COMISEF Computational Optimization Methods in Statistics, Econometrics and Finance

- Marie Curie Research and Training Network funded by the EU Commission through MRTN-CT-2006-034270 -

COMISEF WORKING PAPERS SERIES

WPS-020 06/11/2009

Robust Resource Allocations in Temporal Networks

W. Wiesemann
D. Kuhn
B. Rustem

Robust Resource Allocations in Temporal Networks

Wolfram Wiesemann, Daniel Kuhn and Berç Rustem
Imperial College of Science, Technology and Medicine
London SW7 2AZ, United Kingdom

Abstract

Temporal networks describe workflows of time-consuming tasks whose processing order is constrained by precedence relations. In many cases, the durations of the network tasks can be influenced by the assignment of resources. This leads to the problem of selecting an 'optimal' resource allocation, where optimality is measured by network characteristics such as the makespan (i.e., the time required to complete all tasks). In this paper, we study a robust resource allocation problem where the functional relationship between task durations and resource assignments is uncertain, and the goal is to minimise the worst-case makespan. We show that this problem is generically $\mathcal{NP}$-hard. We then develop convergent bounds for the optimal objective value, as well as feasible allocations whose objective values are bracketed by these bounds. Numerical results provide empirical support for the proposed method.

Keywords: Robust Optimisation · Temporal Networks · Resource Allocation Problem

1 Introduction

Many problems in operations research are concerned with time-consuming tasks whose processing order is constrained by precedence relations. We can describe such situations by temporal networks: a *temporal network* is a directed, acyclic graph $G = (V, E)$ whose nodes $V = \{1, \ldots, n\}$ represent the tasks and whose arcs $E \subset V \times V$ denote the temporal precedences. To simplify the exposition, we assume that $1 \in V$ denotes the unique source and $n \in V$ the unique sink of G. This can always be achieved by adding dummy tasks and/or precedences to the graph. In resource allocation problems on temporal networks, the time required to process task $i \in V$ depends on the chosen resource allocation. The goal is to determine an allocation which optimises a time-related objective, most frequently the makespan of the network (i.e., the time required to complete all tasks). To formalise this idea, we assign to every task $i \in V$ a function $d_i : X \mapsto \mathbb{R}_+$ that maps resource allocations $x \in X$ to non-negative durations. We assume that d_i is continuous and that X, the set of admissible allocations, is a nonempty and compact subset of a finite-dimensional space. The *resource allocation problem on temporal networks* can then be defined as

$$\min_{\substack{x \in X, \\ y \in Y(x)}} \; \{y_n + d_n(x)\}, \tag{1a}$$

where

$$Y(x) = \left\{ y \in \mathbb{R}^n_+ \, : \, y_j \geq y_i + d_i(x) \; \forall \, (i, j) \in E \right\}. \tag{1b}$$

For $x \in X$, $Y(x)$ denotes the set of admissible start time vectors for the network tasks, that is, the ith component of $y \in Y(x)$ assigns a start time to task $i \in V$ such that all precedence constraints are satisfied. Since n is the unique sink of G, $y_n + d_n(x)$ represents the makespan of the network. Note that every admissible start time schedule $y \in Y(x)$ has to satisfy $y_1 \geq 0$ and

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

Since the makespan is a nondecreasing function of y, the *early start schedule* $y^* : X \mapsto \mathbb{R}^n_+$ with $y^*_1(x) = 0$ and

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

is optimal for any fixed allocation $x \in X$. Note that the recursion is well-defined because G is acyclic. The optimality of the early start schedule is an important property that distinguishes model (1) from other optimisation problems on temporal networks.

Resource allocation problems on temporal networks arise in various areas of operations research, such as project management [6, 10], service-oriented computing [26, 35], digital circuit sizing [5] and the scheduling of machines [31], chemical processes [12] and multiprocessor applications [22]. Although there is an abundance of contributions that address instances of the deterministic resource allocation problem, the literature on its counterpart under uncertainty is surprisingly sparse. Broadly speaking, optimisation models can account for uncertainty in two ways. If the uncertain parameters are modelled as random variables with known distributions, solution approaches from stochastic programming can be employed [28, 29]. If, on the other hand, merely the support of the uncertain problem parameters is known, distribution-free risk measures from decision making under strict uncertainty can be optimised. In recent years, optimisation in view of the worst case (minimax criterion) has received considerable attention and is commonly referred to as *robust optimisation* [1, 4, 14]. Most resource allocation problems under uncertainty assume known distributions [17, 24, 30], which can be explained by the novelty of efficient robust optimisation techniques in operations research.

In this text we consider a robust resource allocation problem that employs the minimax objective. Contrary to its counterpart under probabilistic uncertainty, the complexity of this problem is unknown [16]. Instances of the considered problem have been employed in different application areas. The existing contributions have in common that they determine suboptimal solutions and do not provide bounds for the incurred optimality gap. In this paper, we show that the robust resource allocation problem is $\mathcal{NP}$-hard, which provides an explanation for the lack of exact solution approaches in the literature. We then develop families of optimisation problems that provide convergent lower and upper bounds for the optimal value of the problem. The upper bounds correspond to feasible allocations whose objective values are bracketed by these bounds. Hence, we obtain a series of feasible allocations that converge to the optimal solution and whose optimality gaps can be quantified at any time. Although we focus on the minimax objective, our method remains applicable when more information about the probability distribution of the uncertain parameters is available. In such cases, our approach can be used to optimise a conservative approximation of the (conditional) value-at-risk. The details of this extension follow the lines of [7] and are left out for brevity.

The remainder of this paper is organised as follows. In the next section, we define the robust resource allocation problem. After a review of related literature, we show that the problem is generically $\mathcal{NP}$-hard. In Section 3 we discuss a path-wise formulation that provides the basis for our solution technique. In Sections 4 and 5 we develop families of optimisation problems that provide convergent lower and upper bounds, respectively. Section 6 presents the results of a numerical evaluation, and we conclude in Section 7.

Notation Unless stated differently, lower case Latin and Greek letters denote column vectors. We refer to the ith component of vector x by x_i. $\|x\|_p$ denotes the p-norm of vector x. e represents the vector of all ones; its dimension will be clear from the context. For set $A \subseteq \{1, \ldots, n\}$, $\mathbb{I}_A$ denotes the n-dimensional vector with $(\mathbb{I}_A)_i = 1$ if $i \in A$ and $(\mathbb{I}_A)_i = 0$ otherwise.

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.

2 Robust Resource Allocations

We first introduce the robust counterpart of problem (1). We then provide a survey of solution approaches proposed in the literature. In Section 2.3 we show that the robust resource allocation problem is generically $\mathcal{NP}$-hard.

2.1 The Robust Resource Allocation Problem

Throughout this paper, we assume that the structure of the temporal network (i.e., V and E) is deterministic, whereas the task durations are uncertain.[1] 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 and realisations of the uncertain parameters $\xi \in \Xi$ to non-negative durations. We assume that Ξ, the support of the uncertain parameters, is a nonempty and compact subset of a finite-dimensional space. Having in mind the application areas outlined in Section 1, we assume that ξ cannot be observed directly, but that it can only be gradually inferred from the durations of completed tasks. 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 with finite discrete supports Ξ can be formulated as explicit convex programs. Often, however, Ξ is better described by a set of infinite cardinality, for example an ellipsoid around a nominal parameter vector. In this paper, we focus on uncertainty sets that are of infinite cardinality but specific structure.

We define the *<u>r</u>obust resource allocation problem on <u>t</u>emporal <u>n</u>etworks* as

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

[1]Uncertain network structures are addressed in the literature on GERT networks [25].

where

$$Y(x,\xi) = \left\{ y \in \mathbb{R}^n_+ \,:\, y_j \geq y_i + d_i(x;\xi) \;\forall\, (i,j) \in E \right\}. \tag{2}$$

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. Resource allocations are frequently required to be static due to the inflexibility of resources and limitations of the manufacturing process (see Section 6), 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 [15, 21]. In the applications described in Section 1, unlike the resource allocation x, the task start times y may depend on the available knowledge about ξ. We justify this assumption for circuit design problems in Section 6. Note that every component of y is chosen after *all* uncertain parameters are revealed, which seems to violate non-anticipativity [28, 29]: 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 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, however, since the task start times only depend on the completion times of predecessor tasks. As in the deterministic case described in Section 1, the early start schedule is also optimal. Hence, if a solution of $\mathcal{RTN}$ employs an anticipative start time schedule y, we can replace it with the corresponding (non-anticipative) early start schedule without sacrificing optimality.

