aspects. Firstly, due to the nature of the newsvendor problem, the observation history in [WGY09] constitutes a collection of independent samples from a common distribution. Secondly, the newsvendor problem belongs to the class of single-stage stochastic programs, and the techniques developed in [WGY09] do not readily extend to MDPs.

The probability to observe the state-action sequence (6.30) under the policy π^0 and some transition kernel associated with $\xi \in \Xi^0$ is given by

$$p_0(s_1)\,\pi^0(a_n|s_n)\prod_{t=1}^{n-1}\left[\pi^0(a_t|s_t)\,p^{\xi}(s_{t+1}|s_t,a_t)\right]. \tag{6.32}$$

The log-likelihood function $\ell_n : \Xi^0 \mapsto \mathbb{R}\cup\{-\infty\}$ is given by the logarithm of (6.32), where we use the convention that $\log(0) := -\infty$. Thus, we set

$$\ell_n(\xi) := \sum_{t=1}^{n-1}\log\left[p^{\xi}(s_{t+1}|s_t,a_t)\right]+\zeta, \qquad \text{where} \qquad \zeta := \log\left[p_0(s_1)\right]+\sum_{t=1}^{n}\log\left[\pi^0(a_t|s_t)\right]. \tag{6.33}$$

Note that the remainder term ζ is finite and does not depend on ξ. Due to the monotonicity of the logarithmic transformation, the expressions (6.32) and (6.33) attain their maxima over Ξ^0 at the same points. Note also that we index the log-likelihood function with the length n of the observation (6.30). This will be useful later when we investigate its asymptotic behaviour as n tends to infinity.

The order of the transitions (s_t, a_t, s_{t+1}) in the observation (6.30) is irrelevant for the log-likelihood function (6.33). Hence, we can reexpress the log-likelihood function as

$$\ell_n(\xi) = \sum_{(s,a,s')\in N} n_{sas'}\log\left[p^{\xi}(s'|s,a)\right]+\zeta, \tag{6.33'}$$

where $n_{sas'}$ denotes the number of transitions from state $s \in \mathcal{S}$ to state $s' \in \mathcal{S}$ under action $a \in \mathcal{A}$ in (6.30), and $N := \{(s,a,s') \in \mathcal{S}\times\mathcal{A}\times\mathcal{S} : n_{sas'} > 0\}$ represents the set of observed transitions.

We obtain a maximum likelihood estimator ξ^n by maximising the concave log-likelihood function ℓ_n over Ξ^0. Since the observation (6.30) has strictly positive probability under the transition kernel associated with ξ^0, we conclude that $\ell_n(\xi^n) \geq \ell_n(\xi^0) > -\infty$. Note that the maximum likelihood estimator may not be unique if ℓ_n fails to be strictly concave.

Remark 6.5.1 (Analytical Solution) *Sometimes the maximum likelihood estimator can be calculated analytically. Consider, for instance, the log-likelihood function associated with Example 6.5.1.*

$$\ell_n(\xi) = \sum_{\substack{(s,a,s')\in N:\\ s'\in\overline{\mathcal{S}}_{sa}}} n_{sas'} \log\left[\xi_{\mu(s,a,s')}\right] + \sum_{(s,a,\overline{s}_{sa})\in N} n_{sa\overline{s}_{sa}} \log\left[1 - \sum_{s'\in\overline{\mathcal{S}}_{sa}} \xi_{\mu(s,a,s')}\right] + \zeta$$

The gradient of ℓ_n vanishes at ξ^n defined through $\xi^n_{\mu(s,a,s')} := n_{sas'}/\sum_{s''\in\mathcal{S}} n_{sas''}$ if $\sum_{s''\in\mathcal{S}} n_{sas''} > 0$ and $\xi^n_{\mu(s,a,s')} := 0$ otherwise. Since $\xi^n \in \Xi^0$, see (6.31) in Example 6.5.1, it constitutes a maximum likelihood estimator.

For $\xi \in \Xi^0$, the log-likelihood $\ell_n(\xi)$ describes the (logarithm of the) probability to observe the state-action sequence (6.30) under the transition kernel associated with ξ. For a sufficiently long observation, we therefore expect the log-likelihood $\ell_n(\xi^0)$ of the unknown true parameter vector ξ^0 to be 'not much smaller' than the log-likelihood $\ell_n(\xi^n)$ of the maximum likelihood estimator ξ^n. Guided by this intuition, we intersect the set Ξ^0 with a constraint that bounds this log-likelihood difference.

$$\Xi^0 \cap \{\xi \in \mathbb{R}^q \,:\, \ell_n(\xi) \geq \ell_n(\xi^n) - \delta\} \tag{6.34}$$

Here, $\delta \in \mathbb{R}_+$ determines the upper bound on the anticipated log-likelihood difference between ξ^0 and ξ^n. Expression (6.34) raises two issues. Firstly, it is not clear how δ should be chosen. Secondly, the intersection does not constitute a valid uncertainty set since it is not of the form (6.3b). In the following, we address the choice of δ. We postpone the discussion of the second issue to the next section.

Our choice of δ relies on statistical inference and requires two further assumptions:

(A2) The MC with state set $\mathcal{S}$ and transition kernel $\widehat{P}(\pi^0;\xi)$ is irreducible for some $\xi \in \Xi^0$, see (6.7a).

(A3) The matrix with rows $[K_{sa}]^\top_{s'\cdot}$ for $(s,a,s') \in \mathcal{S}\times\mathcal{A}\times\mathcal{S}$ with $\pi^0(a|s) > 0$ has rank $\kappa > 0$.

Remember that a finite MC with state set $\mathcal{S}$ is called *irreducible* if for any pair of states $s, s' \in \mathcal{S}$, there is a strictly positive probability that the MC visits state s' in the future if it is currently in state s. Assumption (A2) therefore guarantees that the MDP visits every state infinitely often as the observation length n tends to infinity. Assumption (A3) ensures that the historical policy π^0 chooses at least one state-action pair with unknown transition probabilities $p^{\xi^0}(\cdot|s,a)$. If this was not the case, then the observation (6.30) would not allow any inference about ξ^0, and the tightest possible uncertainty set for the unknown true transition kernel P^0 would be the structural uncertainty set $\mathcal{P}^0$.

We can now establish an asymptotic relation between ξ^n and ξ^0.

Theorem 6.5.1 *Under the assumptions (A1)–(A3), we have*

$$2\left[\ell_n(\xi^n) - \ell_n(\xi^0)\right] \underset{n\to\infty}{\longrightarrow} \chi^2_\kappa, \tag{6.35}$$

where '$\longrightarrow$' denotes convergence in distribution and χ^2_κ is a χ^2-distribution with κ degrees of freedom.

Remark 6.5.2 *A sequence of random variables X_i with cumulative distribution functions F_i, $i = 1, 2, \ldots$, is said to* converge in distribution *to a random variable X with cumulative distribution function F if* $\lim_{n\to\infty} F_n(x) = F(x)$ *at all points $x \in \mathbb{R}$ where F is continuous.*

Proof of Theorem 6.5.1 See Appendix C. ∎

Theorem 6.5.1 can be interpreted as follows. The observation (6.30) constitutes a random vector whose true distribution is determined by the expression (6.32) if we set $\xi = \xi^0$. Since ξ^0 is unknown, the distribution of the observation (6.30) is unknown as well. Similarly, the maximum likelihood estimator ξ^n depends on the observation (6.30) and is therefore a random vector with an unknown distribution. Theorem 6.5.1 shows, however, that the distribution of the random variable $2\left[\ell_n(\xi^n) - \ell_n(\xi^0)\right]$ is asymptotically known: it converges to a χ^2_κ distribution. Thus, under the assumptions (A1)–(A3), we obtain a $(1-\beta)$-confidence region for ξ^0 if we set δ in (6.34) to one half of the $(1-\beta)$-quantile of the χ^2_κ distribution.

$$\mathbb{P}\left(\xi^0 \in \Xi^0 \cap \{\xi \in \mathbb{R}^q \,:\, \ell_n(\xi) \geq \ell_n(\xi^n) - \delta\}\right) \;\geq\; 1-\beta$$

The support of the χ^2_κ distribution is unbounded above, and thus δ grows indefinitely if β goes to zero. For a fixed observation length n, the set (6.34) therefore reduces to Ξ^0 for $\beta \longrightarrow 0$.

Theorem 6.5.1 provides an asymptotic convergence result for robust *infinite* horizon MDPs. Robust *finite* horizon MDPs, on the other hand, are not directly amenable to an asymptotic analysis since they reach a terminal state after finitely many transitions. The most natural way to estimate the transition kernel of a finite horizon MDP is to assume that the MDP is 'restarted', that is, the same MDP is run several times. Theorem 6.5.1 can be applied to this situation as follows. We construct an infinite horizon MDP whose state space consists of the states of the finite horizon MDP, together with an auxiliary 'restarting' state τ. Apart from the transitions of the finite horizon MDP, the infinite horizon MDP contains deterministic transitions from all terminal states $s \in \mathcal{S}_T$ to τ, as well as transitions from τ to all initial states $s \in \mathcal{S}_1$ with action-independent transition probabilities $p_0(s)$. We do not specify a discount factor λ or one-step rewards r since they are irrelevant for Theorem 6.5.1. We interpret m observation histories $(s^i_1, a^i_1, \ldots, s^i_{T-1}, a^i_{T-1}, s^i_T)$, where $i = 1, \ldots, m$, of the finite horizon MDP as one observation

$$(s^1_1, a^1_1, \ldots, s^1_{T-1}, a^1_{T-1}, s^1_T, a^1_T; \;\; \ldots \;\; ; s^m_1, a^m_1, \ldots, s^m_{T-1}, a^m_{T-1}, s^m_T, a^m_T)$$

of the corresponding infinite horizon MDP. In this concatenated observation, the terminal actions $a^i_T \in \mathcal{A}$ may be chosen freely. We can now apply Theorem 6.5.1 to the constructed infinite horizon MDP if it satisfies the assumptions (A1)–(A3). This is the case if the finite horizon MDP satisfies the assumptions (A1) and (A3) and if each of its states can be reached from an initial state $s \in \mathcal{S}_1$ with $p_0(s) > 0$.

We close with a variant of Theorem 6.5.1 that relaxes the assumption (A2).

Remark 6.5.3 *Even if assumption (A2) is violated, the MDP will eventually enter a set of irreducible states $\overline{\mathcal{S}} \subseteq \mathcal{S}$ from which it cannot escape. If we remove from the observation (6.30) all state-action pairs $(s_1, a_1, \ldots, s_\tau, a_\tau)$ for which $s_t \notin \overline{\mathcal{S}}$, $t = 1, \ldots, \tau$, then Theorem 6.5.1 can be applied to the reduced MDP that only consists of the states in $\overline{\mathcal{S}}$.*

6.5.3 Quadratic Approximation

The confidence region for the unknown parameter vector ξ^0 in (6.34) is not consistent with the definition (6.3b) that underlies our computational techniques developed in Sections 6.3 and 6.4. We therefore approximate the left-hand side of the constraint $\ell_n(\xi) \geq \ell_n(\xi^n) - \delta$ in (6.34) by a second-order Taylor expansion around the maximum likelihood estimator ξ^n and set

$$\Xi^n := \Xi^0 \cap \{\xi \in \mathbb{R}^q \,:\, \varphi_n(\xi) \geq 0\}\,, \tag{6.36}$$

where

$$\varphi_n(\xi) := \left[\nabla_\xi \, \ell_n(\xi^n)\right]^\top (\xi - \xi^n) - \frac{1}{2}\,(\xi - \xi^n)^\top \left[\nabla^2_\xi \, \ell_n(\xi^n)\right] (\xi - \xi^n) + \delta \tag{6.37a}$$

with

$$\left[\nabla_\xi \, \ell_n(\xi^n)\right]^\top = \sum_{(s,a,s')\in N} \frac{n_{sas'}}{p^{\xi^n}(s'|s,a)} \left[K_{sa}\right]^\top_{s'\cdot} \tag{6.37b}$$

$$\text{and} \qquad \nabla^2_\xi \, \ell_n(\xi^n) = \sum_{(s,a,s')\in N} \frac{n_{sas'}}{[p^{\xi^n}(s'|s,a)]^2} \left(\left[K_{sa}\right]^\top_{s'\cdot}\right)^\top \left(\left[K_{sa}\right]^\top_{s'\cdot}\right). \tag{6.37c}$$

Note that the expressions in (6.37b) and (6.37c) are well-defined since $p^{\xi^n}(s'|s,a) > 0$ for all $(s, a, s') \in N$, see our discussion surrounding the log-likelihood function (6.33'). Moreover, Ξ^n is of the form (6.3b) since it emerges from the intersection of Ξ^0 with an ellipsoid. One can show that Ξ^n contains a Slater point whenever δ is strictly positive.

The set Ξ^n in (6.36) induces an uncertainty set of the form

$$\mathcal{P}^n := \left\{ P \in [\mathcal{M}(\mathcal{S})]^{S \times A} \ : \ \exists\, \xi \in \Xi^n \ \text{ such that } \ P_{sa} = p^{\xi}(\cdot|s,a) \ \ \forall\, (s,a) \in \mathcal{S} \times \mathcal{A} \right\}.$$

We now investigate the asymptotic properties of this uncertainty set as n tends to infinity. In Theorem 6.5.2 below we establish that $\mathcal{P}^n$ converges to the unknown true transition kernel P^0 of the MDP and analyse the speed of convergence. Afterwards, we show that the solutions of the robust policy evaluation and improvement problems converge to the solutions of the nominal policy evaluation and improvement problems under the unknown true transition kernel P^0. All subsequent convergence results rely on the following stronger version of assumption (A3).

