Artificial Intelligence 289 (2020) 103384

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Quantifying controllability in temporal networks with uncertainty ☆

Shyan Akmal [a,b,1], Savana Ammons [a,c,1], Hemeng Li [a,d,1], Michael Gao [a,e,1], Lindsay Popowski [a], James C. Boerkoel Jr. [a,*]

[a] Harvey Mudd College, Claremont, CA, USA
[b] Massachusetts Institute of Technology, Cambridge, MA, USA
[c] University of Illinois at Urbana-Champaign, Champaign, IL, USA
[d] Cornell University, Ithaca, NY, USA
[e] Stanford University, Stanford, CA, USA

ARTICLE INFO

Article history:
Received 1 November 2019
Received in revised form 16 July 2020
Accepted 7 September 2020
Available online 23 September 2020

Keywords:
Scheduling
Temporal planning
Controllability
Probabilistic simple temporal networks
Simple temporal networks with uncertainty

ABSTRACT

Controllability for Simple Temporal Networks with Uncertainty (STNUs) has thus far been limited to three levels: strong, dynamic, and weak. Because of this, there is currently no systematic way for an agent to assess just how far from being controllable an uncontrollable STNU is. We provide new insights inspired by a geometric interpretation of STNUs to introduce the degrees of strong and dynamic controllability — continuous metrics that measure how far a network is from being controllable. We utilize these metrics to approximate the probabilities that an STNU can be dispatched successfully offline and online respectively. We introduce new methods for predicting the degrees of strong and dynamic controllability for uncontrollable networks. We further generalize these metrics by defining likelihood of controllability, a controllability measure that applies to Probabilistic Simple Temporal Networks (PSTNs). Finally, we empirically demonstrate that these metrics are good predictors of actual dispatch success rate for STNUs and PSTNs.

1. Introduction

When tasked with making a schedule, a planner would ideally have the capability to pick exact times for every event that could occur. In practice however, autonomous agents rarely have control over all events affecting their plans. For example, imagine a chemist named Dr. V running a small experiment. She first combines 20 mL of chemical W and chemical X in a beaker. The exact amount of time it takes these chemicals to react is uncertain: all Dr. V knows is that it will take between twenty and thirty-one minutes for the reaction to finish (unfortunately, not much is known about the reaction rates of chemicals W and X). Within ten minutes of this reaction completing, she must add 20 mL of chemical Y to the mixture,

☆ This paper is an invited revision of a paper [1] which first appeared at the 2019 International Conference on Automated Planning and Scheduling (ICAPS-19).

* Corresponding author.
E-mail addresses: naysh@mit.edu (S. Akmal), sammons@hmc.edu (S. Ammons), hl2359@cornell.edu (H. Li), mgao@hmc.edu (M. Gao), lpopowski@hmc.edu (L. Popowski), boerkoel@hmc.edu (J.C. Boerkoel).
URL: https://www.heatlab.org (J.C. Boerkoel).

[1] Work on this paper was primarily completed while an undergraduate student at Harvey Mudd College.

https://doi.org/10.1016/j.artint.2020.103384

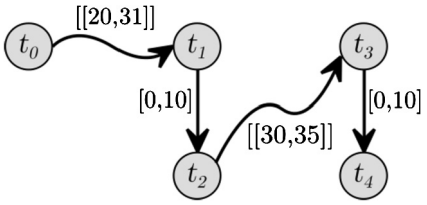

Fig. 1. An STNU representation of Dr. V's experiment.

which catalyzes a new reaction taking thirty to thirty-five minutes. Finally, Dr. V must collect a product, precipitate Z, from the solution within ten minutes of the reaction completing.

To run the experiment successfully, Dr. V needs to schedule several different events. She is able to select times for some events, such as the addition of chemical Y, but other events, such as the completion times of the reactions, are not under her control. There are multiple different strategies Dr. V can use to deal with this uncertainty in timing and ensure the experiment's success. The effectiveness of different types of scheduling strategies is related to the scheduling problem's *controllability* — how easily the experiment can be successfully executed given its inherent uncertainty. Currently, all forms of controllability discussed in the literature are discrete — they tell Dr. V whether her experiment is controllable or not, but do not tell her "how" controllable or uncontrollable the experiment is.