$\mathcal{RTN}$ has relevance in all application areas outlined in the previous section. The solution approach proposed in this paper 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 (see Section 6). Alternative formulations, such as the minimisation of deviations from a static baseline schedule $\widehat{y}$ (which itself becomes part of the first-stage decision), are discussed in [17].

2.2 Literature Review

$\mathcal{RTN}$ 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.

Observation 2.1 *For the robust resource allocation problem $\mathcal{RTN}$, we have*

$$\min_{x \in X} \max_{\xi \in \Xi} \min_{y \in Y(x,\xi)} \; \{ y_n + d_n(x;\xi) \} \;=\; \min_{\substack{x \in X, \\ y \in \mathcal{Y}(x)}} \; \max_{\xi \in \Xi} \; \{ y_n(\xi) + d_n(x;\xi) \}, \tag{3a}$$

where for $x \in X$,

$$\mathcal{Y}(x) = \left\{ (y : \Xi \mapsto \mathbb{R}^n_+) \,:\, y(\xi) \in Y(x,\xi) \;\forall\, \xi \in \Xi \right\}. \tag{3b}$$

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

We can thus reduce the min-max-min problem $\mathcal{RTN}$ to a min-max problem at the cost of augmenting the set of first-stage decisions. 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 points, $\Xi = \{\xi^1, \ldots, \xi^L\}$, then $\mathcal{Y}(x)$ is isomorphic to a subset of $\mathbb{R}_+^{Ln}$ and we can reformulate $\mathcal{RTN}$ as

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

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 [18]. This approach, however, would only provide lower bounds for the optimal value of $\mathcal{RTN}$, and it is not clear how to efficiently obtain upper bounds.[2] Furthermore, one would not be able to exploit structural properties of Ξ and d beyond convexity. Finally, the number of constraints and variables grows 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. It has been suggested in [24] to restrict $\mathcal{Y}$ to *constant decision rules*, that is, to

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

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

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

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 [18]. For specific choices of Ξ and d, robust optimisation techniques can be used to obtain equivalent (or approximate) explicit reformulations [1, 2, 3, 4]. 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.

[2] As we will see in Section 2.3, evaluating the worst-case makespan of the optimal second-stage policy in $\mathcal{RTN}$ constitutes a difficult problem even for fixed $x \in X$.

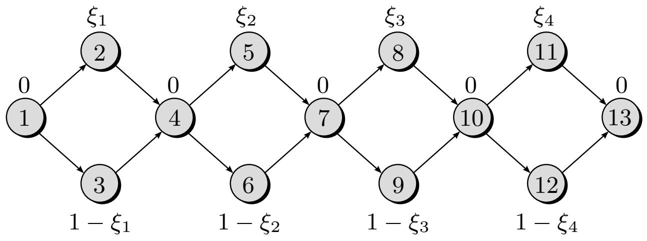


Figure 1: Example temporal network that illustrates the suboptimality of affine decision rules. The graph visualises the network structure for $k = 4$. The task durations (next to the nodes) are defined in the text.

Example 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 [8] to approximate $\mathcal{Y}(x)$ by a set of *affine decision rules*: for $x \in X$ and $\Xi \subseteq \mathbb{R}^k$, we define

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

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

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

with

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

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 2.2 Consider the class of temporal networks illustrated in Figure 1. For $k \in \mathbb{N}$, the network structure is given by $V = \{1, \ldots, 3k + 1\}$ and $E = \{(3l + 1, 3l + p), (3l + p, 3l + 4) : 0 \leq l < k, \, p = 2, 3\}$. 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 $\Xi = \left\{\xi \in \mathbb{R}^k_+ : \|\xi - 1/2\mathrm{e}\|_1 \leq 1/2\right\}$, the optimal second-stage policy leads to a worst-case 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 $\xi \in \left\{1/2\mathrm{e} \pm 1/2\mathrm{e}^{l+1}\right\}$, 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

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

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 [9].

The Examples 2.1 and 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 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.

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 *w̲orst-c̲ase m̲akespan of a t̲emporal n̲etwork* (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 $w, u \in \mathbb{N}^n_0$ 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*

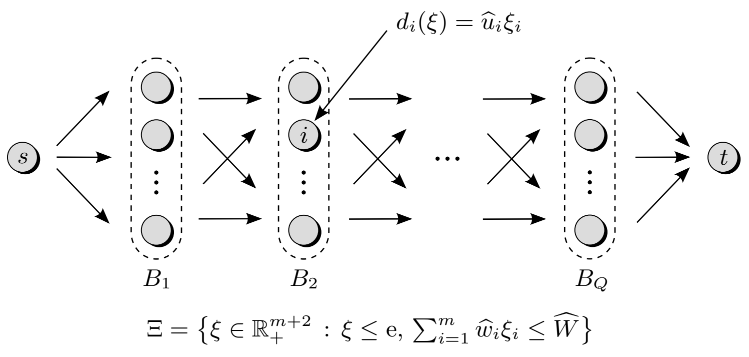
$$\min_{y \in \mathbb{R}^n_+} \ \{y_n + u_n \xi_n \ : \ y_j \geq y_i + u_i \xi_i \ \forall\, (i,j) \in E\} \geq U? \tag{4}$$

WCMTN considers instances of $\mathcal{RTN}$ with 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 [3]. WCMTN asks whether the worst-case makespan exceeds U when an optimal start time schedule is implemented.

Theorem 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'.

In order to establish WCMTN's membership to $\mathcal{NP}$, we need to show that we can guess a ξ, check whether $\xi \in \Xi$, construct an admissible y^* that minimises the left-hand side of (4) 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



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

Figure 2: WCMTN instance constructed from a CMCK instance.

polynomial. Then we can check in polynomial time whether $\xi \in \Xi$. Moreover, optimality of the early start schedule (see Section 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 (4). In particular, this y^* also possesses a polynomial bit length and can be determined in polynomial time. This implies that validity of (4) can be verified in polynomial time, which in turn implies membership of WCMTN to $\mathcal{NP}$. It remains to be shown that we can indeed restrict attention to values of ξ with polynomial bit lengths. (4) is satisfied for some $\xi \in \Xi$ if and only if

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

Since the inner minimisation represents a convex function of ξ, its maximum over Ξ is attained by at least one extreme point of Ξ [19]. Since Ξ is a polyhedron, however, all of its extreme points possess polynomial bit lengths [23].

In order to prove $\mathcal{NP}$-hardness of WCMTN, we consider the Continuous Multiple Choice Knapsack (CMCK) problem:
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 $\mathcal{B}$, i.e., $\bigcup_q B_q = \mathcal{B}$ 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. Hence, the existence of a polynomial-time solution procedure for WCMTN would imply the existence of such a procedure for CMCK. Since CMCK is known to be $\mathcal{NP}$-hard [13, 20], this implies that WCMTN is $\mathcal{NP}$-hard as well.

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 $E_G = \{(s,i) : i \in B_1\} \cup \{(i,t) : i \in B_Q\}$. 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 $i \in \{s, t\}$. W and U are identified with $\widehat{W}$ and $\widehat{U}$ of the CMCK instance, respectively. The transformation is illustrated in Figure 2.

For the constructed WCMTN instance, assume that there is a $\xi \in \Xi$ which satisfies (4). Let y^* be a minimiser for the left-hand side of (4). 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 $y^*_t = y^*_{b_Q} + u_{b_Q}\xi_{b_Q}$ [10]. 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 (4) implies that the answer to the CMCK instance is negative. ■

Theorem 2.1 immediately extends to problem instances whose uncertainty sets are polyhedral [3] or that result from intersections of general ellipsoids as in [1]. It also serves as an indicator that other uncertainty sets might result in difficult optimisation problems, too. Note, however, 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 with $w = \alpha \mathrm{e}$ and $u = \beta \mathrm{e}$ for $\alpha, \beta \in \mathbb{N}_0$.