(A3') The matrix with rows $[K_{sa}]_{s'\cdot}^{\top}$ for $(s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ with $\pi^0(a|s) > 0$ has full column rank.

Assumption (A3') stipulates that the mapping from ξ to the probabilities of all possible transitions under π^0 is injective. Indeed, if assumption (A3') is violated, then there are different parameter vectors $\xi, \xi' \in \Xi^0$ such that $p^{\xi}(s'|s,a) = p^{\xi'}(s'|s,a)$ for all possible transitions (s,a,s') under the data generating policy π^0. In this case, we cannot distinguish between ξ and ξ' based on the information provided by any observation of the type (6.30), and the uncertainty set $\mathcal{P}^n$ will not converge to a singleton as the observation length n tends to infinity.

In the following proposition, we analyse the Hausdorff distance between the two sets Ξ^n and $\{\xi^0\}$. Recall that the Hausdorff distance between two sets $X, Y \subseteq \mathbb{R}^q$ is defined as

$$d^{\mathrm{H}}(X,Y) := \max \left\{ \sup_{x \in X} \inf_{y \in Y} \|x-y\|_{\infty} \, , \, \sup_{y \in Y} \inf_{x \in X} \|x-y\|_{\infty} \right\}.$$

Theorem 6.5.2 *Under the assumptions (A1), (A2) and (A3'), we have*

$$\operatorname*{plim}_{n \longrightarrow \infty} \left(n^{\alpha} d^{\mathrm{H}} \left[\Xi^n, \{\xi^0\} \right] \right) = 0 \qquad \forall\, \alpha < 1/2, \tag{6.38}$$

where 'plim' denotes convergence in probability.

Remark 6.5.4 *Theorem 6.5.2 is equivalent to the statement that*

$$\lim_{n \longrightarrow \infty} \mathbb{P}\left(\max_{\xi \in \Xi^n} \left\| \xi - \xi^0 \right\|_\infty \leq \frac{\epsilon}{n^\alpha} \right) = 1$$

for every $\alpha < 1/2$ and $\epsilon > 0$.

Proof of Theorem 6.5.2 See Appendix D. ∎

We now show that under the assumptions of Theorem 6.5.2, the solution provided by the constant reward to-go approximation from Proposition 6.3.2 converges to the expected total reward $p_0^\top v(\xi^0)$ of policy π as n tends to infinity. Note that $\mathcal{P}^n$ constitutes a non-rectangular uncertainty set.

Proposition 6.5.1 *Let $\vartheta^n(\xi) = w^n$ be the constant reward to-go approximation described in Proposition 6.3.2 if we set $\Xi = \Xi^n$. Under the assumptions (A1), (A2) and (A3'), we have*

$$\operatorname*{plim}_{n \longrightarrow \infty} \left(n^\alpha \left| p_0^\top w^n - p_0^\top v(\pi; \xi^0) \right| \right) = 0 \qquad \forall\, \alpha < 1/2, \tag{6.39}$$

where $p_0^\top v(\pi; \xi^0)$ denotes the expected total reward under π and the unknown true transition kernel P^0.

Remark 6.5.5 *Proposition 6.5.1 is equivalent to the statement that for every $\alpha < 1/2$ and $\epsilon > 0$, we have*

$$\lim_{n \longrightarrow \infty} \mathbb{P}\left(\left| p_0^\top w^n - p_0^\top v(\pi; \xi^0) \right| \leq \frac{\epsilon}{n^\alpha} \right) = 1.$$

While Ξ^n is constructed from the observation (6.30) under the historical policy π^0, $p_0^\top w^n$ estimates the expected total reward of policy π. Note that π^0 and π can be different.

Proof of Proposition 6.5.1 Fix any $\alpha < 1/2$. By Theorem 6.5.2, we have

$$\operatorname*{plim}_{n \longrightarrow \infty} \left(n^\alpha \max_{\xi \in \Xi^n} \left\| \xi - \xi^0 \right\|_\infty \right) = 0. \tag{6.40}$$

The proof of Theorem 6.3.1 shows that for each w^n, $n \in \mathbb{N}$, there is $\xi^{n,1}, \ldots, \xi^{n,S} \in \Xi^n$ such that

$$w^n = \widehat{r}(\pi; \xi^{n,1}, \ldots, \xi^{n,S}) + \lambda \widehat{P}(\pi; \xi^{n,1}, \ldots, \xi^{n,S})\, w^n, \tag{6.41}$$

where for $\xi^1, \ldots, \xi^S \in \Xi^n$, the rectangular rewards $\widehat{r}(\pi; \xi^1, \ldots, \xi^S)$ and the rectangular transition kernel $\widehat{P}(\pi; \xi^1, \ldots, \xi^S)$ are defined through $\left[\widehat{r}(\pi; \xi^1, \ldots, \xi^S)\right]_s := \widehat{r}_s(\pi; \xi^s)$ and $\left[\widehat{P}(\pi; \xi^1, \ldots, \xi^S)\right]_{s\cdot}^\top := \widehat{P}_{s\cdot}^\top(\pi; \xi^s)$ for all $s \in \mathcal{S}$, respectively. Note that the existence of $\xi^{n,1}, \ldots, \xi^{n,S}$ does not depend on the structure of Ξ^n, see (6.14). By unrolling the recursion (6.41), we see that

$$w^n = v(\pi; \xi^{n,1}, \ldots, \xi^{n,S}) := \sum_{t=0}^{\infty} \left[\lambda \widehat{P}(\pi; \xi^{n,1}, \ldots, \xi^{n,S})\right]^t \widehat{r}(\pi; \xi^{n,1}, \ldots, \xi^{n,S}),$$

where for $\xi^1, \ldots, \xi^S \in \Xi^n$, $v(\pi; \xi^1, \ldots, \xi^S)$ represents a rectangular variant of the reward to-go function v. One can adapt the proof of Proposition 6.3.1 (a) to show that this rectangular reward to-go function is Lipschitz continuous on the compact set Ξ^0. Equation (6.40) therefore implies that

$$\operatorname*{plim}_{n \longrightarrow \infty} \left(n^\alpha \left\|v(\pi; \xi^{n,1}, \ldots, \xi^{n,S}) - v(\pi; \xi^0, \ldots, \xi^0)\right\|_\infty\right) = 0.$$

Equation (6.39) now follows from $w^n = v(\pi; \xi^{n,1}, \ldots, \xi^{n,S})$ and $v(\pi; \xi^0) = v(\pi; \xi^0, \ldots, \xi^0)$. ∎

Proposition 6.5.1 immediately extends to the affine reward to-go approximations obtained from the semidefinite program (6.20).

Corollary 6.5.1 *Let τ^n denote the optimal value of τ in the semidefinite program (6.20) with $\Xi = \Xi^n$. Under the assumptions (A1), (A2) and (A3'), we have*

$$\operatorname*{plim}_{n \longrightarrow \infty} \left(n^\alpha \left|\tau^n - p_0^\top v(\pi; \xi^0)\right|\right) = 0 \qquad \forall\, \alpha < 1/2.$$

Proof Fix $\alpha < 1/2$. Theorem 6.5.2 and the Lipschitz continuity of v, see Proposition 6.3.1 (a), imply that

$$\operatorname*{plim}_{n \longrightarrow \infty} \left(n^\alpha \max_{\xi \in \Xi^n} \left|p_0^\top v(\pi; \xi) - p_0^\top v(\pi; \xi^0)\right|\right) = 0.$$

Proposition 6.3.1 (c) and Theorem 6.3.2 ensure that $\tau^n \leq p_0^\top v(\pi; \xi)$ for all $\xi \in \Xi^n$, $n \in \mathbb{N}$. We conclude that

$$\operatorname*{plim}_{n \longrightarrow \infty} \left(n^\alpha \left[\tau^n - p_0^\top v(\pi; \xi^0) \right]^+ \right) = 0,$$

where $[x]^+ := \max\{x, 0\}$ for $x \in \mathbb{R}$. In a probabilistic sense, τ^n therefore underestimates $p_0^\top v(\pi; \xi^0)$. At the same time, Proposition 6.3.4 guarantees that $\tau^n \geq p_0^\top w^n$ for the vector w^n defined in Proposition 6.5.1. Hence, the assertion follows from the convergence of $p_0^\top w^n$, see Proposition 6.5.1. ∎

The above convergence results extend to the policy improvement problem discussed in Section 6.4. Since the derivation of the following result does not require any new ideas, we state it without a proof.

Proposition 6.5.2 *For $\Xi = \Xi^n$, let π^n denote an optimal policy determined by Algorithm 6.4.1 or the robust value iteration described in Corollary 6.4.2. Under the assumptions (A1), (A2) and (A3'), we have*

$$\operatorname*{plim}_{n \longrightarrow \infty} \left(n^\alpha \left| p_0^\top v(\pi^n; \xi^0) - \min_{\pi \in \Pi} \left\{ p_0^\top v(\pi; \xi^0) \right\} \right| \right) = 0 \qquad \forall\, \alpha < 1/2,$$

where the second term in the absolute value represents the expected total reward of the optimal policy under the MDP's unknown true transition kernel P^0.

Note that both the constant and the affine reward to-go approximations guarantee convergence to the nominal solutions of the policy evaluation and improvement problems as n tends to infinity. However, the next section will show that we can expect the affine approximations to convergence faster if the uncertainty set is non-rectangular.

We close this section with an example that illustrates the construction of uncertainty sets.

Example 6.5.2 *Consider again the robust infinite horizon MDP defined in Example 6.2.1. We interpret the uncertainty set constructed in that example as our structural uncertainty set $\mathcal{P}^0$,* *that is, we have*

$$\mathcal{P}^0 = \left\{ P \in [\mathcal{M}(\mathcal{S})]^{S\times A} \;:\; \exists \xi \in \Xi^0 \;\; \text{such that} \;\; P_{sa} = p^{\xi}(\cdot|s,a) \;\; \forall\, (s,a) \in \mathcal{S} \times \mathcal{A} \right\},$$

where

$$p^{\xi}(1|s,1) = \frac{1}{3} + \frac{\xi_1}{3}, \quad p^{\xi}(2|s,1) = \frac{1}{3} + \frac{\xi_2}{3}, \quad p^{\xi}(3|s,1) = \frac{1}{3} - \frac{\xi_1}{3} - \frac{\xi_2}{3} \quad \text{for } s \in \{1,2,3\}$$

and

$$\Xi^0 = \left\{ \xi \in \mathbb{R}^2 \;:\; \xi_1^2 + \xi_2^2 \le 1,\; \xi_1 \le \xi_2 \right\}.$$

We also remind the reader that we defined the affine mapping from Ξ^0 to $\mathcal{P}^0$ through

$$k_{s1} = \begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{bmatrix} \quad \text{and} \quad K_{s1} = \begin{bmatrix} \frac{1}{3} & 0 \\ 0 & \frac{1}{3} \\ -\frac{1}{3} & -\frac{1}{3} \end{bmatrix} \quad \text{for } s \in \{1,2,3\}.$$

The structural uncertainty set Ξ^0 is visualised in Figure 6.7.