In this paper, we extend the notion of controllability by introducing two new continuous metrics on temporal networks: the degrees of strong and dynamic controllability [1]. These metrics are motivated by a geometric interpretation of STNUs as pairs of polytopes and characterize how far a network is from being controllable given different types of execution strategies. These metrics naturally relate to the probability that a network can be successfully dispatched. In that context, we define optimization problems for determining the probability of successful dispatch given online/offline plans, and offer approximate solutions to these problems. Finally, we show that the degrees of both strong and dynamic controllability extend naturally and effectively to PSTNs [2].

2. Background

2.1. Simple Temporal Networks

A *Simple Temporal Network* (STN) is a tuple $S = \langle T, C \rangle$, where T is the set of temporal events t_i, and C is the set of binary constraints on T [3]. Each element in C is of the form $t_j - t_i \leq c_{ij}$, for some $c_{ij} \in \mathbb{R}$. By convention, the event $t_0 \in T$ in an STN represents a fixed reference point assigned time zero. Because of this, when we say that an STN has n temporal events, we really mean that it has n time-points in addition to this reference event. A solution to an STN is an assignment of values to the timepoints t_i that satisfies all constraints. A common approach to checking if an STN is *consistent*, that it has a solution, is to apply a shortest-path algorithm to a weighted, directed graph representation of the STN called a *distance graph*. If a negative cycle is discovered during the shortest-path process, it implies that the STN is inconsistent and thus no solutions exist; otherwise the resulting distance graph is a representation of the space of solutions.

2.2. Simple Temporal Networks with Uncertainty

A *Simple Temporal Network with Uncertainty* (STNU) is an STN that explicitly models uncertainty [4]. In an STNU, the set of events T is partitioned into a set of controllable events T^c and a set of uncontrollable events T^u. An agent is allowed to schedule specific times for the events in T^c, but the times for events in T^u are determined by "Nature," a force external to the agent. For every $t_j \in T^u$, there exists a unique $t_i \in T^c$ forming a *contingent* constraint of the form $t_j - t_i \in [\ell_j, u_j]$. We use C_c to denote the set of these contingent constraints, and C_r to denote the remaining *requirement* constraints between events in T. Then an STNU is defined as a quadruple $\langle T^c, T^u, C_r, C_c \rangle$ satisfying the aforementioned properties.

In an STNU, a *decision* is an assignment of time values to each of the controllable events. A *realization* is a selection of values for contingent edges (relative start times of the uncontrollable events) assigned by Nature. The set of all possible realizations forms the *realization space* Ω for the STNU. If a decision does not violate any of the constraints between pairs of controllable events, we say the decision is *admissible*. A decision together with a realization determines a schedule σ for the network. If σ satisfies all constraints in the STNU, we call σ a *valid* schedule.

For example, Dr. V's experiment from the introduction can be modeled as an STNU. There are five events of interest: the beginning of the first reaction, the conclusion of this initial reaction, the addition of chemical Y, the completion of the subsequent reaction, and the extraction of the precipitate. If we measure time relative to the experiment's start, we can refer to these events as t_i, where i ranges from 0 to 4 respectively. Events t_0, t_2 and t_4 are under Dr. V's control, while events t_1 and t_3 are uncontrollable. This STNU representation of the experiment is depicted in Fig. 1. Events in the network are drawn as nodes. Requirement and contingent edges are drawn as straight and curvy arrows respectively. The edges are labeled by intervals indicating the lower and upper bounds of the STNU's constraints.

2.3. Controllability

An STNU is controllable when an agent has a reasonable way of working around the uncertainty in the network to schedule events. Prior research has focused primarily on detecting three types of controllability: strong, dynamic, and weak. We focus on strong and dynamic controllability, since they are important for dispatch. We omit weak controllability from our discussion because it is less relevant to STNU execution [4]. Henceforth in this paper, we use the term "uncontrollable STNU" to refer to *both* STNUs that are not strongly controllable and those that are not dynamically controllable.

An STNU is *strongly controllable* if there is a single fixed decision the agent can make that guarantees success regardless of how Nature behaves. More formally, a network is strongly controllable if there exists a decision δ such that for all $\omega \in \Omega$, the schedule determined by δ and ω is valid [4]. We call such a decision a *strong decision*. Using the example from the introduction, one can check that Dr. V's experiment is not strongly controllable because there is no fixed decision she can make to guarantee the experiment's success before it takes place.