3 Path-Wise Problem Formulation

In contrast to the techniques reviewed in Section 2.2, our solution approach for $\mathcal{RTN}$ does not rely on approximating decision rules. Instead, we eliminate the inner minimisation in $\mathcal{RTN}$ by enumerating the task paths of the network. A solution scheme based on path enumeration has recently been proposed in project scheduling under a probabilistic uncertainty model [34]. In this section, we present a path-wise reformulation of $\mathcal{RTN}$ and argue that it is unsuited for direct solution. In the next two sections, we will use this reformulation to derive convergent bounds for 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 e.g. [10]) 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 3.1 *For a temporal network $G = (V, E)$ with fixed allocation $x \in X$ and parameters $\xi \in \Xi$, the minimal makespan is given by*

$$\min_{y \in Y(x,\xi)} \{y_n + d_n(x;\xi)\} = \max_{P \in \mathcal{P}} \left\{\mathbb{I}_P^\top d(x;\xi)\right\}, \tag{5}$$

where $d(x;\xi) = (d_1(x;\xi), \ldots, d_n(x;\xi))^\top$ and $Y(x,\xi)$ is defined in (2).

Note that the maximum on the right-hand side of (5) can be attained by several task paths $P \in \mathcal{P}$. Observation 3.1 allows us to replace the inner minimisation in $\mathcal{RTN}$ by the right-hand side of (5), and thus we find

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

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*:

$$\min_{\substack{x \in X, \\ \tau \in \mathbb{R}_+}} \max_{P \in \mathcal{P}} \phi(x; P), \qquad (\mathcal{ARTN})$$

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

$$\phi(x; P) \geq \max_{\xi \in \Xi} \left\{ \mathbb{I}_P^\top d(x; \xi) \right\} \quad \text{for } x \in X,\ P \subseteq V. \tag{6}$$

If (6) 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 (6) can be replaced by 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 for exact and conservative reformulations of $\mathcal{RTN}$ alike. Note that we do not consider progressive reformulations where the inequality in (6) is inverted, because we seek for resource allocations with guaranteed upper bounds for the makespan.

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 $\mathbb{I}_P \leq \mathbb{I}_{P'}$. 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 $\mathcal{P}$ by $\overline{\mathcal{P}}$. (A2) implies that $\phi(x; P)$ is bounded from above by $\sum_r \phi(x; P_r)$ for all $x \in X$ if $\{P_r\}_r$ forms a partition of P. As we will see, this bounding property facilitates the construction of lower and upper bounds for the optimal value of $\mathcal{ARTN}$. The following proposition shows that exact reformulations of $\mathcal{RTN}$ necessarily satisfy (A1) and (A2).

Proposition 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 $\mathbb{I}_P \leq \mathbb{I}_{P'}$ 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.

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

1. ***Affine Uncertainty.*** *$d_i(x;\xi) = \delta_i^0(x) + \xi^\top \left[\delta_i^1(x)\right]$ 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*

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

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

2. ***Quadratic Uncertainty.*** *$d_i(x;\xi) = \delta_i^0(x) + \xi^\top \left[\delta_i^1(x)\right] + \left\| \left[\Delta_i^2(x)\right] \xi \right\|_2^2$ 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*

$$\Xi = \Big\{ \xi \in \mathbb{R}^k \, : \, \exists \, u \in \mathbb{R}^J \, . \, \xi = \sigma + \Sigma u, \ \left\| u \right\|_2 \leq 1 \Big\},$$

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

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

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

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

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

Here, the operator 'vec' returns the concatenation of its arguments as a column vector. Note that we have $\left[\mathbb{I}_P\right]_i = 1$ if P contains task i and $\left[\mathbb{I}_P\right]_i = 0$ otherwise. For both definitions of ϕ, the epigraph of ϕ can be described by a semi-infinite constraint which has to hold for all $\xi \in \Xi$. Robust optimisation techniques [1] enable us to reformulate these semi-infinite constraints such that (A3) is satisfied. Due to Proposition 3.1, (A1) and (A2) are satisfied as well. ∎

The uncertainty sets considered in Proposition 3.2 cover all bounded polyhedra as special cases. It is desirable to extend the results of Proposition 3.2 also to problems with conic-quadratic uncertainty.

Proposition 3.3 (Conic-Quadratic Uncertainty) *Let* $d_i(x;\xi) = \delta_i^0(x) + \xi^\top[\delta_i^1(x)] + \left\|[\Delta_i^2(x)]\,\xi\right\|_2$ *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*

$$\Xi = \left\{\xi \in \mathbb{R}^k \,:\, \exists\, u \in \mathbb{R}^J \,.\, \xi = \sigma + \Sigma u,\ \|u\|_2 \le 1\right\},$$

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

Remark. 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 3.3 We construct an upper bound for

$$\max_{\xi\in\Xi}\ \left\{\sum_{i\in P}\left(\delta_i^0(x) + \xi^\top[\delta_i^1(x)] + \left\|[\Delta_i^2(x)]\xi\right\|_2\right)\right\} \quad \text{for } x \in X,\ P \in \mathcal{P}. \tag{7}$$

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 [4] and bound (7) from above by

$$\phi(x;P) = \max_{\widehat{u}\in\widehat{\mathcal{U}}}\ \left\{\mathbb{I}_P^\top\left[\widehat{d}(x;\widehat{u})\right]\right\}, \tag{8}$$

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

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

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

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

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 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 $\|u\|_2 \le 1$. Hence, for $x \in X$,

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

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

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

Proposition 3.3 provides a conservative reformulation of $\mathcal{RTN}$. Exact reformulations of robust optimisation problems subject to conic-quadratic uncertainty are discussed in [1]. However, the path durations $\phi(x;P)$ resulting from conic-quadratic uncertainty are not of the form required in [1], and the corresponding reformulation is not applicable in our context.

$\mathcal{ARTN}$ may appear to be efficiently solvable whenever (A3) holds. Note, however, that the size of $\mathcal{ARTN}$ depends on the cardinality of $\mathcal{P}$, which in turn can be exponential in the size of G. As an illustration, consider the temporal network of Figure 1: it has $|V| = 3k + 1$ nodes and $|E| = 4k$ arcs, $k \in \mathbb{N}$, but contains $\left|\overline{\mathcal{P}}\right| = 2^k$ inclusion-maximal paths. It can further be shown that the expected number of paths in a uniformly sampled random temporal network is exponential. We defer the proof of this statement to the appendix. Hence, even though $\mathcal{ARTN}$ can be expressed as an explicit convex optimisation problem, it remains difficult to solve.

4 Lower Bounds

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

Algorithm 4.1 Convergent lower bounds for $\mathcal{ARTN}$.

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 $\mathcal{P}_t$:

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

   Let x^t denote an optimal solution of $\mathcal{LARTN}_t$ 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 of $\mathcal{ARTN}$ 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 4.1 *Algorithm 4.1 terminates with an optimal allocation x^* for $\mathcal{ARTN}$, together with its worst-case makespan τ^*. Furthermore, $\{\tau^t\}_t$ represents a monotonically nondecreasing sequence of lower bounds for 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 nondecreasing. Similarly, every τ^t constitutes a lower bound for 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 for $\mathcal{ARTN}$. 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 4.1 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 4.1 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 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. On one hand, this approach is likely to require many iterations since there is no prioritisation among the paths P that satisfy $\phi(x^t; P) > \tau^t$. On the other hand, in the last (Tth) iteration of Algorithm 4.1 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 2.3), this is computationally prohibitive. To alleviate both problems, we replace Step 3 of Algorithm 4.1 by the following procedure.

Algorithm 4.2 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 $\delta_i = \max\left\{\phi(x^t; \{i\}), \epsilon\right\}$. 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) If $\mathbb{I}_{P_s}^\top \delta \leq \tau^t$ or G contains less than s paths, stop: $x^* = x^t$ is an optimal allocation for $\mathcal{ARTN}$ and $\tau^* = \tau^t$ its worst-case makespan.
- (ii) If $\phi(x^t; P_s) > \tau^t$, set $\mathcal{P}_{t+1} = \mathcal{P}_t \cup \{P_s\}$, $t \to t+1$ and go to Step 2 of Algorithm 4.1.
- (iii) Otherwise, 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). $\phi(x^t; \{i\})$ represents the worst-case duration of task i over Ξ.