Assume that the unknown true parameter vector is $\xi^0 = (1/4,\ 1/2)^\top \in \Xi^0$. Table 6.2 presents three observation histories of lengths 100, 1,000 *and* 10,000 *that were randomly generated under this choice of ξ^0. We obtain the maximum likelihood estimator ξ^{100} for the observation history of length 100 from the optimal solution to the following optimisation problem.*

$$\begin{aligned}
\underset{\xi}{\text{maximise}} \quad & (17+18+4)\log\left(\frac{1}{3}+\frac{\xi_1}{3}\right) + (19+30+4)\log\left(\frac{1}{3}+\frac{\xi_2}{3}\right) + && \text{(6.42a)}\\
& (3+5)\log\left(\frac{1}{3}-\frac{\xi_1}{3}-\frac{\xi_2}{3}\right) && \text{(6.42b)}\\
\text{subject to} \quad & \xi \in \mathbb{R}^2 && \text{(6.42c)}\\
& \xi_1^2 + \xi_2^2 \le 1, && \text{(6.42d)}\\
& \xi_1 \le \xi_2. && \text{(6.42e)}
\end{aligned}$$

Similar optimisation problems allow us to determine the maximum likelihood estimators $\xi^{1{,}000}$

	100 obs.			1,000 obs.			10,000 obs.		
	1	2	3	1	2	3	1	2	3
1	17	19	3	172	197	33	1,693	2,080	345
2	18	30	5	192	288	40	2,050	2,504	456
3	4	4	0	38	35	5	374	427	71

Table 6.2: Observation histories for an example MDP. Shown are the results of 100, 1,000 and 10,000 transitions of the MDP defined in Example 6.2.1. For each observation, the entry in row s and column s' denotes $n_{sas'}$ for the only action $a = 1$.

and $\xi^{10{,}000}$. The optimal solution to (6.42) is $\xi^{100} \approx (0.17,\ 0.59)^\top$. Similarly, the maximum likelihood estimators for the observation histories of length 1,000 *and* 10,000 *are $\xi^{1{,}000} \approx (0.21,\ 0.56)^\top$ and $\xi^{10{,}000} \approx (0.24,\ 0.50)^\top$, respectively. From these maximum likelihood estimators, we obtain the following estimates $p^{\xi^n}(\cdot|s,1)$ for the transition probabilities of the MDP:*

$$p^{\xi^{100}}(\cdot|s,1) \approx \begin{bmatrix} 0.39 \\ 0.53 \\ 0.08 \end{bmatrix}, \quad p^{\xi^{1,000}}(\cdot|s,1) \approx \begin{bmatrix} 0.40 \\ 0.52 \\ 0.08 \end{bmatrix} \quad \text{and} \quad p^{\xi^{10,000}}(\cdot|s,1) \approx \begin{bmatrix} 0.41 \\ 0.50 \\ 0.09 \end{bmatrix}.$$

We now construct the quadratic approximations (6.37) to the 99% *confidence regions Ξ^n. To this end, we set δ to half the value of $\chi^2_2 \approx 9.21$ and obtain for $n = 100$:*

$$\begin{aligned}\varphi_{100}(\xi) = &\left(\frac{17+18+4}{0.39} \begin{bmatrix} \frac{1}{3} \\ 0 \end{bmatrix} + \frac{19+30+4}{0.53} \begin{bmatrix} 0 \\ \frac{1}{3} \end{bmatrix} + \frac{3+5}{0.08} \begin{bmatrix} -\frac{1}{3} \\ -\frac{1}{3} \end{bmatrix} \right)^\top (\xi - \xi^n) + \\ &(\xi - \xi^n)^\top \left(\frac{17+18+4}{0.39^2} \begin{bmatrix} \frac{1}{9} & 0 \\ 0 & 0 \end{bmatrix} + \frac{19+30+4}{0.53^2} \begin{bmatrix} 0 & 0 \\ 0 & \frac{1}{9} \end{bmatrix} + \frac{3+5}{0.08^2} \begin{bmatrix} \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} \end{bmatrix} + \right) (\xi - \xi^n) + 9.21.\end{aligned}$$

Similar quadratic approximations can be obtained for $n = 1{,}000$ and $n = 10{,}000$. Figure 6.7 visualises the sets Ξ^{100}, $\Xi^{1{,}000}$ and $\Xi^{10{,}000}$ that result from these quadratic approximations.

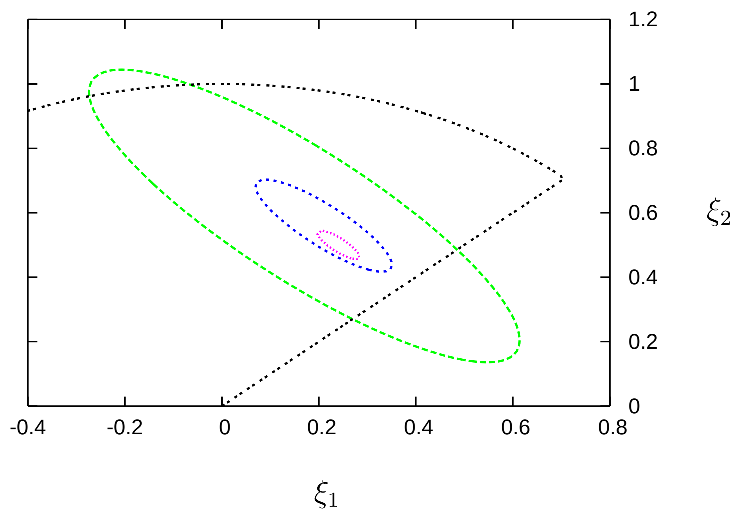


Figure 6.7: Confidence regions for an example MDP. Shown are Ξ^0 (dotted arc) and the three quadratic approximations used to construct Ξ^{100} (outer ellipsoid), $\Xi^{1,000}$ (second-largest ellipsoid) and $\Xi^{10,000}$ (innermost ellipsoid) from the observation histories in Table 6.2.

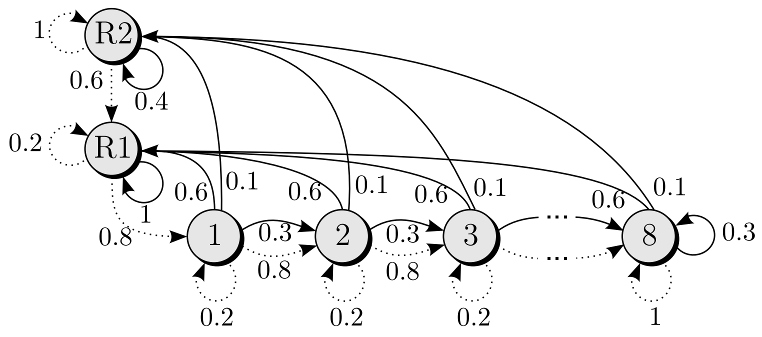


Figure 6.8: MDP for the machine replacement problem. Shown are the transition probabilities for the two actions 'do nothing' (dashed arcs) and 'repair' (solid arcs). The states 8, R1 and R2 pay an expected reward of -20, -2 and -10, respectively, while no reward is received in the other states. We use the same drawing conventions as in Figure 6.1.

6.6 Numerical Example

We apply the policy evaluation and improvement methods from Sections 6.3 and 6.4 to the machine replacement problem presented in [DM10]. The problem concerns a single machine whose condition is described by eight 'operative' states $1, \ldots, 8$ and two 'repair' states R1 and R2. At each time period, the decision maker receives an expected reward that depends on the machine's current state. The state in the subsequent time period is random and depends on both the current state and the chosen action ('do nothing' or 'repair'). The goal is to find a policy that maximises the expected total reward under the discount factor $\lambda = 0.8$. If all transition probabilities are known, we can model this problem as an MDP, see Figure 6.8. It is

n	RVI	SDP (LB)	SDP (UB)	$P^0 \in \mathcal{P}^n$?
500	-43.90	-30.37	-26.97	87%
1000	-32.34	-20.74	-18.81	92%
2500	-20.35	-15.36	-15.32	91%
500	-16.82	-14.95	-14.95	87%
1000	-15.20	-14.00	-13.99	88%
2500	-14.07	-13.31	-13.30	92%

Table 6.3: Policy evaluation results for 100 randomly generated observation histories of different observation length n. From left to right, the columns report the observation length, the average lower bound provided by the robust value iteration (RVI), the average lower and upper bounds obtained from the semidefinite program (6.20), and the percentage of instances in which P^0 is contained in $\mathcal{P}^n$. The first three rows were obtained without a priori knowledge, whereas the last three rows exploit the structural knowledge described in the text.

easy to transform this MDP into an equivalent one that satisfies the definitions in Section 6.1.

Consider the policy that chooses the actions 'do nothing' and 'repair' with probability 0.8 and 0.2, respectively, in each operative state $1, \ldots, 7$. In states 8 and R2, the policy always chooses the action 'repair', while the action 'do nothing' is chosen in state R1. The expected total reward of this policy is -12.34. Assume now that instead of the transition probabilities, we only have access to an observation history. We can use the structural uncertainty set $\mathcal{P}^0$ described in Example 6.5.1 and intersect it with a 90% confidence region for the unknown transition probabilities, see Section 6.5.3. The resulting uncertainty set is non-rectangular, and we can apply the robust value iteration from Proposition 6.3.2 or solve the semidefinite program (6.20) to obtain a lower bound on the worst-case expected total reward (6.2). The results for randomly generated observation histories are presented in the first part of Table 6.3. Note that the uncertainty set $\mathcal{P}^n$ contains the MDP's true transition kernel P^0 in about 90% of the observation histories. As the observation length n increases, the lower bounds obtained from both the robust value iteration and the semidefinite program (6.20) converge to the true expected total reward. However, the lower bounds provided by the semidefinite program are significantly tighter. From the optimality gaps we conclude that the semidefinite programming approximation performs well in this example.

The transition kernel in Figure 6.8 is highly structured. In particular, the probabilities as-

	RVI		SCO	
n	LB	nominal	LB	nominal
500	-12.35	-8.05	-10.45	-8.05
1000	-10.64	-8.00	-9.51	-8.00
2500	-9.50	-7.99	-8.99	-7.99

Table 6.4: Policy improvement results for 100 randomly generated observation histories of different observation length n. From left to right, the columns report the observation length, the average lower bound and nominal performance of the robust value iteration (RVI), and the average lower bound and nominal performance of the sequential convex optimisation procedure (SCO). In both cases, the nominal performance describes the expected total reward of the worst-case optimal policy under the unknown true transition kernel P^0.

sociated with the transitions emanating from state s under either action are identical for $s \in \{1, \ldots, 7\}$. We now assume that although these probabilities are unknown, they are known to be identical for $s \in \{1, \ldots, 7\}$. This additional information can be incorporated into the structural uncertainty set $\mathcal{P}^0$ to reduce the dimension of Ξ^0. The results are presented in the second part of Table 6.3. As the table shows, the incorporation of the additional structural information leads to significantly tighter bounds.

We now use the random observation histories to solve the robust policy improvement problem. The optimal policy for the unknown true transition kernel P^0 achieves an expected total reward of -7.98. Table 6.4 reports on the performance of the policies determined by the robust value iteration and the sequential convex optimisation algorithm from Section 6.4. Both methods perform well in this example. Nevertheless, the sequential convex optimisation algorithm provides tighter worst-case estimates. This is not surprising since the algorithm employs affine approximations of the reward to-go function.

We finally remark that we have considered variants of the MDP in Figure 6.8 with up to 1000 states. On average, the solution of the associated semidefinite program (6.20) required between 0.38 secs (10 states) and 228.92 secs (1000 states). Numerical results for the robust value iteration are reported in [Iye05, NG05].

6.7 Application to Temporal Networks

We now establish the connection between MDPs and temporal networks. Our discussion will be brief; further details can be found in [BR97, KA86, TSS06]. We consider a temporal network $G = (V, E)$ with tasks $V = \{1, \ldots, n\}$ and finish-start precedences $E \subseteq V \times V$. The tasks have random durations and give rise to uncertain cash flows at their start times. Our goal is to find a task start schedule that maximises the network's expected NPV. Apart from the precedence type, we are thus confronted with the same setting as in Chapter 3. Now, however, we allow for a much more expressive class of start time policies than in Chapter 3.

We start with the simplifying assumption that the task durations follow independent geometric distributions, that is, the probability that the duration of task $i \in V$ is $t \in \mathbb{N}$ is given by $(1 - \xi_i)^{t-1}\xi_i$, where the parameter vector $\xi \in (0, 1)^n$ is known. Later we will outline how the methods developed in this chapter can be used to maximise the network's expected NPV under generic task durations and unknown parameter vectors ξ.

At the beginning of each time period, the decision maker observes which network tasks have been completed during the previous time period. The decision maker then decides which of the yet unprocessed tasks should be started in the current time period. We can model this situation as an infinite horizon MDP with state set

$$\mathcal{S} = \{\text{not yet started}, \text{active}, \text{completed}\}^n, \tag{6.43}$$

that is, each state assigns one of the labels 'not yet started', 'active' and 'completed' to every network task. For state $s \in \mathcal{S}$, we denote the set of not yet started tasks, currently active tasks and completed tasks by $N(s)$, $A(s)$ and $C(s)$, respectively. Note that the state set $\mathcal{S}$ defined in (6.43) contains precedence-infeasible states. A state $s \in \mathcal{S}$ is *precedence-infeasible* if the temporal network contains a precedence $(i, j) \in E$ such that $j \in A(s) \cup C(s)$ but $i \notin C(s)$, that is, task j is processed or completed although not all of its predecessors have been completed. For the sake of efficiency, we should remove from $\mathcal{S}$ all precedence-infeasible states. The initial state σ and terminal state τ of the MDP satisfy $N(\sigma) = V$ and $C(\tau) = V$, respectively. The terminal state will be absorbing, that is, once the MDP enters state τ, it never leaves τ. The discount factor of the MDP is identical to the discount factor of the NPV maximisation problem.

In state s, the decision maker may start any subset of the tasks

$$\{j \in N(s) \, : \, i \in C(s) \;\; \forall \, (i,j) \in E\} \, ,$$

including the empty set. Each such subset corresponds to one feasible action. For action a, we denote by $T(a)$ the set of tasks that are started by a. Note that contrary to our definitions earlier in this chapter, the set of admissible actions is state-dependent. This state-dependency can be eliminated by penalising infeasible actions. Alternately, one can adapt the models in this chapter to allow for state-dependent action sets.

We now consider the transition probabilities $p(s'|s,a)$. We set $p(s'|s,a)$ to zero for all acausal transitions, that is, for all triples $(s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ that satisfy

1. $C(s) \cap [N(s') \cup A(s')] \neq \emptyset$, or
2. $[A(s) \cup T(a)] \cap N(s') \neq \emptyset$, or
3. $[N(s) \setminus T(a)] \cap [A(s') \cup C(s')] \neq \emptyset$.