An STNU is *dynamically controllable* if an agent can always make decisions in real-time to guarantee success. This means that there exists a strategy an agent can follow to find a valid schedule by picking times for controllable events during execution based on the values taken by all *past* events. For example, Dr. V's experiment is dynamically controllable, because she could wait until each reaction finishes and then proceed to the next step immediately. This corresponds to scheduling $t_2 = t_1$ and $t_4 = t_3$, and involves "reacting" dynamically to previously observed timepoints. This strategy guarantees the experiment's success.

We formally define dynamic controllability by following the treatment in [5]. The definition is somewhat technical, but is necessary to precisely capture the intuitions of the previous paragraph. The actions an agent makes during dispatch describes an *execution strategy*, a mapping Σ from realizations to decisions. We say a strategy is *successful* if for any $\omega \in \Omega$, the decision $\Sigma(\omega)$ together with the realization ω produces a valid schedule $\sigma = \sigma(\omega)$.

As defined so far, the strategy Σ is "omniscient," in the sense that it can pick a schedule after having access to perfect information about the realizations which will occur. To model real-world applications, the decisions Σ makes should be limited to depend only on the time values of events which have already taken place. To that end, take an arbitrary time value τ. Let $[\sigma]^{<\tau}$ denote the set of realized values of all contingent edges whose uncontrollable event was assigned a value strictly less than τ. In other words, $[\sigma]^{<\tau}$ is the collection of realized values in ω that an agent is able to observe before time τ. We also let $[\sigma]_\tau$ denote the set of controllable events executed at time τ.

Then a strategy Σ is called *dynamic* if for any realizations ω_1, ω_2 and time value τ, whenever

$$[\sigma(\omega_1)]^{<\tau} = [\sigma(\omega_2)]^{<\tau}$$

we necessarily have

$$[\sigma(\omega_1)]_\tau = [\sigma(\omega_2)]_\tau.$$

This definition captures the principle that a dynamic strategy is not allowed to make decisions by looking into the future. Finally, we say a network is dynamically controllable if it admits a successful, dynamic execution strategy.

A common technique for testing an STN for dynamic controllability is to check for conflicts. The process largely parallels the consistency check for STNs, but operates on a *labeled distance graph*, which is a distance graph that is augmented with labels capturing the implications of uncontrollable events [6]. Additionally, a set of reductions compiles the implications of both required and contingent constraints throughout the labeled distance graph. A *conflict* arises when a *semi-reducible negative cycle* is discovered as part of this reduction process [6]. If no conflict arises, the network is dynamically controllable and the resulting labeled distance graph characterizes when the agent must wait for the result of an uncontrollable event before executing particular controllable events.

2.4. Probabilistic Simple Temporal Networks

A *Probabilistic Simple Temporal Network* (PSTN) augments the STNU framework by providing probability distributions over the uncertain duration of contingent constraints [7]. In a PSTN, for every $t_j \in T^u$, there exists a unique t_i forming a contingent constraint $c_{ij} \in C^u$ of the form $t_j - t_i = X_{ij}$ where X_{ij} is a random variable with probability density function P_{ij}.

For example, $X_{01} \sim \mathcal{N}(\mu_1, \sigma_1^2)$ and $X_{23} \sim \mathcal{N}(\mu_2, \sigma_2^2)$ could be normal random variables representing t_1 and t_3's start times (relative to t_0 and t_2) in Dr. V's experiment. Then, as either σ_1 or σ_2 increase, a given dispatch strategy becomes more risky and less robust.

The distributions of contingent constraints allow planners to evaluate PSTNs in terms of *risk*, the probability a dispatch strategy might fail [8–10], and *robustness*, the likelihood that a particular network can be executed successfully [11–13]. These approaches all work by first approximating the PSTN with an STNU. This is done by establishing a risk parameter α that determines how much of the probability mass is acceptable to sacrifice when converting the PSTN. For each contingent edge of the PSTN, a new STNU edge is created with the bounds that result from sacrificing some proportion of α from the probability mass. The risk-based approaches [8–10] allow α to be a parameter that is set by the user to establish an acceptable level of risk, and proceed to optimize for various other objective functions. The Static Robust Execution Algorithm (SREA), on the other hand, attempts to find the least risky α that results in an STNU by conducting a binary search over

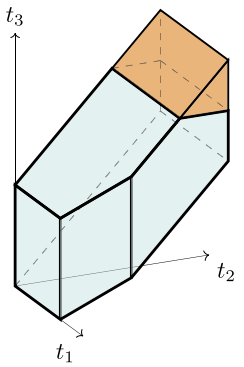

Fig. 2. The polytopes associated with network $\mathcal{S}$.