Depending on the problem instance, Algorithm 4.2 may certify the optimality of allocation x^t without inspecting all paths in $\mathcal{P}$. Furthermore, if ϵ is chosen to be smaller than $\min_{i \in V} \left\{\phi(x^t; \{i\}) \;:\; \phi(x^t; \{i\}) > 0\right\}/n$, 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 4.1 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 $\mathcal{O}(|E| + s\,|V|)$ [11]. The following proposition establishes the correctness of Algorithm 4.2.

Proposition 4.2 *Algorithm 4.2 terminates and either correctly concludes that x^t is an optimal allocation for $\mathcal{ARTN}$ 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 4.1 we conclude that x^t constitutes an optimal allocation for $\mathcal{ARTN}$.

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 $P \in \{P_S, \ldots, P_{|\mathcal{P}|}\}$,

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

where δ is defined in Step 3(a) of Algorithm 4.2. 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 4.1 ensures that x^t is an optimal allocation for $\mathcal{ARTN}$.

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 4.1 only provides monotonically increasing *lower* bounds for 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 for the optimal value of $\mathcal{ARTN}$. From Theorem 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 to pursue a different approach to generate upper bounds efficiently.

5 Upper Bounds

By construction, the objective value of $\mathcal{ARTN}$ exceeds $\phi(x;P)$ for all $P \in \mathcal{P}$. Due to (A1) and (A2), an approximate problem provides an upper bound on $\mathcal{ARTN}$ if its objective value exceeds $\sum_r \phi(x;P_r)$ for partitions $\{P_r\}_r$ of all $P \in \overline{\mathcal{P}}$. The granularity of these partitions trades off the quality of the upper bound with the size of the approximate problem. We start with partitions $\{\{i\}\}_{i \in P}$ of singleton blocks for $P \in \overline{\mathcal{P}}$, which imply that the worst parameter values $\xi \in \Xi$ are chosen for all network tasks individually. By coarsening the partitions, we reduce this over-pessimism and enforce the same parameter values ξ to be chosen for sets of several tasks. However, coarser partitions also increase the size of the upper bound problem since we can exploit fewer similarities between blocks from different partitions. We iteratively coarsen the partitions until each of them degenerates to a single block that corresponds to one path in $\overline{\mathcal{P}}$, in which case the approximate problem coincides with $\mathcal{ARTN}$.

We employ auxiliary graphs to represent the partitions of the task paths. For a temporal network $G = (V,E)$, these auxiliary graphs constitute directed acyclic graphs $\overline{G}_t = (\overline{V}, \overline{E}_t)$, $t \in \mathbb{N}$, with nodes $\overline{V} = V \cup \{n+1\}$ and labelled arcs $\overline{E}_t \subseteq \overline{V} \times \overline{V} \times \mathcal{P}$. $(i,j,P) \in \overline{E}_t$ represents an arc from i to j with label P. We allow for multiple arcs between i and j if they have different labels.

We associate with $\overline{G}_t$ the optimisation problem

$$\min_{\substack{x \in X, \\ y \in \mathbb{R}^{n+1}_+}} \left\{ y_{n+1} \,:\, y_j - y_i \geq \phi(x;P) \;\; \forall\, (i,j,P) \in \overline{E}_t \right\}. \qquad (\mathcal{UARTN}_t)$$

The problem assigns a variable y_i to every node $i \in \overline{V}$. The constraints ensure that y_j exceeds y_i by at least $\phi(x;P)$ time units if $(i,j,P) \in \overline{E}_t$. For $\overline{G}_t = (\overline{V}, \overline{E}_t)$ with $\overline{E}_t = \left\{(1, n+1, P) \,:\, P \in \overline{\mathcal{P}}\right\}$, $\mathcal{UARTN}_t$ is equivalent to $\mathcal{ARTN}$.

We define the set of *induced task paths* $\mathcal{P}(\overline{G}_t)$ as

$$\mathcal{P}(\overline{G}_t) = \left\{ P \in \mathcal{P} \,:\, \exists\, (i_1, i_2, P_1), \ldots, (i_R, i_{R+1} = n+1, P_R) \in \overline{E}_t \,.\, P \subseteq \bigcup_{r=1}^{R} P_r \right\}.$$

Hence, $P \in \mathcal{P}(\overline{G}_t)$ if P is contained in the union of arc labels on a path in $\overline{G}_t$ that ends at node $n+1$. Let (x,y) denote a feasible solution of $\mathcal{UARTN}_t$ and $f = y_{n+1}$ its objective value. If $P \in \mathcal{P}(\overline{G}_t)$, then $f \geq \phi(x;P)$ because there is $\{(i_r, i_{r+1}, P_r)\}_{r=1}^{R} \subseteq \overline{E}_t$ with $i_{R+1} = n+1$, $P \subseteq \bigcup_{r=1}^{R} P_r$ and

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

Here, (a) follows from non-negativity of y, (b) from the definition of $\mathcal{UARTN}_t$ and (c) from (A1) and (A2). Hence, the objective value of any feasible solution (x,y) for $\mathcal{UARTN}_t$ provides an upper bound for the worst-case makespan of allocation x with respect to the task paths in $\mathcal{P}(\overline{G}_t)$. We conclude that $\mathcal{UARTN}_t$ provides an upper bound for $\mathcal{ARTN}$ if $\overline{\mathcal{P}} \subseteq \mathcal{P}(\overline{G}_t)$.

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

$$\overline{G}_1 = (\overline{V}, \overline{E}_1) \text{ with } \overline{E}_1 = \{(i,j,\{i\}) \,:\, (i,j) \in E\} \cup \{(n, n+1, \{n\})\}. \qquad (9)$$

$\overline{G}_1$ is illustrated in Figure 3. As required, $\overline{G}_1$ constitutes an acyclic graph whose arc labels are subsets of $\mathcal{P}$. $\mathcal{UARTN}_1$ comprises one constraint for every arc $(i, j, P) \in \overline{E}_1$. Since $\overline{E}_1$ contains $|E| + 1$ arcs, $\mathcal{UARTN}_1$ constitutes a tractable optimisation problem. The following lemma establishes that the optimal value of $\mathcal{UARTN}_1$ provides an upper bound for the optimal value of $\mathcal{ARTN}$.

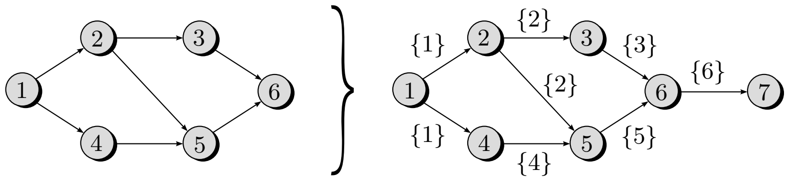


Figure 3: For the temporal network on the left, the graph on the right illustrates $\overline{G}_1$ as defined in (9).

Lemma 5.1 *$\overline{\mathcal{P}} \subseteq \mathcal{P}(\overline{G}_1)$ for $\overline{G}_1$ defined in (9).*

Proof Consider $P = \{i_1 = 1, i_2, \ldots, i_R = n\} \in \overline{\mathcal{P}}$ with $(i_r, i_{r+1}) \in E$ for $r = 1, \ldots, R - 1$. We set $i_{R+1} = n + 1$ and $P_r = \{i_r\}$ for $r = 1, \ldots, R$. By construction, $\{(i_r, i_{r+1}, P_r)\}_{r=1}^R \subseteq \overline{E}_1$ and $P = \bigcup_{r=1}^R P_r$, so that $P \in \mathcal{P}(\overline{G}_1)$. Since P was chosen arbitrarily, the claim follows. ∎

Intuitively, $\mathcal{UARTN}_1$ represents a conservative approximation of $\mathcal{ARTN}$ since it allows different parameter values ξ for different arcs in $\overline{G}_1$. $\mathcal{ARTN}$, on the other hand, requires the same parameter values ξ to be chosen within every path $P \in \mathcal{P}$. Note that $\mathcal{UARTN}_1$ and $\mathcal{ARTN}$ are equivalent if $\phi(x; P) = \sum_{i \in P} \phi(x; \{i\})$ for all $x \in X$ and $P \in \mathcal{P}$. This is the case, for example, if the task durations depend on disjoint parts of ξ that are mutually independent. 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 for the optimal value of $\mathcal{ARTN}$.