In the first case, a task that is completed in state s would not be completed in state s' anymore. Similarly, in the second case, a task that is active would not have been started in the subsequent time period. In the third case, finally, a task that has not been started would become active or completed in the subsequent time period. For all causal transitions $(s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ that satisfy $A(s) \cup T(a) = \emptyset$, we set $p(s'|s,a) = 1$ if $s' = s$ and $p(s'|s,a) = 0$ otherwise. Indeed, if no tasks are being processed in state s and if the decision maker's action a does not start any tasks, then no tasks can be active in the subsequent state of the MDP either. For all other causal transitions $(s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$, we set

$$p(s'|s,a) = \prod_{i \in A(s) \cup T(a)} \left(\xi_i \, \mathbb{I}_{[i \in C(s')]} + (1 - \xi_i) \, \mathbb{I}_{[i \in A(s')]} \right) . \tag{6.44}$$

$$\begin{tikzcd}
 & & & 1c2a \arrow[r, "\emptyset"] \arrow[dd, "\{3\}"] & 1c2c \arrow[d, "\{3\}"] & & & \\
 & & & & 1c2c3a \arrow[dr, "\emptyset"] & & & \\
\sigma \arrow[r, "\{1\}"] & 1a \arrow[r, "\emptyset"] & 1c \arrow[uur, "\{2\}"] \arrow[r, "\{2,3\}"'] \arrow[ddr, "\{3\}"'] & 1c2a3a \arrow[ur, "\emptyset"] \arrow[dr, "\emptyset"'] & & 1c2c3c \arrow[r, "\{4\}"] & 1c2c3c4a \arrow[r, "\emptyset"] & \tau \\
 & & & & 1c2a3c \arrow[ur, "\emptyset"'] & & & \\
 & & & 1c3a \arrow[uu, "\{2\}"'] \arrow[r, "\emptyset"'] & 1c3c \arrow[u, "\{2\}"'] & & &
\end{tikzcd}$$

Figure 6.9: MDP generated from a temporal network. The nodes and arcs correspond to the states and transitions of the MDP, respectively. The label of node $s \in \mathcal{S}$ lists the tasks that are active (suffix 'a') or completed (suffix 'c') in s. The arc labels denote the actions that trigger the associated transitions. To improve readability, we omit the transition probabilities, the rewards, all 'transitive' transitions (*e.g.*, from state σ to state 1c under action $\{1\}$ or from state 1c to state 1c2c3c under action $\{2, 3\}$) and the self-loops associated with each node.

Here, the indicator function $\mathbb{I}$ satisfies $\mathbb{I}_x = 1$ if the logical expression x is true and $\mathbb{I}_x = 0$ otherwise. The probability of reaching state s', given that the current state is s and that action a is taken, is determined by the probabilities that each of the active tasks $i \in A(s) \cup T(a)$ remains active. Since the task durations are assumed to be independent, the transition probabilities between the states of the MDP result from the products of the individual probabilities for all active tasks. We set the reward of transition (s, a, s') to the sum of the expected cash flows associated with the network tasks that are started by action a, that is, $r(s, a, s') = \sum_{i \in T(a)} \mathbb{E}[\zeta_i]$, where $\mathbb{E}[\zeta_i]$ denotes the expected cash flow associated with the start time of task $i \in V$. Note that some of the rewards may be negative. Negative rewards can be avoided by adding a sufficiently large positive constant to all rewards, see Section 2.2.3.

Example 6.7.1 *Consider the temporal network $G = (V, E)$ with tasks $V = \{1, 2, 3, 4\}$ and precedences $E = \{(1, 2), (1, 3), (2, 4), (3, 4)\}$. Figure 6.9 visualises the MDP for this network.*

Our solution approaches for robust MDPs immediately apply to the MDPs generated from temporal networks. In this case, we assume that the parameter vector ξ defining the task durations is not known precisely. In some application areas, temporal networks are executed multiple times. For example, in production planning the same set of goods may be manufactured every week, and in microprocessor scheduling the same set of batch jobs may be run every night. In such cases, observation histories of the MDP may exist, and the techniques from Section 6.5 can be applied to construct an uncertainty set for the parameter vector ξ.

The transformation of temporal networks into MDPs is very general. We close with an outline of possible extensions.

1. **Abandonment Option.** By adding an action that transfers the MDP from any state to the terminal state τ with probability one, we can model the option to abort the processing of the temporal network. Abandonment options can be important in project management and are discussed in the literature on real options [DP94].
2. **Renewable Resources.** By restricting the admissible actions in each state, we can allow for renewable resources. In this case, action a is allowed in state s if and only if the joint resource consumption of the tasks in $A(s) \cup T(a)$ do not exceed specified quotas. In contrast to our solution approach in Chapter 3, renewable and doubly-constrained resources do not complicate the solution when using MDPs.
3. **Generic Task Durations.** So far we assumed that the task durations follow geometric distributions. We can allow for any discrete duration distribution if we split each task into subtasks. The kth subtask of task i corresponds to the processing required in the kth time period of i's execution. Contrary to our previous definition of admissible actions, the decision maker must start the next subtask of each active task in order to ensure that all tasks are executed in a non-preemptive manner.

6.8 Conclusion

We studied robust Markov decision processes (MDPs) in which the transition kernel is unknown. Traditionally, the policy evaluation and improvement problems for robust MDPs are solved in two steps. In the first step, one constructs a confidence region for the unknown parameters. Afterwards, one solves a robust optimisation problem over this confidence region.

We proposed a variant of this approach that differs in two important aspects. Firstly, existing methods rely on transition sampling to construct the confidence region for the MDP's transition kernel. In contrast, we use observation histories which are much easier to obtain in practise. Secondly, previous approaches solve an unduly conservative approximation of the aforementioned robust optimisation problem. As we pointed out in Section 6.2, this approximation can destroy vital characteristics of robust MDPs. We developed two novel approximations that retain these characteristics. Moreover, our approximations provide tighter bounds than the existing techniques. We applied our method to the machine replacement problem, and we discussed how our approach can be used to solve multi-stage NPV maximisation problems in temporal networks under uncertainty.

Chapter 7

Conclusion

Optimisation problems in temporal networks arise in a variety of application areas, such as the design of digital circuits and the scheduling of projects, production processes and microprocessors. Although it is widely accepted that these optimisation problems are affected by uncertainty, research on adequate models and solution approaches is still in its infancy. We believe that this is due to the network structure that is inherent to these optimisation problems. The network structure entails two crucial differences to ordinary optimisation problems under uncertainty: the times at which uncertain parameters are observed depend on the decision maker's actions, and the values of the uncertain parameters are not directly observable. It is these two differences that severely complicate the development of modelling and solution techniques for optimisation problems in temporal networks under uncertainty.

In this thesis we developed several techniques to model and solve optimisation problems in temporal networks under uncertainty. We considered problems that minimise the network's makespan (*i.e.*, the time required to complete all network tasks) and formulations that maximise the network's net present value (*i.e.*, the sum of discounted cash flows generated by the network tasks). We studied two-stage and multi-stage formulations, and we considered the optimisation of the expected outcome, quantiles of the outcome distribution (*i.e.*, the value-at-risk), and the worst-case outcome. All of these problems are $\mathcal{NP}$-hard, and there is little hope that one can ever reliably solve large instances of these problems to global optimality. We developed a number of strategies to address this challenge. In Chapter 3 we restricted our attention to the suboptimal class of target processing time policies. In Chapter 4 we employed central limit theorems to approximate the task path durations via normal distributions. In Chapter 5 we generated hierarchies of lower and upper bounds on the optimal objective values of robust resource allocation problems. In Chapter 6, finally, we used affine decision rules to approximate the reward to-go function in stochastic dynamic programming. We believe that the development of approximations that are theoretically sound and practically relevant will become a central topic in research on stochastic optimisation problems in temporal networks.

During our research, we identified several interesting avenues for future work. Firstly, with the exception of Chapter 6, all of our models assume absence of renewable and doubly-constrained resources. The literature on stochastic optimisation problems that incorporate such resources is very sparse. This is due to the fact that the early start policy discussed in Section 1.1 is no longer optimal for such problems, which severely complicates the scheduling problem. Nevertheless, we believe that it is worthwhile to investigate how the bounding approach developed in Chapter 5 could be extended to accommodate such resources. A second topic for future research is multi-stage optimisation in temporal networks. With the exception of Chapter 6, we restricted our attention to two-stage optimisation problems. Similar to renewable and doubly-constrained resources, computational difficulties have kept most researchers away from multi-stage formulations. The existing solution approaches for multi-stage problems have in common that they provide suboptimal solutions without bounding the incurred optimality gap. It would be desirable to extend the bounding approach presented in Chapter 5 to multi-stage problems. Finally, many application areas of temporal networks lag behind the recent developments in stochastic programming and robust optimisation theory. Ultimately, the value of theoretical work can only be appreciated by its success in practise. We therefore believe that it is imperative to further explore the applicability of our work in the various application areas of temporal networks.

Appendix A

Expected Cardinality of $\mathcal{P}$

For a fixed connectivity $\rho \in (0,1]$ and network size $n \in \mathbb{N}$, we construct a random temporal network $G = (V, E)$ with $V = \{1, \dots, n\}$ as follows. For each node $i \in V \setminus \{n\}$, we choose the number of immediate successors $\{1, \dots, \lceil \rho(n-i) \rceil\}$ uniformly at random. Afterwards, we choose the indices of the successor nodes from $\{i+1, \dots, n\}$, again uniformly at random. The resulting network is acyclic and has the unique sink n. We show that the expected number of paths in this network is exponential in n.

The probability that j is a successor of i, $i < j$, is

$$\frac{1}{\lceil \rho(n-i) \rceil} \sum_{j=1}^{\lceil \rho(n-i) \rceil} \frac{j}{n-i} = \frac{\lceil \rho(n-i) \rceil \left(\lceil \rho(n-i) \rceil + 1 \right)}{2 \lceil \rho(n-i) \rceil (n-i)} = \frac{\lceil \rho(n-i) \rceil + 1}{2(n-i)}.$$

Let X_i be the random variable that describes the number of paths from node i to node n. We have $\mathbb{E}(X_n) = 1$ and obtain

$$\mathbb{E}(X_i) = \frac{\lceil \rho(n-i) \rceil + 1}{2(n-i)} \sum_{j=i+1}^{n} \mathbb{E}(X_j) \quad \text{for } i < n.$$

In particular, $\mathbb{E}(X_{n-1}) = 1$. For $i < n$, we can express $\mathbb{E}(X_i)$ as follows.

$$\begin{aligned}\mathbb{E}(X_i) &= \frac{\lceil \rho(n-i)\rceil + 1}{2(n-i)} \left(1 + \frac{2(n-i-1)}{\lceil \rho(n-i-1)\rceil + 1}\right) \mathbb{E}(X_{i+1}) \\ &= \frac{\lceil \rho(n-i)\rceil + 1}{2(n-i)} \frac{\lceil \rho(n-i-1)\rceil + 1 + 2(n-i-1)}{\lceil \rho(n-i-1)\rceil + 1} \mathbb{E}(X_{i+1}).\end{aligned}$$

Partially unrolling the recursion, we obtain for $\mathbb{E}(X_1)$ and $m \in \{2, \dots, n\}$:

$$\begin{aligned}\mathbb{E}(X_1) &= \left(\prod_{i=1}^{m-1} \frac{\lceil \rho(n-i)\rceil + 1}{2(n-i)} \frac{\lceil \rho(n-i-1)\rceil + 1 + 2(n-i-1)}{\lceil \rho(n-i-1)\rceil + 1}\right) \mathbb{E}(X_m) \\ &= \frac{\lceil \rho(n-1)\rceil + 1}{\lceil \rho(n-m)\rceil + 1} \left(\prod_{i=1}^{m-1} \frac{\lceil \rho(n-i-1)\rceil + 1 + 2(n-i-1)}{2(n-i)}\right) \mathbb{E}(X_m) \\ &= \frac{\lceil \rho(n-1)\rceil + 1}{\lceil \rho(n-m)\rceil + 1} \left(\prod_{i=1}^{m-1} \left[1 + \frac{\lceil \rho(n-i-1)\rceil - 1}{2(n-i)}\right]\right) \mathbb{E}(X_m).\end{aligned}$$

Let us investigate the term $(\lceil \rho(n-i-1)\rceil - 1)/(2[n-i])$. We show that for a specific choice of m, this term is greater than or equal to some $\delta > 0$. Note that

$$\frac{\lceil \rho(n-i-1)\rceil - 1}{2(n-i)} \geq \frac{\rho(n-i-1) - 1}{2(n-i)} = \frac{\rho(n-i) - \rho - 1}{2(n-i)} = \frac{\rho}{2} - \frac{\rho + 1}{2(n-i)}.$$

Assume that $n \geq 2/\rho + 4$. Then the last expression is greater than or equal to $\rho/4$, a strictly positive number, for all $i \leq \overline{m} := n - \lceil (2\rho + 2)/\rho \rceil$. We obtain:

$$\begin{aligned}\mathbb{E}(X_1) &= \frac{\lceil \rho(n-1)\rceil + 1}{\lceil \rho(n-\overline{m})\rceil + 1} \left(\prod_{i=1}^{\overline{m}-1} \left(1 + \frac{\lceil \rho(n-i-1)\rceil - 1}{2(n-i)}\right)\right) \mathbb{E}(X_{\overline{m}}) \\ &\geq \frac{\lceil \rho(n-1)\rceil + 1}{\lceil \rho(n-\overline{m})\rceil + 1} \prod_{i=1}^{\overline{m}-1} \left(1 + \frac{\lceil \rho(n-i-1)\rceil - 1}{2(n-i)}\right) \\ &\geq \frac{\lceil \rho(n-1)\rceil + 1}{\lceil \rho(n-\overline{m})\rceil + 1} \prod_{i=1}^{\overline{m}-1} \left(1 + \frac{\rho}{4}\right) \\ &= \frac{\lceil \rho(n-1)\rceil + 1}{\lceil \rho(n-\overline{m})\rceil + 1} \left(1 + \frac{\rho}{4}\right)^{\overline{m}-1} \in \Omega\left(n(1+\rho/4)^n\right),\end{aligned}$$

where $\Omega(\cdot)$ denotes the asymptotic lower bound in Bachmann-Landau notation. Since the expected number of paths from node 1 to node n is already exponential, the expected number of all paths in network G is exponential, too.