α and uses an LP that attempts to recover as much of the lost probability mass per contingent edge as possible [12]. The Dynamic Robust Execution Algorithm (DREA) attempts to make SREA more responsive to dynamic updates by reapplying SREA every time new information about contingent events is received [12].

3. Visualizing controllability

An STN with n temporal events determines a region in $\mathbb{R}^n$ as follows. Each axis corresponds to the value of a particular event, and every constraint in the STN is a linear inequality, which consequently defines a half-space. The solution space of the STN is the intersection of these half-spaces, and therefore a convex polytope. If the STN has a finite *makespan* — time during which all its temporal events must occur — then this polytope is bounded. Points within the polytope correspond to valid schedules of the STN. Previous research has leveraged this geometric perspective to determine ways of assessing a network's flexibility [14] and detect weak controllability [15]. Motivated by this, we characterize STNUs as geometric objects to inform our ideas on controllability.

3.1. Geometry of STNUs

An STNU with n controllable events and m uncontrollable events can be represented geometrically by a pair of regions $\mathcal{P}$ and $\mathcal{P}'$ in $\mathbb{R}^{n+m}$, where $\mathcal{P}' \subseteq \mathcal{P}$. The larger region $\mathcal{P}$ is the set of all schedules (including those outside the solution space) that can arise from an admissible decision. These are the situations the agent could find itself in. The smaller region $\mathcal{P}'$ is the set of all valid schedules: the situations the agent desires to end up in.

Comparing the relative locations of $\mathcal{P}$ and $\mathcal{P}'$ (in particular, the amount of $\mathcal{P}$ that "sticks out" of $\mathcal{P}'$) gives a sense of how difficult it is to control for the uncertainty present in the STNU. This geometric perspective motivates our consideration of *volume* as a way to measure uncertainty in scheduling problems. Since $\mathcal{P}'$ is the shape we get if we treat the STNU as if it were just an STN, by viewing each contingent edge as if it were a requirement edge, $\mathcal{P}'$ is a convex polytope. Similarly, $\mathcal{P}$ is the region formed by the inequalities if we ignore all requirement constraints involving an uncontrollable event, so it is also a convex polytope. As an example, consider a network $\mathcal{S}$ with one uncontrollable event t_3 and two controllable events t_1 and t_2. This network has requirement constraints $t_1 - t_0 \in [0, 8]$, $t_2 - t_0 \in [0, 12]$, $t_3 - t_0 \in [0, 16]$, and $t_1 - t_2 \in (-\infty, 8]$, as well as one contingent constraint $t_3 - t_2 \in [0, 6]$. Using the method we just described, $\mathcal{S}$ determines two polytopes, shown in Fig. 2. The entire polytope is $\mathcal{P}$, while the sub-region in the lower left is $\mathcal{P}'$.

3.2. Geometry of uncertainty & controllability

To supplement this geometric view, we associate polytopes independently to the set of uncontrollable and controllable events in the network. The regions for uncontrollables and controllables provide information on the uncertainty and controllability of the network respectively. Each uncontrollable timepoint depends on a unique contingent edge, and the possible values each contingent edge can take on form the realization space Ω defined earlier. If we define $\mathcal{U}$ to be the set of contingent intervals in the STNU, it follows that Ω is just the Cartesian product of the intervals in $\mathcal{U}$. Geometrically then, Ω is an m-dimensional hyperrectangle.

The set of all strong decisions in an STNU can be determined in polynomial time [4]. This region is determined by tightening each constraint in the STNU by considering the "worst-case" values for the contingent edges. For example, if $\mathcal{T}$ is a controllable event and $t' - t \in [l_t, u_t]$ is a contingent edge, then the original constraint $\mathcal{T} - t' \leq c$, involving the uncontrollable t', would be replaced with $\mathcal{T} - t \leq c + l_t$, a constraint between two controllables. The area determined by these tightened inequalities is the *strongly controllable region* of the network. This region is the set of *strong decisions*, and is a polytope in $\mathbb{R}^n$ since it is described completely by inequalities on the controllable events. The original STNU is strongly controllable exactly when the strongly controllable region is nonempty.