By suitably transforming the graph $\overline{G}_1$, we can tighten the initial upper bound provided by $\mathcal{UARTN}_1$.

Definition 5.1 (Replacements) *For $\overline{G}_t = (\overline{V}, \overline{E}_t)$, $t \in \mathbb{N}$, we construct $\overline{G}_{t+1} = (\overline{V}, \overline{E}_{t+1})$ via the following two types of replacements.*

1. ***Predecessor Replacement.*** *$\overline{G}_{t+1}$ results from a* predecessor replacement *of $(j, l, P) \in \overline{E}_t$ if $j \neq 1$ and*

$$\overline{E}_{t+1} = \overline{E}_t \setminus \{(j, l, P)\} \cup \bigcup_{\substack{i \in \overline{V}, P' \in \mathcal{P}: \\ (i, j, P') \in \overline{E}_t}} \{(i, l, P \cup P')\}.$$

2. ***Successor Replacement.*** *$\overline{G}_{t+1}$ results from a* successor replacement *of $(j, l, P) \in \overline{E}_t$ if $l \neq n + 1$ and*

$$\overline{E}_{t+1} = \overline{E}_t \setminus \{(j, l, P)\} \cup \bigcup_{\substack{m \in \overline{V}, P' \in \mathcal{P}: \\ (l, m, P') \in \overline{E}_t}} \{(j, m, P \cup P')\}.$$

We call $(j, l, P) \in \overline{E}_t$ *replaceable* if it qualifies for either of the two replacements. The replacements are illustrated in Figures 4 and 5. Loosely speaking, the application of a predecessor replacement in Figure 4 reduces the pessimism in the associated upper bound problem by enforcing the same value of ξ to be chosen for the tasks in P and P_1 (and, similarly, for those in P and P_2). At the same time, however, the number of arcs in the resulting auxiliary graph (and hence the size of the associated upper bound problem) increases. In the following, we assume that $\{\overline{G}_t\}_t$ with $\overline{G}_t = (\overline{V}, \overline{E}_t)$ constitutes a sequence of auxiliary graphs where $\overline{G}_1$ is defined in (9) and $\overline{G}_2, \overline{G}_3, \ldots$ result from an iterated application of Definition 5.1. One can show by induction that every $\overline{G}_t$ is acyclic and only contains arc labels that are elements of $\mathcal{P}$.

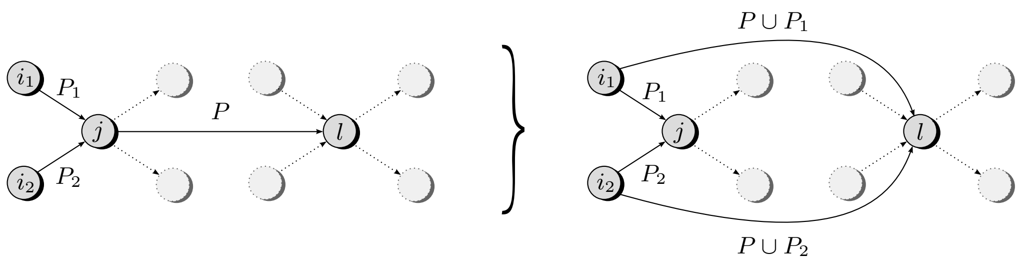


Figure 4: Predecessor replacement of (j, l, P) with two predecessor nodes.

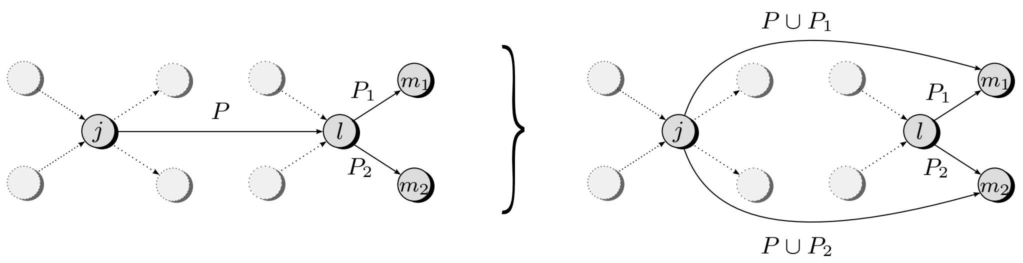


Figure 5: Successor replacement of (j, l, P) with two successor nodes.

We now show that the upper bound property of $\mathcal{UARTN}_1$ is preserved in $\mathcal{UARTN}_t$. The proof requires the following technical result.

Lemma 5.2 *If* $\{(i_r, i_{r+1}, P_r)\}_{r=1}^R \subseteq \overline{E}_t$ *satisfies* $1 \in \bigcup_r P_r$, *then* $i_1 = 1$.

Proof First we show that node 1 is a source in $\overline{G}_t$. Then we prove that if $(j, l, P_{jl}) \in \overline{E}_t$ satisfies $1 \in P_{jl}$, then $j = 1$. The assertion then follows.

By construction, node 1 is a source in $\overline{G}_1$. Definition 5.1 implies that node 1 remains a source under replacements of arcs (j, l, P_{jl}) with $j \neq 1$, as well as successor replacements of arcs $(1, l, P_{1l})$. Since Definition 5.1 precludes predecessor replacements of arcs $(1, l, P_{1l})$, node 1 is a source in any $\overline{G}_t$.

We now show by induction on t that if $(j, l, P_{jl}) \in \overline{E}_t$ satisfies $1 \in P_{jl}$, then $j = 1$. By construction, this property holds for $t = 1$. Assume now that the property holds for $\overline{G}_t$ and that $\overline{G}_{t+1}$ results from a predecessor replacement of $(j, l, P_{jl}) \in \overline{E}_t$ (an analogous argument can be constructed for successor replacements). From Definition 5.1 we know that $j \neq 1$, and the induction hypothesis implies that $1 \notin P_{jl}$. Hence, if an arc $(i, l, P_{il}) \in \overline{E}_{t+1} \setminus \overline{E}_t$ satisfies $1 \in P_{il}$, it must result from an arc $(i, j, P_{ij}) \in \overline{E}_t$ with $1 \in P_{ij}$. In this case, the induction hypothesis implies that $i = 1$, thus proving the claim. ■

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

Lemma 5.3 *If $\overline{\mathcal{P}} \subseteq \mathcal{P}(\overline{G}_t)$, then $\overline{\mathcal{P}} \subseteq \mathcal{P}(\overline{G}_{t+1})$.*

Proof Choose an arbitrary $P \in \overline{\mathcal{P}}$. By assumption, $P \in \mathcal{P}(\overline{G}_t)$, that is, there exists $\{(i_r, i_{r+1}, P_r)\}_{r=1}^{R} \subseteq \overline{E}_t$ with $i_{R+1} = n + 1$ and $P \subseteq \bigcup_r P_r$. We show that $P \in \mathcal{P}(\overline{G}_{t+1})$. Assume that $\overline{G}_{t+1}$ results from a predecessor replacement of $(j, l, P_{jl}) \in \overline{E}_t$; the proof is widely parallel for successor replacements.

If $(j, l) \neq (i_r, i_{r+1})$ for all $r \in \{1, \ldots, R\}$, then $P \in \mathcal{P}(\overline{G}_{t+1})$ is vacuously satisfied. Hence, assume that $(j, l) = (i_s, i_{s+1})$ for some $s \in \{1, \ldots, R\}$. Since 1 is the unique source of G (see Section 1) and $P \in \overline{\mathcal{P}}$, we have $1 \in P$. Lemma 5.2 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 $\left\{(i'_r, i'_{r+1}, P'_r)\right\}_{r=1}^{R-1} \subseteq \overline{E}_{t+1}$, $i'_R = n + 1$ and $P \subseteq \bigcup_r P'_r$, which ensures that $P \in \mathcal{P}(\overline{G}_{t+1})$. Since P was chosen arbitrarily, the assertion follows. ■

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

Proposition 5.1 *Let (x^t, y^t) denote an optimal solution of $\mathcal{UARTN}_t$ and $f^t = y^t_{n+1}$ its objective value. Then:*

(a) For every t, x^t is a feasible allocation for $\mathcal{ARTN}$ and f^t is an upper bound for 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 $\overline{G}_T$. For this T, x^T is an optimal allocation for $\mathcal{ARTN}$ and f^T is the worst-case makespan of x^T in $\mathcal{ARTN}$.