Appendix B

Saddle Point Condition for s-Rectangular Uncertainty Sets

Proposition B.1 *For an infinite horizon MDP with an s-rectangular uncertainty set $\mathcal{P}$, we have*

$$\sup_{\pi\in\Pi}\ \inf_{P\in\mathcal{P}} \mathbb{E}^{P,\pi}\left[\sum_{t=0}^{\infty}\lambda^t r(s_t,a_t,s_{t+1}) \;\middle|\; s_0\sim p_0\right] \;=\; \inf_{P\in\mathcal{P}}\ \sup_{\pi\in\Pi} \mathbb{E}^{P,\pi}\left[\sum_{t=0}^{\infty}\lambda^t r(s_t,a_t,s_{t+1}) \;\middle|\; s_0\sim p_0\right]. \tag{B.1}$$

Proof It follows from the proof of Theorem 6.4.1 that the left-hand side of (B.1) is equivalent to

$$\max_{w\in\mathbb{R}^S}\left\{p_0^\top w \;:\; w_s \leq \max_{\pi\in\Pi}\ \min_{\xi^s\in\Xi}\left\{\widehat{r}_s(\pi;\xi^s)+\lambda\widehat{P}_{s\cdot}^\top(\pi;\xi^s)\,w\right\}\quad \forall\, s\in\mathcal{S}\right\}.$$

The constraints in this problem are equivalent to $w\leq\varphi(w)$, see (6.25). Since φ is a contraction mapping, see Theorem 6.4.1, non-negativity of p_0 and Theorem 6.2.2 in [Put94] allow us to reexpress the problem as

$$\min_{w\in\mathbb{R}^S}\left\{p_0^\top w \;:\; w_s \geq \max_{\pi\in\Pi}\ \min_{\xi^s\in\Xi}\left\{\widehat{r}_s(\pi;\xi^s)+\lambda\widehat{P}_{s\cdot}^\top(\pi;\xi^s)\,w\right\}\quad \forall\, s\in\mathcal{S}\right\}.$$

The max-min expressions in the constraints satisfy the conditions of Corollary 37.3.2 in [Roc70]. Hence, we can interchange the order of the operators in the constraints to obtain the following reformulation.

$$\min_{w \in \mathbb{R}^S} \left\{ p_0^\top w \;:\; w_s \geq \min_{\xi^s \in \Xi} \; \max_{\pi \in \Pi} \left\{ \widehat{r}_s(\pi; \xi^s) + \lambda \widehat{P}_{s\cdot}^\top(\pi; \xi^s)\, w \right\} \;\; \forall\, s \in \mathcal{S} \right\}.$$

The uncertainty set $\mathcal{P}$ is s-rectangular, and the sth constraint only depends on the components $\pi(\cdot|s)$ of π. Hence, similar transformations as in Theorems 6.3.1 and 6.4.1 yield the following reformulation.

$$\min_{w \in \mathbb{R}^S} \; \min_{\xi \in \Xi} \left\{ p_0^\top w \;:\; w_s \geq \widehat{r}_s(\pi; \xi) + \lambda \widehat{P}_{s\cdot}^\top(\pi; \xi)\, w \;\; \forall\, s \in \mathcal{S},\; \pi \in \Pi \right\}. \tag{B.2}$$

Since p_0 is non-negative, Theorems 6.1.1 and 6.2.2 in [Put94] imply that for a given $\xi \in \Xi$, the optimal solution w satisfies $w = \max_{\pi \in \Pi} \{v(\pi; \xi)\}$. The equivalence of (B.2) and the right-hand side of (B.1) now follows from the property (6.6) of the reward to-go function v. ∎

Appendix C

Proof of Theorem 6.5.1

The proof of Theorem 6.5.1 relies on the Theorems 2.1, 2.2 and 5.1 in [Bil61], which establish asymptotic properties of maximum likelihood estimators of ordinary MCs. To keep the thesis self-contained, we summarise these results in Theorem C.1.

Theorem C.1 *Consider a finite MC with state set $\mathcal{X} = \{1, \ldots, X\}$ and transition probabilities $p_{xy}(\theta)$, $x, y \in \mathcal{X}$, that depend on an unknown parameter vector θ ranging over an open set $\Theta \subseteq \mathbb{R}^U$. Assume that the following conditions are satisfied:*

(C1) *Each function p_{xy} has continuous partial derivatives of third order throughout Θ.*

(C2) *The set-valued mapping $D(\theta) := \{(x,y) \in \mathcal{X} \times \mathcal{X} : p_{xy}(\theta) > 0\}$ is constant, that is, there is a set $D \subseteq \mathcal{X} \times \mathcal{X}$ such that $D(\theta) = D$ for all $\theta \in \Theta$.*

(C3) *The Jacobian matrix of the transition kernel $\left(p_{xy}(\theta)\right)_{x,y}$ has rank U throughout Θ.*

(C4) *For each $\theta \in \Theta$, the MC is irreducible.*

Let $(x_1, \ldots, x_m)$ denote an observation of the MC under its true transition kernel $p_{xy}(\theta^0)$, where $\theta^0 \in \Theta$, and let m_{xy} denote the number of observations of transition $(x,y) \in \mathcal{X} \times \mathcal{X}$. For the sequence of functions $f_m(\theta) := \sum_{(x,y)\in D} m_{xy} \log\left[p_{xy}(\theta)\right]$, Θ contains a sequence of random *vectors $\overline{\theta}^m$ that satisfy*

$$2\left[f_m(\overline{\theta}^m) - f_m(\theta^0)\right] \quad \underset{m\to\infty}{\longrightarrow} \quad \chi^2_U, \tag{C.1a}$$

$$m^{1/2}\left(\overline{\theta}^m - \theta^0\right) \quad \underset{m\to\infty}{\longrightarrow} \quad \mathcal{N}(0,\Gamma). \tag{C.1b}$$

Here, $\mathcal{N}(0,\Gamma)$ is a multivariate normal distribution with zero mean and finite covariance matrix $\Gamma \succ 0$. Moreover, $\overline{\theta}^m$ is a strict local maximiser of f_m with probability going to one as m tends to infinity.

In order to apply Theorem C.1 to MDPs, we interpret the state-action sequence (6.30) as an observation history of an ordinary MC. Theorem 6.5.1 then follows from (C.1a). To simplify the exposition, we prove Theorem 6.5.1 first under assumption (A3') on page 209. At the end of this section, we extend our proof to hold under the weaker assumption (A3).

We interpret the state-action sequence (6.30) as an observation of n states of an MC with states

$$\mathcal{X} := \left\{(s,a) \in \mathcal{S} \times \mathcal{A} \,:\, \pi^0(a|s) > 0\right\}. \tag{C.2a}$$

The MC is in state $(s,a) \in \mathcal{X}$ whenever the underlying MDP is in state s and the decision maker chooses action a. Note that we omit state-action pairs $(s,a) \in \mathcal{S} \times \mathcal{A}$ with $\pi^0(a|s) = 0$ in (C.2a). As we will see, this is a necessary (but not sufficient) condition for the MC to be irreducible, see condition (C4) of Theorem C.1. By construction, the MC starts in state $(s,a) \in \mathcal{X}$ with probability $p_0(s)\,\pi^0(a|s)$, and it moves from state $(s,a) \in \mathcal{X}$ to state $(s',a') \in \mathcal{X}$ with probability $p^{\xi^0}(s'|s,a)\,\pi^0(a'|s')$, where ξ^0 is the unknown true parameter of the underlying MDP. Since the historical policy π^0 is stationary, the MC indeed satisfies the Markov property.

We can establish the following relationship between the MC and the MDP.

$$\Theta := \operatorname{int} \Xi^0 \tag{C.2b}$$

$$\text{and} \qquad p_{xy}(\theta) := p^{\theta}(s'|s,a)\,\pi^0(a'|s') \qquad \text{for } \theta \in \Theta \text{ and } x = (s,a),\ y = (s',a') \in \mathcal{X}. \tag{C.2c}$$

By assumption (A1), we have $\xi^0 \in \operatorname{int} \Xi^0$. Hence, Θ indeed contains the unknown true parameter vector $\theta^0 := \xi^0$ of the MC as required by Theorem C.1.

We now show that the MC defined through (C.2) satisfies the conditions of Theorem C.1.

Lemma C.1 *If the MDP satisfies assumptions (A2) and (A3'), then the MC defined through (C.2) satisfies the conditions (C1)–(C4) of Theorem C.1.*

Proof Condition (C1) is satisfied since p_{xy} is affine in θ for all $x, y \in \mathcal{X}$, see definitions (C.2c) and (6.3).

As for condition (C2), the definitions (C.2a) and (C.2c) imply that

$$D(\theta) = \left\{ (x, y) \in \mathcal{X} \times \mathcal{X} \,:\, p^{\theta}(s'|s,a) > 0 \;\text{ for } x = (s,a) \text{ and } y = (s',a') \right\}.$$

We recall that $p^{\theta}(\cdot|s,a) = k_{sa} + K_{sa}\theta$. We claim that for any $\theta \in \Theta$, the set $D(\theta)$ equals

$$D := \left\{ (x, y) \in \mathcal{X} \times \mathcal{X} \,:\, [k_{sa}\; K_{sa}]_{s'\cdot}^{\top} \neq 0 \;\text{ for } x = (s,a) \text{ and } y = (s',a') \right\}.$$

By construction, $D(\theta) \subseteq D$ for all $\theta \in \Theta$. It remains to show that $D \subseteq D(\theta)$ for all $\theta \in \Theta$. Assume to the contrary that $[k_{sa}\; K_{sa}]_{s'\cdot}^{\top} \neq 0$ but $p^{\theta}(s'|s,a) = 0$ for $x = (s,a)$, $y = (s',a') \in \mathcal{X}$ and $\theta \in \Theta$. Since Θ is an open set, there is a neighbourhood of θ that is contained in Θ, and all points θ' in this neighbourhood have to satisfy $p^{\theta'}(s'|s,a) \geq 0$. Since $p^{\theta}(s'|s,a) = 0$, this implies that $[K_{sa}]_{s'\cdot}^{\top} = 0$, and hence $[k_{sa}]_{s'} = 0$ as well. This contradicts our assumption that $[k_{sa}\; K_{sa}]_{s'\cdot}^{\top} \neq 0$. We therefore conclude that $p^{\theta}(s'|s,a) > 0$ for all $\theta \in \Theta$, that is, $D \subseteq D(\theta)$ for all $\theta \in \Theta$.

We now consider condition (C3). The Jacobian $J(\theta) \in \mathbb{R}^{|\mathcal{X}|^2 \times U}$ of the MC's transition kernel is defined through $J_{xy,u} := \partial p_{xy}(\theta)/\partial \theta_u$ for $x, y \in \mathcal{X}$ and $u = 1, \ldots, U$. For $x = (s,a)$, $y = (s',a') \in \mathcal{X}$, we have $\partial p_{xy}(\theta)/\partial \theta_u = \pi^0(a'|s')\, [K_{sa}]_{s'u}$. Thus, assumption (A3') ensures that $J(\theta)$ has rank U.

In view of condition (C4), we note that the irreducibility of a finite MC only depends on the structure of the set of transitions with strictly positive probability; the actual probabilities are irrelevant. However, the proof of condition (C2) implies that for all state pairs $(x, y) \in \mathcal{X} \times \mathcal{X}$, either $p_{xy}(\theta) > 0$ for all $\theta \in \Theta$ or $p_{xy}(\theta) = 0$ for all $\theta \in \Theta$. Hence, the set of transitions with strictly positive probability does not depend on θ, and the MC defined through (C.2) is irreducible for *all* $\theta \in \Theta$ if and only if it is irreducible for *some* $\theta \in \Theta$. Condition (C4) therefore follows from assumption (A2). ∎

We can now apply Theorem C.1 to the MC defined through (C.2). This allows us to prove Theorem 6.5.1 under the stronger assumption (A3').

Proof of Theorem 6.5.1 Under assumption (A3') the assumptions of Lemma C.1 are satisfied, and we can apply Theorem C.1 to the MC defined through (C.2). Hence, we know that Θ contains a sequence $\overline{\theta}^n$ that satisfies (C.1a), and each $\overline{\theta}^n$ constitutes a strict local maximiser of f_n with probability going to one as n tends to infinity. By definition (C.2c) of p, every function f_n is concave, which implies that $\overline{\theta}^n$ is indeed the unique global maximiser of f_n with probability going to one as n tends to infinity.