4. Beyond discrete categories of controllability

Currently, controllability is categorical — either a network is strongly or dynamically controllable or it is not. This limited categorization does not always yield sufficient information for applications. Ideally, we would like to know the likelihood of success using offline dispatch (in strong controllability) or online dispatch (in dynamic controllability) methods for a given temporal network. In practice, the problem of finding dispatch techniques that maximize controllability is infeasible, so we focus on finding good approximations. We first investigate this idea in the context of strong controllability. Our novel "degree of controllability" characterization provides a new way to understand and evaluate existing dispatch and approximate controllability approaches, particularly in the context of STNUs. Moreover, our geometric framing offers new insights that allow us to improve previous approaches for working with STNUs and provides a more rigorous definition of what it means to be "maximally strongly controllable."

4.1. Degree of strong controllability

In a strongly controllable STNU, execution is simple. The agent can guarantee success simply by finding and committing to a single strong decision. This makes strong controllability a highly desirable property for STNUs. When a network is not strongly controllable, the traditional approach has been to check for dynamic controllability and, if this check is successful, seeks out a dynamic execution strategy for dispatch. Any such strategy necessarily keeps track of when uncontrollable events occur, and is thus more complicated than pre-committing to one particular decision. Applications that require offline planning or low-overhead are not amenable to these more involved strategies. In these cases, it is useful to understand *how* controllable a network is, because it may be worthwhile to pre-commit to a single decision if the agent knows this will yield successful execution *most*, even if not all, of the time.

For instance, we observed earlier that Dr. V could ensure her experiment runs successfully by scheduling t_2 and t_4 as soon as the times for t_1 and t_3 are realized. This plan requires Dr. V to stay in the lab and monitor all reactions, so she can act as soon as they terminate. Suppose however, that Dr. V is busy the morning of the experiment. She would prefer to pick fixed times beforehand to schedule t_2 and t_4, so she could drop by the lab at only those specific times, and attend to other tasks during the rest of the experiment. As noted earlier, the experiment is not strongly controllable, so there is no such fixed strategy Dr. V could employ to guarantee success. However, if the possible execution times for t_1 and t_3 are all equally likely, then it turns out that by setting $t_2 = 30$ and $t_4 = 65$, Dr. V can guarantee that the experiment will succeed 90% of the time (i.e. we get an admissible decision if the first reaction does not take over thirty minutes). This level of controllability might be high enough that even though success is not guaranteed, Dr. V is willing to employ this simpler offline strategy.

Definition. Let S be an STNU with realization space Ω. For a given subset Ω' of Ω, we let S' denote the network with the same required constraints as S, but realization space Ω'. Then the **Degree of Strong Controllability (DSC)** for S is a continuous, 0-dimensional metric (i.e., value in $[0, 1]$) defined as the maximum possible value of

$$\frac{\mathrm{Vol}\left(\Omega'\right)}{\mathrm{Vol}\left(\Omega\right)}$$

taken over all axis-parallel hyperrectangles $\Omega' \subseteq \Omega$ such that S' is strongly controllable. Here, $\mathrm{Vol}\left(\cdot\right)$ measures the volume of a region.

STNUs and strong controllability are agnostic about how the underlying uncertainty is distributed. Similarly, the degree of strong controllability measures the maximum proportion of the realization space that an execution strategy could account for, while being completely agnostic to how that uncertainty is distributed. Networks with lower degrees are "less controllable," while networks with degrees closer to 1 have a fixed decision that are "more controllable". For example, a strongly controllable STNU has DSC equal to 1.

We demonstrate in our empirical evaluation that when uncertainty over contingent edges is uniformly distributed, DSC is a strong predictor for the likelihood of success of offline dispatch. Although in general the uncertainty encoded in an STNU does not imply any quantifiable risk, for many applications assuming a uniform prior is a common, reasonable, and sound choice in the absence of prior information [16]. Thus, in the remainder of this section, we will assume that all times within the bounds of a contingent edge of STNU are equally likely, so that capturing volume corresponds directly to probability of success. However, even if no assumptions about the distribution of uncertainty can be made, DSC remains an important measure of the controllability of a given STNU, albeit we would expect it to be a weaker predictor of the likelihood of success. In Section 6, we show how we can adapt this definition in the case where we have explicit information about the distribution of uncertain events.

In the definition of DSC, Ω' is a hyperrectangle. This means it is the Cartesian product of m intervals, each of which is necessarily a subinterval of one of S's contingent intervals (otherwise Ω' would not be a subset of Ω). We denote the collection of these intervals by $\mathcal{U}'$, and observe that this collection uniquely determines Ω' (and vice-versa).