(c) The sequence $\{f^t\}_{t=1}^{T}$ is monotonically nonincreasing.

Proof By construction, x^t constitutes a feasible allocation for every t. Hence, assertion (a) is satisfied if $\overline{\mathcal{P}} \subseteq \mathcal{P}(\overline{G}_t)$ for every t. Employing Lemmas 5.1 and 5.3, this follows by induction on t.

As for (b), we recall that $\overline{G}_1$ is acyclic. Hence, we can relabel the nodes of $\overline{G}_1$ such that all $(j, l, P) \in \overline{E}_1$ satisfy $j < l$. Every replacement removes one arc $(j, l, P) \in \overline{E}_t$, $t = 1, 2, \ldots$, and adds less than $\left|\overline{E}_t\right|$ arcs (i, m, P') with $i \leq j$ and $m \geq l$, where one of these inequalities is strict. Since all $(j, l, P) \in \overline{E}_t$ satisfy $1 \leq j, l \leq n + 1$, there is $T \in \mathbb{N}$ such that there are no replaceable arcs in $\overline{G}_T$.

All arcs $(j, l, P) \in \overline{E}_T$ satisfy $j = 1$ and $l = n + 1$ since otherwise, further replacements would be possible. Hence, $\mathcal{UARTN}_T$ is equivalent to

$$\min_{x \in X} \max_{\substack{P \in \mathcal{P}: \\ (j,l,P) \in \overline{E}_T}} \phi(x; P).$$

Since $\overline{\mathcal{P}} \subseteq \mathcal{P}(\overline{G}_T)$, we have that $\overline{\mathcal{P}} \subseteq \{P \in \mathcal{P} : (j, l, P) \in \overline{E}_T\} \subseteq \mathcal{P}$, and therefore $\mathcal{UARTN}_T$ is equivalent to $\mathcal{ARTN}$. This proves assertion (b).

To prove (c), we first show that if (x, y) is feasible for $\mathcal{UARTN}_t$, $t \in \{1, \ldots, T-1\}$, then it is also feasible for $\mathcal{UARTN}_{t+1}$. Assume that $\overline{G}_{t+1}$ is obtained from a predecessor replacement of $(j, l, P) \in \overline{E}_t$. The argument is widely parallel for successor replacements. $\mathcal{UARTN}_{t+1}$ results from $\mathcal{UARTN}_t$ by replacing the constraint $y_l - y_j \geq \phi(x; P)$ with new constraints of the form $y_l - y_i \geq \phi(x; P \cup P')$ for $i \in \overline{V}$ and $P' \in \mathcal{P}$ with $(i, j, P') \in \overline{E}_t$. These new constraints are less restrictive, however, because

$$y_l - y_i = (y_l - y_j) + (y_j - y_i) \overset{\text{(i)}}{\geq} \phi(x; P) + \phi(x; P') \overset{\text{(ii)}}{\geq} \phi(x; P \cup P').$$

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

Proposition 5.1 provides the justification for the following procedure.

Algorithm 5.1 Convergent upper bounds for $\mathcal{ARTN}$.

1. **Initialisation.** Construct $\overline{G}_1$ as defined in (9). Set $t = 1$.
2. **Upper Bound Problem.** Determine an optimal solution (x^t, y^t) for $\mathcal{UARTN}_t$ and let $f^t = y^t_{n+1}$ denote its objective value.
3. **Replacement.** Choose a replaceable arc $(j, l, P) \in \overline{E}_t$.
   (a) <u>If there is no such arc</u>, terminate: $x^* = x^t$ is an optimal allocation for $\mathcal{ARTN}$ and $f^* = f^t$ is the worst-case makespan of x^* in $\mathcal{ARTN}$.
   (b) <u>Otherwise</u>, construct $\overline{G}_{t+1}$ by applying a replacement to arc (j, l, P), set $t \to t + 1$ and go to Step 2.

The following algorithm properties are a direct consequence of Proposition 5.1.

Corollary 5.1 *Algorithm 5.1 terminates with an optimal allocation x^* for $\mathcal{ARTN}$ and the worst-case makespan f^* of x^* in $\mathcal{ARTN}$. Moreover, $\{x^t\}_t$ represents a sequence of feasible allocations for $\mathcal{ARTN}$ and $\{f^t\}_t$ a monotonically nonincreasing sequence of upper bounds for their objective values in $\mathcal{ARTN}$.*

By combining Algorithms 4.1 and 5.1, we obtain monotonically convergent lower and upper bounds for 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 5.1 as follows.

Proposition 5.2 *Let (x^*, y^*) denote any optimal solution of $\mathcal{UARTN}_t$ and $f^* = y^*_{n+1}$ its objective value. If (A3) holds, we have:*

(a) If $y^*_l - y^*_j > \phi(x^*; P)$ for a replaceable arc $(j, l, P) \in \overline{E}_t$, then $\mathcal{UARTN}_{t+1}$ with $\overline{G}_{t+1}$ obtained from $\overline{G}_t$ by replacing (j, l, P) has an optimal value of f^*, too.

(b) If $y_l^* - y_j^* > \phi(x^*; P)$ for all *replaceable arcs $(j, l, P) \in \overline{E}_t$, then $\mathcal{UARTN}_s$ with $s > t$ and $\overline{G}_s$ obtained from $\overline{G}_t$ by any sequence of replacements has an optimal value of f^*, too.*

Remark. According to assertion (a), replacing any arc $(j, l, P) \in \overline{E}_t$ 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 5.1. Part (b) describes a condition under which x^* is the optimal allocation and f^* the optimal value of $\mathcal{ARTN}$.

Proof of Proposition 5.2 Assume that (a) is false, that is, $y_l^* - y_j^* > \phi(x^*; P)$, but there is a feasible solution (x', y') for $\mathcal{UARTN}_{t+1}$ that has an objective value smaller than f^*. From the argumentation in the proof of Proposition 5.1 (c) we know that (x^*, y^*) is feasible for $\mathcal{UARTN}_{t+1}$. Due to (A3),

$$(x^\lambda, y^\lambda) = \lambda(x', y') + (1 - \lambda)(x^*, y^*) \quad \text{for } \lambda \in (0, 1]$$

is also feasible for $\mathcal{UARTN}_{t+1}$ and has an objective value smaller than f^*. We show that for small λ, (x^λ, y^λ) is feasible for $\mathcal{UARTN}_t$, too. Since $\overline{E}_t \setminus \overline{E}_{t+1} = \{(j, l, P)\}$, we only need to show that $y_l^\lambda - y_j^\lambda \geq \phi(x; P)$. For sufficiently small λ, this follows from continuity of $\phi(\cdot; P)$ in its first component, which is a consequence of (A3), and the fact that $y_l^* - y_j^* > \phi(x^*; P)$. Since $\mathcal{UARTN}_t$ and $\mathcal{UARTN}_{t+1}$ share the same objective function, this implies that (x^*, y^*) is not optimal for $\mathcal{UARTN}_t$. Thus, our assumption is false, that is, (a) must be true.