Let m_{xy} denote the number of observations of transition $(x, y) \in \mathcal{X} \times \mathcal{X}$ in (6.30). We additionally set $m_{xy} := 0$ for $(x, y) \in (\mathcal{S} \times \mathcal{A})^2 \setminus (\mathcal{X} \times \mathcal{X})$. For any $\theta \in \Theta$, we have

$$\begin{aligned}\ell_n(\theta) &= \sum_{(s,a,s') \in N} n_{sas'} \log\left[p^\theta(s'|s,a)\right] + \zeta \;=\; \sum_{\substack{x=(s,a)\in\mathcal{X},\\ y=(s',a')\in\mathcal{X}:\\ m_{xy}>0}} m_{xy} \log\left[p^\theta(s'|s,a)\right] + \zeta \\ &= \sum_{\substack{x,y\in\mathcal{X}:\\ m_{xy}>0}} m_{xy} \log\left[p_{xy}(\theta)\right] + \psi \;=\; \sum_{(x,y)\in D} m_{xy} \log\left[p_{xy}(\theta)\right] + \psi = f_n(\theta) + \psi, \qquad \text{(C.3)}\end{aligned}$$

where $\psi := \log\left[p_0(s_1)\right] + \log\left[\pi^0(a_1|s_1)\right]$. The first equality follows from the definition of ℓ_n in (6.33'). The second equality holds because $n_{sas'} = \sum_{a'\in\mathcal{A}} m_{(s,a),(s',a')}$ and $m_{(s,a),(s',a')} = 0$ if $\pi^0(a|s) = 0$ or $\pi^0(a'|s') = 0$. The third equality follows from the definition (C.2c) of p and our choice of ψ. As for the fourth equality, note that all $x, y \in \mathcal{X}$ with $m_{xy} > 0$ satisfy $p_{xy}(\theta^0) > 0$ for $\theta^0 = \xi^0$. Lemma C.1 therefore ensures that $(x, y) \in D(\theta^0) = D$. The last equality follows from the definition of f_n in Theorem C.1.

From (C.3) and the fact that $\theta^0 = \xi^0$ we conclude that $l_n(\xi^0) = f_n(\theta^0) + \psi$. Moreover, (C.3) implies that $\overline{\theta}^n$ defined in Theorem C.1 represents the unique global maximiser of ℓ_n with probability going to one as n tends to infinity. The assertion of Theorem 6.5.1 now follows from (C.1a). ∎

Remark C.1 *Throughout this section, we replaced assumption (A3) with the stronger assumption (A3') from page 209. Under assumption (A3), the Jacobian of the MC's transition kernel may violate condition (C3) of Theorem C.1. We circumvent this problem by decomposing the affine mapping p in (C.2c) into the composition of a linear surjection, followed by an affine injection. If we replace Θ with the image of* $\operatorname{int} \Xi^0$ *under the surjection and p with the injection, all conditions of Theorem C.1 remain satisfied.*

Appendix D

Proof of Theorem 6.5.2

We first investigate the convergence behaviour of the sequence φ_n of quadratic functions defined in (6.37a). To this end, Lemma D.1 investigates the asymptotic properties of the observation frequencies $n_{sas'}$, while Lemma D.2 investigates ξ^n, $\nabla_\xi \ell_n(\xi^n)$ and $\nabla^2_\xi \ell_n(\xi^n)$. These auxiliary results will then allow us to establish the convergence of the sequence of confidence regions Ξ^n defined in (6.36).

We recall that the *expected return time* of a state s in an MC is defined as the expected number of transitions between two successive visits of state s. We extend this definition to MDPs by defining the expected return time of state s under policy π as the expected return time of s in the MC defined through the state set $\mathcal{S}$ and the transition kernel (6.7a) with $\xi = \xi^0$.

Lemma D.1 *Under the assumptions (A1) and (A2), we have*

$$\frac{n_{sas'}}{n} \underset{n\to\infty}{\longrightarrow} \frac{\pi^0(a|s)\, p^{\xi^0}(s'|s,a)}{\mu_s} \qquad \text{almost surely for all } (s,a,s') \in \mathcal{S}\times\mathcal{A}\times\mathcal{S}, \tag{D.1}$$

where $\mu_s \in [1,\infty)$ denotes the expected return time of state $s \in \mathcal{S}$ under policy π^0.

Proof We first show that the expected return times μ_s are finite. To this end, let $\mathrm{MC}_{\mathcal{S}}(\pi;\xi)$ denote the MC defined through the state set $\mathcal{S}$ and the transition kernel (6.7a). Due to assumption (A2), $\mathrm{MC}_{\mathcal{S}}(\pi^0;\xi)$ is irreducible for some $\xi \in \Xi^0$. By a similar argument as in the proof of Lemma C.1, we may conclude that $\mathrm{MC}_{\mathcal{S}}(\pi^0;\xi)$ is indeed irreducible for all $\xi \in \operatorname{int}\Xi^0$. Assumption (A1) then guarantees that $\mathrm{MC}_{\mathcal{S}}(\pi^0;\xi^0)$ is irreducible, which implies that its expected return times μ_s are finite.

In view of equation (D.1), let n_s and n_{sa} denote the numbers of occurrences of state $s \in \mathcal{S}$ and state-action pair $(s,a) \in \mathcal{S} \times \mathcal{A}$ in the observation (6.30), respectively. As usual, $n_{sas'}$ denotes the number of occurrences of the state-action sequence $(s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$, and n represents the observation length. Note that the random variables n_s, n_{sa} and $n_{sas'}$ depend on n. If $\pi^0(a|s) = 0$, then $n_{sas'} = 0$, and (D.1) is trivially satisfied. We therefore assume that $\pi^0(a|s) > 0$. We show that

$$\text{(A)}\ \frac{n_s}{n} \underset{n\to\infty}{\longrightarrow} \frac{1}{\mu_s}\ \text{a.s.,}\quad \text{(B)}\ \frac{n_{sa}}{n_s} \underset{n\to\infty}{\longrightarrow} \pi^0(a|s)\ \text{a.s., and}\quad \text{(C)}\ \frac{n_{sas'}}{n_{sa}} \underset{n\to\infty}{\longrightarrow} p^{\xi^0}(s'|s,a)\ \text{a.s.,}$$

where 'a.s.' abbreviates 'almost surely'. Statements (A) and (B) imply that n_s and n_{sa} become nonzero a.s. as n tends to infinity, and therefore the identity $n_{sas'}/n = (n_{sas'}/n_{sa})(n_{sa}/n_s)(n_s/n)$ holds a.s. as n tends to infinity. The assertion of this lemma then follows from the continuous mapping theorem [Bil95].

As for claim (A), note that n_s represents the number of visits of $\mathrm{MC}_{\mathcal{S}}(\pi^0;\xi^0)$ to state $s \in \mathcal{S}$. Since $\mathrm{MC}_{\mathcal{S}}(\pi^0;\xi^0)$ is irreducible, the ergodic theorem ensures that $n_s/n \longrightarrow 1/\mu_s$ a.s. as n tends to infinity [Bil95].

In order to prove claims (B) and (C), we introduce a new MC denoted as $\mathrm{MC}_{\mathcal{S}\mathcal{A}}$. By construction, $\mathrm{MC}_{\mathcal{S}\mathcal{A}}$ is in state $s \in \mathcal{S}$ whenever the underlying MDP is in state s and the decision maker has not yet chosen any action, while $\mathrm{MC}_{\mathcal{S}\mathcal{A}}$ is in state $(s,a) \in \mathcal{S} \times \mathcal{A}$ whenever the MDP is in state s and the decision maker has chosen action a (but before the MDP moves to a new state s'). We can interpret the state-action sequence (6.30) as an observation of $2n$ states of $\mathrm{MC}_{\mathcal{S}\mathcal{A}}$, where $\mathrm{MC}_{\mathcal{S}\mathcal{A}}$ starts in state s_1, then moves to state (s_1,a_1), after which it enters state s_2 and so on. Formally, we define $\mathrm{MC}_{\mathcal{S}\mathcal{A}}$ through the state set $\mathcal{S} \cup (\mathcal{S} \times \mathcal{A})$ and the transition probabilities

$$p_{xy} = \begin{cases} \pi^0(a|s) & \text{if } x = s \in \mathcal{S} \text{ and } y = (s,a) \in \mathcal{S} \times \mathcal{A}, \\ p^{\xi^0}(s'|s,a) & \text{if } x = (s,a) \in S \times \mathcal{A} \text{ and } y = s' \in \mathcal{S}, \\ 0 & \text{otherwise.} \end{cases}$$

To prove claim (B), fix $(s,a) \in \mathcal{S} \times \mathcal{A}$ and let X_i be a random binary variable that adopts the value 1 if and only if $\mathrm{MC}_{\mathcal{SA}}$ moves to state (s,a) after the ith visit of state s. By the strong Markov property, the random variables X_i are independent and identically distributed with expected value $\pi^0(a|s)$ [Bil95]. Thus, the strong law of large numbers implies that $\sum_{i=1}^m X_i/m \longrightarrow \pi^0(a|s)$ a.s. as m tends to infinity. According to claim (A), $n_s \longrightarrow \infty$ a.s. as n tends to infinity. Hence, we obtain that $\sum_{i=1}^{n_s} X_i/n_s \longrightarrow \pi^0(a|s)$ a.s. as n tends to infinity. Claim (B) then follows from the fact that $n_{sa} = \sum_{i=1}^{n_s} X_i$.

The proof of claim (C) widely parallels the above argumentation for claim (B). ∎

Lemma D.2 *Under the assumptions (A1), (A2) and (A3'), observation (6.30) satisfies*

$$\lim_{n \longrightarrow \infty} \mathbb{P}\big(\nabla_\xi \ell_n(\xi^n) = 0\big) = 1, \tag{D.2a}$$

$$\operatorname*{plim}_{n \longrightarrow \infty} \left(n^\alpha \left\|\xi^n - \xi^0\right\|\right) = 0 \qquad \forall\, \alpha < 1/2, \tag{D.2b}$$

$$\operatorname*{plim}_{n \longrightarrow \infty} \left(\left\|\frac{1}{n}\left[\nabla^2_\xi \ell_n(\xi^n)\right] - \Sigma\right\|\right) = 0, \tag{D.2c}$$

where $\nabla_\xi \ell_n(\xi^n)$ and $\nabla^2_\xi \ell_n(\xi^n)$ are defined in (6.37b) and (6.37c), respectively, and

$$\Sigma := \sum_{(s,a,s') \in N_0} \frac{\pi^0(a|s)}{\mu_s\, p^{\xi^0}(s'|s,a)} \left([K_{sa}]_{s'\cdot}^\top\right)^\top \left([K_{sa}]_{s'\cdot}^\top\right), \tag{D.2d}$$

where $N_0 := \Big\{(s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S} \,:\, [k_{sa}\; K_{sa}]_{s'\cdot}^\top \neq 0\Big\}$. Moreover, the matrix Σ is positive definite.

Proof The proof of Theorem 6.5.1 shows that the unique global maximiser ξ^n of ℓ_n is an element of $\operatorname{int}\Xi^0$ with probability going to one as n tends to infinity. This proves (D.2a).

In view of (D.2b), consider any sequence X_n of random variables. One can show that if $n^\alpha X_n$ converges in distribution, then $n^\beta X_n$ converges to zero in probability for all $\beta < \alpha$. Thus, (D.2b) follows from (C.1b).

Let us now consider (D.2c). We can replace the set N in the summation index of $\nabla^2_\xi \ell_n(\xi^n)$ in (6.37c) with the set N_0 used in (D.2d). Indeed, $N \subseteq N_0$ holds because $n_{sas'} > 0$ implies that $p^{\xi^0}(s'|s,a) > 0$ and therefore $[k_{sa}\ K_{sa}]^\top_{s'\cdot} \neq 0$. Likewise, the numerator in (6.37c) vanishes for each index $(s,a,s') \in N_0 \setminus N$. Equation (D.2c) now follows from Lemma D.1, (D.2b) and the continuous mapping theorem.

It is clear that Σ is positive semidefinite. Also, $x^\top \Sigma x = 0$ if and only if $[K_{sa}]^\top_{s'\cdot} x = 0$ for all $(s,a,s') \in N_0$ with $\pi^0(a|s) > 0$. Assumption (A3') implies that this is the case if and only if $x = 0$. Thus, the matrix Σ has full rank and is therefore positive definite. ∎

We can now prove Theorem 6.5.2.

Proof of Theorem 6.5.2 Let $\mathbb{B}$ denote the closed unit ball centred at the origin of $\mathbb{R}^q$. For fixed $\alpha < 1/2$, (6.38) is satisfied if and only if for all $\epsilon, \gamma > 0$, there is $m \in \mathbb{N}$ such that for all $n \geq m$,

$$\mathbb{P}\left(n^\alpha\left(\Xi^n - \xi^0\right) \subseteq \epsilon\mathbb{B}\right) \geq 1 - \gamma, \tag{D.3}$$

where operations on sets are understood in the Minkowski sense. We define $\phi_n(x) := \varphi_n\left(n^{-\alpha}x + \xi^0\right)$. According to the definition (6.36) of Ξ^n, we have

$$n^\alpha\left(\Xi^n - \xi^0\right) \subseteq \{x \in \mathbb{R}^q : \phi_n(x) \geq 0\}$$

because the set on the right-hand side ignores the constraints from Ξ^0. Hence, (D.3) holds if

$$\mathbb{P}\left(\{x \in \mathbb{R}^q : \phi_n(x) \geq 0\} \subseteq \epsilon\mathbb{B}\right) \geq 1 - \gamma,$$

which is equivalent to

$$\mathbb{P}\left(\{x \in \mathbb{R}^q : \phi_n(x) < 0\} \supseteq \epsilon\mathbb{B}^c\right) \geq 1 - \gamma, \tag{D.4}$$

where $\epsilon\mathbb{B}^c := \mathbb{R}^q \setminus \epsilon\mathbb{B}$ denotes the complement of $\epsilon\mathbb{B}$. We prove (D.4) in two steps. We first show that ϕ_n is negative on $\epsilon\mathbb{B}^c \cap 2\epsilon\mathbb{B}$. Afterwards, we show that $\phi_n(0) > \phi_n(x)$ for all $x \in \epsilon\mathbb{B}^c \cap 2\epsilon\mathbb{B}$. Since ϕ_n is concave, this implies that ϕ_n remains negative on $\mathbb{R}^q \setminus 2\epsilon\mathbb{B}$ with high probability. We can then conclude that ϕ_n is negative on the whole set $\epsilon\mathbb{B}^c$ with high probability, which proves (D.4).

Using the definition (6.37a) of φ_n and Lemma D.2, one can show that

$$\operatorname*{plim}_{n \longrightarrow \infty} \left(\sup_{x \in 2\epsilon\mathbb{B}} \left| n^{2\alpha-1}\phi_n(x) - \frac{1}{2} x^\top \Sigma\, x \right| \right) = 0, \tag{D.5}$$

where Σ is defined in (D.2d). In a probabilistic sense, $n^{2\alpha-1}\phi_n(x)$ therefore converges uniformly to $x^\top \Sigma\, x/2$ over $2\epsilon\mathbb{B}$. Since Σ is positive definite, see Lemma D.2, there is $\nu > 0$ such that $\Sigma \succeq \nu I$, that is, $x^\top \Sigma\, x \geq \nu \|x\|^2$ for all x. We thus obtain that for any $\eta > 0$, we can choose m such that for all $n \geq m$,

$$\mathbb{P}\left(n^{2\alpha-1}\phi_n(0) \geq -\eta,\;\; n^{2\alpha-1}\phi_n(x) \leq -\frac{\nu}{2}\epsilon^2 + \eta \;\; \forall\, x \in \epsilon\mathbb{B}^c \cap 2\epsilon\mathbb{B} \right) \geq 1 - \gamma.$$

For $\eta < \nu\epsilon^2/4$ this is equivalent to

$$\mathbb{P}\left(\phi_n(0) > \phi_n(x),\;\; \{x \in \mathbb{R}^q \,:\, \phi_n(x) < 0\} \supseteq \epsilon\mathbb{B}^c \cap 2\epsilon\mathbb{B} \right) \geq 1 - \gamma.$$

According to our previous discussion, this proves equation (D.4) and the assertion of the theorem. ∎

Bibliography

[ADEH99] P. Artzner, F. Delbaen, J.-M. Eber, and D. Heath. Coherent measures of risk. *Mathematical Finance*, 9(3):203–228, 1999.

[AG03] F. Alizadeh and D. Goldfarb. Second-order cone programming. *Mathematical Programming*, 95(1):3–51, 2003.

[AK89] V. G. Adlakha and V. G. Kulkarni. A classified bibliography of research on stochastic pert networks: 1966–1987. *INFOR*, 27(3):272–296, 1989.

[Ave01] I. Averbakh. On the complexity of a class of combinatorial optimization problems with uncertainty. *Mathematical Programming*, 90(2):263–272, 2001.

[AZ07] A. Atamtürk and M. Zhang. Two-stage robust network flow and design under demand uncertainty. *Operations Research*, 55(4):662–673, 2007.

[BDM$^+$99] P. Brucker, A. Drexl, R. Möhring, K. Neumann, and E. Pesch. Resource-constrained project scheduling: Notation, classification, models, and methods. *European Journal of Operational Research*, 112(1):3–41, 1999.

[Ben06] S. Benati. An optimization model for stochastic project networks with cash flows. *Computational Management Science*, 3(4):271–284, 2006.

[BEP$^+$96] J. Błażewicz, K. H. Ecker, E. Pesch, G. Schmidt, and J. Węglarz. *Scheduling Computer and Manufacturing Processes*. Springer, 2nd edition, 1996.

[Ber73] K. N. Berk. A central limit theorem for m-dependent random variables with unbounded m. *The Annals of Probability*, 1(2):352–354, 1973.

[Ber07] D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 2. Athena Scientific, 2007.

[Bil61] P. Billingsley. *Statistical Inference for Markov Processes.* The University of Chicago Press, 1961.

[Bil95] P. Billingsley. *Probability and Measure.* Wiley Blackwell, 3rd edition, 1995.

[BKPH05] S. P. Boyd, S.-J. Kim, D. D. Patil, and M. A. Horowitz. Digital circuit optimization via geometric programming. *Operations Research*, 53(6):899–932, 2005.

[BM95] J. R. Birge and M. J. Maddox. Bounds on expected project tardiness. *Operations Research*, 43(5):838–850, 1995.

[BNS01] J. D. Bagnell, A. Y. Ng, and J. Schneider. Solving uncertain Markov decision problems. Technical Report CMU-RI-TR-01-25, Carnegie Mellon University, 2001.

[BNT02] D. Bertsimas, K. Natarajan, and C.-P. Teo. Applications of semidefinite optimization in stochastic project scheduling. Technical report, Singapore–MIT Alliance, 2002.

[BR97] A. H. Buss and M. J. Rosenblatt. Activity delay in stochastic project networks. *Operations Research*, 45(1):126–139, 1997.

[Bru07] P. Brucker. *Scheduling Algorithms.* Springer, 5th edition, 2007.

[BS03] D. Bertsimas and M. Sim. Robust discrete optimization and network flows. *Mathematical Programming*, 98(1–3):49–71, 2003.

[BS04] D. Bertsimas and M. Sim. The price of robustness. *Operations Research*, 52(1):35–53, 2004.

[BS06] D. Bertsimas and M. Sim. Tractable approximations to robust conic optimization problems. *Mathematical Programming*, 107(1–2):5–36, 2006.

[BSS06] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming – Theory and Algorithms.* John Wiley & Sons, 3rd edition, 2006.

[BTGGN04] A. Ben-Tal, A. Goryashko, E. Guslitzer, and A. Nemirovski. Adjustable robust solutions of uncertain linear programs. *Mathematical Programming*, 99(2):351–376, 2004.

[BTGN09] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust Optimization.* Princeton University Press, 2009.

[BTN98] A. Ben-Tal and A. Nemirovski. Robust convex optimization. *Mathematics of Operations Research*, 23(4):769–805, 1998.

[Bus95] A. H. Buss. Sequential experimentation for estimating the optimal delay of activities in PERT networks. In C. Alexopoulos, K. Kang, W. R. Lilegdon, and D. Goldsman, editors, *Proceedings of the 1995 Winter Simulation Conference*, pages 336–340, 1995.

[CC05] G. Calafiore and M. C. Campi. Uncertain convex programs: Randomized solutions and confidence levels. *Mathematical Programming*, 102(1):25–46, 2005.

[CC06] G. Calafiore and M. C. Campi. The scenario approach to robust control design. *IEEE Transactions on Automatic Control*, 51(5):742–753, 2006.

[CGS07] I. Cohen, B. Golany, and A. Shtub. The stochastic time-cost tradeoff problem: A robust optimization approach. *Networks*, 49(2):175–188, 2007.

[CSS07] X. Chen, M. Sim, and P. Sun. A robust optimization perspective on stochastic programming. *Operations Research*, 55(6):1058–1071, 2007.

[CSSZ08] X. Chen, M. Sim, P. Sun, and J. Zhang. A linear-decision based approximation approach to stochastic programming. *Operations Research*, 56(2):344–357, 2008.

[DDH93] E. L. Demeulemeester, B. Dodin, and W. Herroelen. A random activity network generator. *Operations Research*, 41(5):972–980, 1993.

[DH02] E. L. Demeulemeester and W. S. Herroelen. *Project Scheduling – A Research Handbook*. Kluwer Academic Publishers, 2002.

[DHV$^+$95] R. F. Deckro, J. E. Hebert, W. A. Verdini, P. H. Grimsrud, and S. Venkateshwar. Nonlinear time/cost tradeoff models in project management. *Computers & Industrial Engineering*, 28(2):219–229, 1995.

[DM10] E. Delage and S. Mannor. Percentile optimization for Markov decision processes with parameter uncertainty. *Operations Research*, 58(1):203–213, 2010.

[DP94] A. K. Dixit and R. S. Pindyck. *Investment under Uncertainty*. Princeton University Press, 1994.

[DVH03] E. L. Demeulemeester, M. Vanhoucke, and W. S. Herroelen. RanGen: A random network generator for activity-on-the-node networks. *Journal of Scheduling*, 6(1):17–38, 2003.

[EH90] S. E. Elmaghraby and W. S. Herroelen. The scheduling of activities to maximize the net present value of projects. *European Journal of Operational Research*, 49(1):35–49, 1990.

[EI07] E. Erdoğan and G. Iyengar. On two-stage convex chance constrained problems. *Mathematical Methods of Operations Research*, 65(1):115–140, 2007.

[EK90] S. E. Elmaghraby and J. Kamburowski. On project representation and activity floats. *Arabian Journal of Science and Engineering*, 15(4B):626–637, 1990.

[EK92] S. E. Elmaghraby and J. Kamburowski. The analysis of activity networks under generalized precedence relations (GPRs). *Management Science*, 38(9):1245–1263, 1992.

[Elm00] S. E. Elmaghraby. On criticality and sensitivity in activity networks. *European Journal of Operational Research*, 127(2):220–238, 2000.

[Elm05] S. E. Elmaghraby. On the fallacy of averages in project risk management. *European Journal of Operational Research*, 165(2):307–313, 2005.

[EMS09] A. L. Erera, J. C. Morales, and M. Savelsbergh. Robust optimization for empty repositioning problems. *Operations Research*, 57(2):468–483, 2009.

[Epp94] D. Eppstein. Finding the k shortest paths. In *IEEE Symposium on Foundations of Computer Science*, pages 154–165, 1994.

[Fis70] P. C. Fishburn. *Utility theory for decision making.* John Wiley & Sons, 1970.

[FJMM07] U. Feige, K. Jain, M. Mahdian, and V. Mirrokni. Robust combinatorial optimization with exponential scenarios. In M. Fischetti and D. P. Williamson, editors, *Integer Programming and Combinatorial Optimization*, pages 439–453. Springer Lecture Notes in Computer Science, 2007.

[FL04] C. A. Floudas and X. Lin. Continuous-time versus discrete-time approaches for scheduling of chemical processes: A review. *Computers & Chemical Engineering*, 28(11):2109–2129, 2004.

[Ful61] D. R. Fulkerson. A network flow computation for project cost curves. *Management Science*, 7(2):167–178, 1961.

[GG06] V. Goel and I. E. Grossmann. A class of stochastic programs with decision dependent uncertainty. *Mathematical Programming*, 108(2–3):355–394, 2006.

[GH68] T. J. Gordon and H. Hayward. Initial experiments with the cross-impact matrix method of forecasting. *Futures*, 1(2):100–116, 1968.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman & Co, 1979.

[GLD00] R. Givan, S. Leach, and T. Dean. Bounded-parameter Markov decision processes. *Artificial Intelligence*, 122(1–2):71–109, 2000.

[Gri72] R. C. Grinold. The payment scheduling problem. *Naval Research Logistics Quarterly*, 19(1):123–136, 1972.

[GS09] J. Goh and M. Sim. Distributionally robust optimization and its tractable approximations. *Accepted for Publication in Operations Research*, 2009.

[Hag88] J. N. Hagstrom. Computational complexity of PERT problems. *Networks*, 18(2):139–147, 1988.

[HDD97] W. S. Herroelen, P. Van Dommelen, and E. L. Demeulemeester. Project network models with discounted cash flows – A guided tour through recent developments. *European Journal of Operational Research*, 100(1):97–121, 1997.

[HK93] R. Hettich and K. O. Kortanek. Semi-infinite programming: Theory, methods, and applications. *SIAM Review*, 35(3):380–429, 1993.

[HKR09] R. Henrion, C. Küchler, and W. Römisch. Scenario reduction in stochastic programming with respect to discrepancy distances. *Computational Optimization and Applications*, 43(1):67–93, 2009.

[HL04] W. Herroelen and R. Leus. Robust and reactive project scheduling: A review and classification of procedures. *International Journal of Production Research*, 42(8):1599–1620, 2004.

[HL05] W. S. Herroelen and R. Leus. Project scheduling under uncertainty: Survey and research potentials. *European Journal of Operational Research*, 165(2):289–306, 2005.

[HPT00] R. Horst, P. M. Pardalos, and N. V. Thoai. *Introduction to Global Optimization.* Kluwer Academic Publishers, 2nd edition, 2000.