As for (b), let us now assume that $y_l^* - y_j^* > \phi(x^*; P)$ for all replaceable arcs $(j, l, P) \in \overline{E}_t$. In this case, assertion (a) guarantees that (x^*, y^*) remains optimal for $\overline{G}_{t+1}$ if $\overline{G}_{t+1}$ results from applying one replacement to $\overline{G}_t$. Assume that $\overline{G}_{t+1}$ results from a predecessor replacement of $(j, l, P) \in \overline{E}_t$ (the proof for successor replacements is analogous). We then have

$$\begin{aligned}(y_l^* - y_i^*) = (y_l^* - y_j^*) + (y_j^* - y_i^*) &\overset{\text{(i)}}{>} \phi(x^*; P) + \phi(x^*; P') \\ &\overset{\text{(ii)}}{\geq} \phi(x^*; P \cup P') \qquad\qquad \forall\, (i, j, P') \in \overline{E}_t,\end{aligned}$$

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 $(i, l, P \cup P') \in \overline{E}_{t+1}$ as well. An iterated application of this argument shows that assertion (b) remains valid for $\mathcal{UARTN}_s$ with $\overline{G}_s$ obtained from applying any sequence of predecessor and/or successor replacements to $\overline{G}_t$. This implies that $\mathcal{UARTN}_s$ has an optimal value of f^*, and thus the claim follows. ■

$\mathcal{UARTN}_t$ may have several optimal solutions, and the conditions in Proposition 5.2 may be satisfied for some but not all of them. If an optimal solution (x^*, y^*) of $\mathcal{UARTN}_t$ does not satisfy the condition in Proposition 5.2 (a) for $(j, l, P) \in \overline{E}_t$, we can use its objective value $f^* = y_{n+1}^*$ to determine whether other optimal solutions (x', y') satisfy the condition. Indeed, this is the case if

$$\max_{\substack{x \in X, \\ y \in \mathbb{R}_+^{n+1}}} \left\{ (y_l - y_j) - \phi(x; P) \,:\, y_{n+1} = f^*,\ y_q - y_p \geq \phi(x; P')\ \forall\, (p, q, P') \in \overline{E}_t \right\} > 0. \tag{10}$$

Similarly, Proposition 5.2 (b) implies that x^* is an optimal allocation for $\mathcal{ARTN}$ if all replacement candidates $(j,l,P) \in \overline{E}_t$ satisfy (10). Unfortunately, evaluating the left-hand side of (10) is as difficult as solving $\mathcal{UARTN}_t$, and it is prohibitive to compute it for all $(j,l,P) \in \overline{E}_t$. If we fix x to x^* and optimise (10) only over y, however, the maximisation can be computed in time $\mathcal{O}(|\overline{E}_t|)$ by a combined forward and backward calculation [10]. In this case we might not identify all replacement candidates that satisfy the conditions of Proposition 5.2.

Although Proposition 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 upper bound problems for all replacement candidates. Likewise, 'first fit' approaches are unsuited due to similar reasons as in Section 4. We propose to choose a replacement for $\overline{G}_t$ 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 (10) for fixed x, this evaluation requires time $\mathcal{O}(|\overline{E}_t|)$ 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.

6 Numerical Results

We now apply our bounding technique to a circuit sizing problem with process variations. For a survey of optimisation problems in circuit design, see [5].

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, constitutes an instance of model (1). In practice, 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 can then be cast as a variant of model (1):

$$\inf_{\substack{x \in [\underline{x}, \overline{x}], \\ y \in Y(x)}} \left\{ \sum_{i \in V} A_i x_i \, : \, y_n + d_n(x) \leq T \right\} \quad \text{with } Y(x) \text{ defined in (1b).} \tag{11}$$

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), $y_n + d_n(x)$ 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:

$$d_i(x) = 0.69 \frac{R_i}{x_i} \Big(C_i^{\text{int}} x_i + \sum_{j:(i,j) \in E} C_j^{\text{in}} x_j \Big) \quad \text{for } i \in V,\, x \in X, \tag{12}$$

where R_i, C_i^{int} and C_i^{in} denote the driving resistance, intrinsic capacitance and input capacitance of gate i, respectively [5].

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, 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 [32]. We can account for such correlations by using an ellipsoidal uncertainty set:

$$\Xi = \big\{ \xi \in \mathbb{R}^n \,:\, \exists\, u \in \mathbb{R}^l \,.\, \xi = \Sigma u,\ \|u\|_2 \leq 1 \big\} \quad \text{with } \Sigma \in \mathbb{R}_+^{n \times l}. \tag{13}$$

We thus seek to optimise the following variant of $\mathcal{RTN}$:

$$\inf_{x \in [\underline{x}, \overline{x}]} \ \sup_{\xi \in \Xi} \ \inf_{y \in Y(x,\xi)} \Big\{ \sum_{i \in V} A_i x_i \,:\, y_n + d_n(x;\xi) \leq T \Big\} \tag{14}$$

For a suitable ϕ (see Section 3), this results in the following variant of $\mathcal{ARTN}$:

$$\inf_{x \in [\underline{x}, \overline{x}]} \Big\{ \sum_{i \in V} A_i x_i \,:\, \phi(x;P) \leq T \ \ \forall\, P \in \mathcal{P} \Big\}. \tag{15}$$

Again, problem (15) may be infeasible if T is chosen too small. An inspection of Sections 4 and 5 reveals that we can apply our bounding approach to problem (15) if we allow the bounds to attain values on the extended real line $\mathbb{R} \cup \{\infty\}$. A lower bound of ∞ signalises that problem (15) 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 (14):

Proposition 6.1 *For d and Ξ defined in (12)–(13), let*

$$\phi(x;P) = \mathbb{I}_P^\top d(x) + \Big\| \Sigma^\top \Big(\sum_{i \in P} \big[\nabla d_i(x)\big]^+ \Big) \Big\|_2 + \Big\| \Sigma^\top \Big(\sum_{i \in P} \big[\nabla d_i(x)\big]^- \Big) \Big\|_2, \tag{16}$$

where

$$\big[f(x)\big]^+ = \sum_{i:\alpha_i > 0} \alpha_i \prod_j (x_j)^{\beta_{ij}} \quad \textit{for} \quad f(x) = \sum_i \alpha_i \prod_j (x_j)^{\beta_{ij}}$$

and $\big[f(x)\big]^-$ defined analogously for i with $\alpha_i < 0$. If X has a tractable representation, (15)–(16) constitutes a conservative reformulation of (14) that satisfies (A1)–(A3).

Proof It follows from [32] that ϕ as defined in (16) satisfies conditions (6) 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:

$$\varphi^{+}(x,P) = \Big\| \Sigma^{\top} \Big(\sum_{i \in P} \big[\nabla d_i(x)\big]^{+} \Big) \Big\|_2 \text{ and } \varphi^{-}(x,P) = \Big\| \Sigma^{\top} \Big(\sum_{i \in P} \big[\nabla d_i(x)\big]^{-} \Big) \Big\|_2 .$$

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 $\big[\nabla d_i(x)\big]^{+} \geq 0$ for all $i \in V$. The second inequality follows from the positive homogeneity of norms and the fact that $\big[\nabla d_i(x)\big]^{-} \leq 0$ for all $i \in V$.

(A2) is satisfied if

$$\begin{aligned} \phi(x;P) + \phi(x;P' \setminus P) &= \mathbb{I}_{P}^{\top} d(x) + \varphi^{+}(x;P) + \varphi^{-}(x;P) + \\ &\quad\; \mathbb{I}_{[P' \setminus P]}^{\top} d(x) + \varphi^{+}(x;P' \setminus P) + \varphi^{-}(x;P' \setminus P) \\ &\geq \mathbb{I}_{P'}^{\top} d(x) + \varphi^{+}(x,P') + \varphi^{-}(x;P') \;=\; \phi(x;P') \end{aligned}$$

for all $x \in X$ and $P \subset P' \subseteq V$. Note that $\mathbb{I}_{P}^{\top} d(x) + \mathbb{I}_{[P' \setminus P]}^{\top} d(x) = \mathbb{I}_{P'}^{\top} d(x)$. Furthermore, 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 6.1 to determine robust gate sizes for the ISCAS 85 benchmark circuits.[3] 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 [5, 33]. 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 (11).

The data in Table 1 specifies the temporal networks corresponding to the ISCAS 85 benchmark circuits. For a circuit with $|V|$ tasks and $|\mathcal{P}|$ task paths, the path-wise model (15) can be reformulated as a geometric program with $1 + |V| + 2\,|\mathcal{P}|$ variables and $3\,|\mathcal{P}|$ constraints, see [5, 32]. Due to the choice of ϕ in (16), the Jacobian of the constraints is dense. In view of the cardinality of $\mathcal{P}$ in the benchmark circuits (see Table 1), a direct solution of (15) is prohibitive.

We now use our bounding approach to solve problem (15) 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 4.2), we limit its computation time per iteration to the time required by the upper bound.

[3]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 1: ISCAS 85 benchmark circuits.

All results are generated with CONOPT 3 on an Intel Xeon architecture with 2.83GHz.[4] We employ warm starts for the calculation of both lower and upper bounds, which significantly reduces the computational effort.

Table 2 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 2.2). We remark that the choice of Ξ and ϕ in (13) and (16) 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 lead to drastic reductions in overall circuit size.

7 Conclusion

We studied a resource allocation problem on temporal networks. Our formulation assumes that the functional relation between resource assignments and task durations is 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 for its optimal objective value, as well as feasible resource allocations whose objective values are bracketed by these bounds. We evaluated our approach on benchmark problems in circuit design.

We identify two promising avenues for future research. Firstly, some application domains (e.g., scheduling problems) impose additional restrictions on the consumption rate of resources. Such constraints result in non-convex 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.

Secondly, some application areas allow for adaptive resource allocations. Although the resulting models can be solved as multi-stage robust optimisation problems with decision-dependent structure, the available solution techniques are unlikely to scale to large problems [15]. Recent developments in the area of approximate dynamic programming [27] could constitute viable alternatives.

[4]CONOPT homepage: `http://www.conopt.com`.

circuit	**first it.**	**after 25 its.**	**after 50 its.**	**reduction**
C432	34.13% 0:03	*solved after 11 its.* 1:03		*24.48%*
C499	148.82% 0:12	12.31% 27:35	**8.96%** 128:30	*42.89%*
C880	16.78% 0:11	2.31% 2:44	**0.70%** 8:39	*11.16%*
C1355	113.16% 0:17	*solved after 24 its.* 17:31		*52.95%*
C1908	37.05% 1:17	11.37% 6:58	**6.90%** 21:06	*18.13%*
C2670	14.62% 0:51	1.61% 24:03	**1.02%** 99:35	*11.09%*
C3540	37.66% 4:22	9.19% 16:31	**7.40%** 56:06	*20.50%*
C5315	15.23% 6:56	4.30% 30:39	**2.29%** 52:37	*10.33%*
C6288	68.24% 6:33	3.40% 45:09	**2.52%** 69:08	*39.07%*
C7552	11.03% 5:54	*solved after 12 its.* 15:08		*5.01%*

Table 2: Results for the circuits from Table 1. 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 2.2).

References

[1] A. Ben-Tal and A. Nemirovski. Robust convex optimization. *Mathematics of Operations Research*, 23(4):769–805, 1998.

[2] A. Ben-Tal and A. Nemirovski. Robust solutions of uncertain linear programs. *Operations Research Letters*, 25(1):1–16, 1999.

[3] D. Bertsimas and M. Sim. The price of robustness. *Operations Research*, 52(1):35–53, 2004.

[4] D. Bertsimas and M. Sim. Tractable approximations to robust conic optimization problem. *Mathematical Programming*, 107(1–2):5–36, 2006.

[5] 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.

[6] 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.

[7] W. Chen, M. Sim, J. Sun, and C.-P. Teo. From CVaR to uncertainty set: Implications in joint chance constrained optimization. *Operations Research (in press)*, 2009.

[8] X. Chen, M. Sim, and P. Sun. A robust optimization perspective on stochastic programming. *Operations Research*, 55(6):1058–1071, 2007.

[9] 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.

[10] E. L. Demeulemeester and W. S. Herroelen. *Project Scheduling – A Research Handbook*. Kluwer Academic Publishers, 2002.

[11] D. Eppstein. Finding the k shortest paths. In *IEEE Symposium on Foundations of Computer Science*, pages 154–165, 1994.

[12] 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.

[13] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman & Co, 1979.

[14] L. El Ghaoui, F. Oustry, and H. Lebret. Robust solutions to uncertain semidefinite programs. *SIAM Journal on Optimization*, 9(1):33–52, 1998.

[15] V. Goel and I. E. Grossmann. A class of stochastic programs with decision dependent uncertainty. *Mathematical Programming*, 108(2–3):355–394, 2006.

[16] J. N. Hagstrom. Computational complexity of PERT problems. *Networks*, 18(2):139–147, 1988.

[17] W. S. Herroelen and R. Leus. Project scheduling under uncertainty: Survey and research potentials. *European Journal of Operational Research*, 165(2):289–306, 2005.

[18] R. Hettich and K. O. Kortanek. Semi-infinite programming: Theory, methods, and applications. *SIAM Review*, 35(3):380–429, 1993.

[19] R. Horst, P. M. Pardalos, and N. V. Thoai. *Introduction to Global Optimization.* Kluwer Academic Publishers, 2nd edition, 2000.

[20] T. Ibaraki. Approximate algorithms for the multiple-choice continuous knapsack problems. *Journal of the Operations Research Society of Japan*, 23(1):28–63, 1980.

[21] T. W. Jonsbraten, R. J.-B. Wets, and D. L. Woodruff. A class of stochastic programs with decision dependent random elements. *Annals of Operations Research*, 82:83–106, 1998.

[22] Y.-K. Kwow and I. Ahmad. Static scheduling algorithms for allocating directed task graphs to multiprocessors. *ACM Computing Surveys*, 31(4):406–471, 1999.

[23] 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, Austria, 1994.

[24] 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.

[25] K. Neumann. Scheduling of projects with stochastic evolution structure. In J. Weglarz, editor, *Project Scheduling: Recent Models, Algorithms, and Applications*, pages 309–332. Kluwer Academic Publishers, 1999.

[26] M. P. Papazoglou. Service-oriented computing: Concepts, characteristics and directions. In *4th International Conference on Web Information Systems Engineering*, pages 3–12. IEEE Computer Society, 2003.

[27] W. B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality.* John Wiley & Sons, 2007.

[28] A. Prékopa. *Stochastic Programming.* Kluwer Academic Publishers, 1995.

[29] A. Ruszczyński and A. Shapiro, editors. *Stochastic programming.* Elsevier Science B.V., 2003.

[30] N. V. Sahinidis. Optimization under uncertainty: State-of-the-art and opportunities. *Computers & Chemical Engineering*, 28(6–7):971–983, 2004.

[31] D. Shabtay and G. Steiner. A survey of scheduling with controllable processing times. *Discrete Applied Mathematics*, 155(13):1643–1666, 2007.

[32] 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. ACM, 2005.

[33] I. E. Sutherland, R. F. Sproull, and D. F. Harris. *Logical Effort: Designing fast CMOS Circuits*. Morgan Kaufmann, 1999.

[34] W. Wiesemann, D. Kuhn, and B. Rustem. Multi-resource allocation in stochastic project scheduling. *Annals of Operations Research (in press)*, 2008.

[35] L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang. QoS-Aware middleware for web services composition. *IEEE Transactions on Software Engineering*, 30(5):311–327, 2004.

Appendix: 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 $V = \{1, \ldots, n\}$ as follows. For each node $i \in V \setminus \{n\}$, we choose the number of immediate successors $\{1, \ldots, \lceil \rho(n-i) \rceil\}$ uniformly at random. Afterwards, we choose the indices of the successor nodes from $\{i+1, \ldots, n\}$, 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 $m \in \{2, \ldots, n\}$:

$$\begin{aligned}
\mathbb{E}(X_1) &= \left(\prod_{i=1}^{m-1} \frac{\lceil \rho(n-i) \rceil + 1}{2(n-i)} \, \frac{\lceil \rho(n-i-1) \rceil + 1 + 2(n-i-1)}{\lceil \rho(n-i-1) \rceil + 1} \right) \mathbb{E}(X_m) \\
&= \frac{\lceil \rho(n-1) \rceil + 1}{\lceil \rho(n-m) \rceil + 1} \left(\prod_{i=1}^{m-1} \frac{\lceil \rho(n-i-1) \rceil + 1 + 2(n-i-1)}{2(n-i)} \right) \mathbb{E}(X_m) \\
&= \frac{\lceil \rho(n-1) \rceil + 1}{\lceil \rho(n-m) \rceil + 1} \left(\prod_{i=1}^{m-1} \left[1 + \frac{\lceil \rho(n-i-1) \rceil - 1}{2(n-i)} \right] \right) \mathbb{E}(X_m).
\end{aligned}$$

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.