[HR03] H. Heitsch and W. Römisch. Scenario reduction algorithms in stochastic programming. *Computational Optimization and Applications*, 24(2–3):187–206, 2003.

[HS08] R. Henrion and C. Strugarek. Convexity of chance constraints with independent random variables. *Computational Optimization and Applications*, 41(2):263–276, 2008.

[Iba80] T. Ibaraki. Approximate algorithms for the multiple-choice continuous knapsack problems. *Journal of the Operations Research Society of Japan*, 23(1):28–63, 1980.

[Iye05] G. N. Iyengar. Robust dynamic programming. *Mathematics of Operations Research*, 30(2):257–280, 2005.

[JLF07] S. L. Janak, X. Lin, and C. A. Floudas. A new robust optimization approach for scheduling under uncertainty: II. Uncertainty with known probability distribution. *Computers & Chemical Engineering*, 31(3):171–195, 2007.

[JM99] A. S. Jain and S. Meeran. Deterministic job-shop scheduling: Past, present and future. *European Journal of Operational Research*, 113(2):390–434, 1999.

[JW00] T. Jørgensen and S. W. Wallace. Improving project cost estimation by taking into account managerial flexibility. *European Journal of Operational Research*, 127(2):239–251, 2000.

[JWW98] T. W. Jonsbråten, R. J.-B. Wets, and D. L. Woodruff. A class of stochastic programs with decision dependent random elements. *Annals of Operations Research*, 82(1):83–106, 1998.

[KA86] V. G. Kulkarni and V. G. Adlakha. Markov and Markov-regenerative PERT networks. *Operations Research*, 34(5):769–781, 1986.

[KBY$^+$07] S.-J. Kim, S. P. Boyd, S. Yun, D. D. Patil, and M. A. Horowitz. A heuristic for optimizing stochastic activity networks with applications to statistical digital circuit sizing. *Optimization and Engineering*, 8(4):397–430, 2007.

[Kel61] J. E. Kelley. Critical-path planning and scheduling: Mathematical basis. *Operations Research*, 9(3):296–320, 1961.

[KKMS08] R. Khandekar, G. Kortsarz, V. Mirrokni, and M. R. Salavatipour. Two-stage robust network design with exponential scenarios. In D. Halperin and K. Mehlhorn, editors, *Algorithms – ESA 2008*, pages 589–600. Springer Lecture Notes in Computer Science, 2008.

[KPK07] J. Korski, F. Pfeuffer, and K. Klamroth. Biconvex sets and optimization with biconvex functions: A survey and extensions. *Mathematical Methods of Operations Research*, 66(3):373–407, 2007.

[KW94] P. Kall and S. W. Wallace. *Stochastic programming.* John Wiley & Sons, 1994.

[KWG09] D. Kuhn, W. Wiesemann, and A. Georghiou. Primal and dual linear decision rules in stochastic and robust optimization. *Accepted for Publication in Mathematical Programming*, 2009.

[KY97] P. Kouvelis and G. Yu. *Robust Discrete Optimization and its Applications.* Kluwer Academic Publishers, 1997.

[LA08] J. Luedtke and S. Ahmed. A sample approximation approach for optimization with probabilistic constraints. *SIAM Journal on Optimization*, 19(2):674–699, 2008.

[LI08] Z. Li and M. Ierapetritou. Process scheduling under uncertainty: Review and challenges. *Computers & Chemical Engineering*, 32(4–5):715–727, 2008.

[LJF04] X. Lin, S. L. Janak, and C. A. Floudas. A new robust optimization approach for scheduling under uncertainty: I. Bounded uncertainty. *Computers & Chemical Engineering*, 28(6–7):1069–1085, 2004.

[LLMS09] C. Liebchen, M. E. Lübbecke, R. H. Möhring, and S. Stiller. The concept of recoverable robustness, linear programming recovery, and railway applications. In R. K. Ahuja, R. H. Möhring, and C. D. Zaroliagis, editors, *Robust and Online Large-Scale Optimization*, pages 1–27. Springer Lecture Notes in Computer Science, 2009.

[LMS01] A. Ludwig, R. H. Möhring, and F. Stork. A computational study on bounding the makespan distribution in stochastic project networks. *Annals of Operations Research*, 102(1):49–64, 2001.

[LP94] C. Limongelli and R. Pirastu. Exact solution of linear equation systems over rational numbers by parallel p-adic arithmetic. RISC Report Series 94–25, University of Linz, 1994.

[LVBL98] M. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284(1–3):193–228, 1998.

[M01] R. H. Möhring. Scheduling under uncertainty: Bounding the makespan distribution. In H. Alt, editor, *Conputational Discrete Mathematics*, pages 79–97. Springer Lecture Notes in Computer Science, 2001.

[Mar52] H. Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952.

[MCWG95] A. Mas-Colell, M. D. Whinston, and J. R. Green. *Microeconomic Theory.* Oxford University Press, 1995.

[MN79] I. Meilijson and A. Nádas. Convex majorization with an application to the length of critical paths. *Journal of Applied Probability*, 16(3):671–677, 1979.

[Mon82] G. E. Monahan. A survey of partially observable Markov decision processes: Theory, models, and algorithms. *Management Science*, 28(1):1–16, 1982.

[MRCF59] D. G. Malcolm, J. H. Roseboom, C. E. Clark, and W. Fazar. Application of a technique for research and development program evaluation. *Operations Research*, 7(5):646–669, 1959.

[MS00] R. Möhring and F. Stork. Linear preselective policies for stochastic project scheduling. *Mathematical Methods of Operations Research*, 52(3):501–515, 2000.

[MSST07] S. Mannor, D. Simester, P. Sun, and J. N. Tsitsiklis. Bias and variance approximation in value function estimates. *Management Science*, 53(2):308–322, 2007.

[Neu79] K. Neumann. Recent advances in temporal analysis of GERT networks. *Zeitschrift für Operations Research*, 23:153–177, 1979.

[Neu99] K. Neumann. Scheduling of projects with stochastic evolution structure. In J. Węglarz, editor, *Project Scheduling: Recent Models, Algorithms, and Applications*, pages 309–332. Kluwer Academic Publishers, 1999.

[NG05] A. Nilim and L. El Ghaoui. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.

[NS06a] A. Nemirovski and A. Shapiro. Convex approximations of chance constrained programs. *SIAM Journal on Optimization*, 17(4):969–996, 2006.

[NS06b] A. Nemirovski and A. Shapiro. Scenario approximations of chance constraints. In G. Calafiore and F. Dabbene, editors, *Probabilistic and Randomized Methods for Design under Uncertainty*, pages 3–47. Springer, 2006.

[NSZ03] K. Neumann, C. Schwindt, and J. Zimmermann. *Project scheduling with time windows and scarce resources.* Springer, 2003.

[NZ00] K. Neumann and J. Zimmermann. Procedures for resource leveling and net present value problems in project scheduling with general temporal and resource constraints. *European Journal of Operational Research*, 127(2):425–443, 2000.

[Ö98] L. Özdamar. On scheduling project activities with variable expenditure rates. *IIE Transactions*, 30(8):695–704, 1998.

[OD97] L. Özdamar and H. Dündar. A flexible heuristic for a multi-mode capital constrained project scheduling problem with probabilistic cash inflows. *Computers and Operations Research*, 24(12):1187–1200, 1997.

[OZ07] F. Ordóñez and J. Zhao. Robust capacity expansion of network flows. *Networks*, 50(2):136–145, 2007.

[Pet75] V. V. Petrov. *Sums of Independent Random Variables.* Springer, 1975.

[Pin08] M. L. Pinedo. *Scheduling: Theory, Algorithms, and Systems.* Springer, 3rd edition, 2008.

[PK08] I. C. Paschalidis and S.-C. Kang. A robust approach to Markov decision problems with uncertain transition probabilities. In *Proceedings of the 17th IFAC World Congress*, pages 408–413, 2008.

[Pré95] A. Prékopa. *Stochastic Programming.* Kluwer Academic Publishers, 1995.

[Pri66] A. A. B. Pritsker. GERT: Graphical evaluation and review technique. Memorandum RM–49730–NASA, The RAND Corporation, April 1966.

[Put94] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* John Wiley & Sons, 1994.

[PW07] G. C. Pflug and D. Wozabal. Ambiguity in portfolio selection. *Quantitative Finance*, 7(4):435–442, 2007.

[Roc70] R. T. Rockafellar. *Convex Analysis.* Princeton University Press, 1970.

[RS03] A. Ruszczyński and A. Shapiro, editors. *Stochastic programming.* Elsevier Science B.V., 2003.

[RU00] R. T. Rockafellar and S. Uryasev. Optimization of conditional value-at-risk. *Journal of Risk*, 2(3):21–41, 2000.

[Rus70] A. H. Russell. Cash flows in networks. *Management Science*, 16(5):357–373, 1970.

[Sah04] N. V. Sahinidis. Optimization under uncertainty: State-of-the-art and opportunities. *Computers & Chemical Engineering*, 28(6–7):971–983, 2004.

[Sch95] P. J. H. Schoemaker. Scenario planning: A tool for strategic thinking. *Sloan Management Review*, 36(2):25–40, 1995.

[Sch01] A. Scholl. *Robuste Planung und Optimierung – Grundlagen, Konzepte und Methoden, Experimentelle Untersuchungen.* Physica-Verlag, 2001. (IN GERMAN.).

[Sch05] C. Schwindt. *Resource Allocation in Project Management.* Springer, 2005.

[SL73] J. K. Satia and R. E. Lave. Markovian decision processes with uncertain transition probabilities. *Operations Research*, 21(3):728–740, 1973.

[SN05] A. Shapiro and A. Nemirovski. On complexity of stochastic programming problems. In V. Jeyakumar and A. Rubinov, editors, *Continuous Optimization – Current Trends and Modern Applications*, pages 111–146. Springer, 2005.

[SNLS05] J. Singh, V. Nookala, Z.-Q. Luo, and S. Sapatnekar. Robust gate sizing by geometric programming. In *DAC '05: Proceedings of the 42nd Annual Conference on Design Automation*, pages 315–320, 2005.

[SSH99] I. E. Sutherland, R. F. Sproull, and D. F. Harris. *Logical Effort: Designing fast CMOS Circuits.* Morgan Kaufmann, 1999.

[Sti09] S. Stiller. *Extending Concepts of Reliability. Network Creation Games, Realtime Scheduling, and Robust Optimization.* PhD thesis, Technische Universität Berlin, 2009.

[SZ01] C. Schwindt and J. Zimmermann. A steepest ascent approach to maximizing the net present value of projects. *Mathematical Methods of Operations Research*, 53(3):435–450, 2001.

[Tah97] H. A. Taha. *Operations Research: An Introduction.* Prentice Hall, 6th edition, 1997.

[TFC98] L. V. Tavares, J. A. A. Ferreira, and J. S. Coelho. On the optimal management of project risk. *European Journal of Operational Research*, 107(2):451–469, 1998.

[Tsi07] J. N. Tsitsiklis. Computational complexity in Markov decision theory. *HERMIS – An International Journal of Computer Mathematics and its Applications*, 9(1):45–54, 2007.

[TSS06] V. Tilson, M. J. Sobel, and J. G. Szmerekovsky. Scheduling projects with stochastic activity duration to maximize EPV. Technical Memorandum No. 812, Case Western Reserve University, 2006.

[VB96] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.

[vdAtHKB03] W. M. P. van der Aalst, A. H. M. ter Hofstede, B. Kiepuszewski, and A. P. Barros. Workflow patterns. *Distributed and Parallel Databases*, 14(1):5–51, 2003.

[WA08] W. Wang and S. Ahmed. Sample average approximation of expected value constrained stochastic programs. *Operations Research Letters*, 36(5):515–519, 2008.

[WE94] C. C. White and H. K. Eldeib. Markov decision processes with imprecise transition probabilities. *Operations Research*, 42(4):739–749, 1994.

[WGY09] Z. Wang, P. W. Glynn, and Y. Ye. Likelihood robust optimization for data-driven newsvendor problems. Working paper, Stanford University, 2009.

[WHK08] W. Wiesemann, R. Hochreiter, and D. Kuhn. A stochastic programming approach for QoS-aware service composition. In *Proceedings of the 8th IEEE International Symposium on Cluster Computing and the Grid*, 2008.

[Wil99] H. P. Williams. *Model building in mathematical programming.* John Wiley & Sons, 1999.

[Wol85] R. D. Wollmer. Critical path planning under uncertainty. *Mathematical Programming Study*, 25(1):164–171, 1985.

[WWS00] J. Wang, Z. Wu, and Y. Sun. Optimum activity delay in stochastic activity networks. In W. Kubiak, C.-Y. Lee, J. Węglarz, and R. Willis, editors, *Proceedings of the 7th International Workshop on Project Management and Scheduling*, 2000.

[XM06] H. Xu and S. Mannor. The robustness-performance tradeoff in Markov decision processes. In *Advances in Neural Information Processing Systems*, pages 1537–1544, 2006.

[Yan05] I.-T. Yang. Impact of budget uncertainty on project time-cost tradeoff. *IEEE Transactions on Engineering Management*, 52(2):176–174, 2005.

[ZF09] S.-S. Zhu and M. Fukushima. Worst-case conditional value-at-risk with application to robust portfolio management. *Operations Research*, 57(5):1155–1168, 2009.