The DSC of an STNU is a lower bound on the maximum probability of obtaining a valid schedule for the given network by committing to an admissible decision before any events have executed. This is because there always exists a decision

that can be paired with every realization in Ω' (which is a subset of Ω) to create a valid schedule. So, by comparing Ω''s volume to Ω's volume, we are calculating the *proportion* of time that a random realization yields a valid schedule when paired with an admissible decision.

In the following subsection, we introduce an efficient method that both approximates the DSC (dealing with the *optimization* problem) and finds a fixed schedule that is guaranteed to succeed with probability greater than or equal to the estimated DSC value (addressing the corresponding *decision* problem).

4.2. Estimating DSC

To compute the DSC, it suffices to find an Ω' of maximum volume such that S' is strongly controllable. The strong controllability of S' can be verified by checking whether a set of linear constraints are satisfied [4]. However, solving this general problem is difficult, since maximizing a nonlinear polynomial over linear constraints is NP-hard [17]. Thus we propose our approximation method for finding the optimal Ω'.

4.2.1. Degree of Strong Controllability Linear Program (DSC-LP):

Our DSC-LP is adapted from the maximum subinterval problem [18]. The goal of DSC-LP is to find a set of contingent subintervals $\mathcal{U}'$ that approximately maximize the volume of Ω' and a fixed decision that ensures success in the associated STNU S'. Given an STNU, $S = \langle T^c, T^u, C_r, C_c \rangle$, a set of maximal subintervals for S can be obtained by solving the following LP:

$$\text{minimize: } \sum_{t \in T^u} \frac{\epsilon_t^- + \epsilon_t^+}{u_t - l_t}$$

$$\begin{aligned}
\text{subject to: } & t^- \leq t^+ && \forall t \in T^u && (1)\\
& t^- = t^+ && \forall t \in T^c && (2)\\
& t^+ - t'^- \leq c && \forall (t - t' \leq c) \in C_r && (3)\\
& t^- - t'^- = l_t + \epsilon_t^- && \forall (t - t' \geq l_t) \in C_c && (4)\\
& t^+ - t'^+ = u_t - \epsilon_t^+ && \forall (t - t' \leq u_t) \in C_c && (5)\\
& \epsilon_t^-, \epsilon_t^+ \geq 0 && \forall t \in T^u && (6)\\
& t_0^- = t_0^+ = 0 && && (7)
\end{aligned}$$

Following convention [18], we introduce two variables t^- and t^+ for each event t, representing the lower bound and upper bound of the time interval in which event t occurs and constrain these variables so that they respect all requirement constraints (lines 1-3,7). Our LP differs from the previous maximum subinterval approach of [18] in three key ways. First, it allows us to modify the upper and lower bounds of contingent intervals by different amounts (lines 4-6). We do this by introducing two variables ϵ_t^- and ϵ_t^+ for every contingent constraint $t - t' \in [l_t, u_t]$, which represent the amount the lower and upper bounds of the original contingent interval are tightened by to guarantee strong controllability. The resulting contingent subinterval becomes $t - t' \in [l_t + \epsilon_t^-, u_t - \epsilon_t^+]$. Second, our LP sets $t^- = t^+$ for all $t \in T^c$. By doing this, we are guaranteed to have a specific schedule returned by the LP. The third and final way our LP differs from the original is that it uses a new objective function, which sums the amount each contingent interval is decreased by, normalized by its original length. Let $\ell_t = u_t - l_t$ denote the uncontrollable timepoint t's contingent interval length, and let $\epsilon_t = \epsilon_t^+ + \epsilon_t^-$ be the amount that interval is shrunk by. Then, the objective function is equal to

$$\sum_{t \in T^u} \frac{\epsilon_t}{\ell_t}.$$

This choice is motivated by considering the amount of volume lost

$$f(\boldsymbol{\epsilon}) = \prod_{t \in T^u} \ell_t - \prod_{t \in T^u} (\ell_t - \epsilon_t)$$

in going from Ω to Ω' as a function of the ϵ_t variables, and setting the objective function equal to a constant multiple of the gradient ∇f evaluated at $\boldsymbol{\epsilon} = \vec{0}$. Thus, minimizing the objective function corresponds to minimizing the first order approximation of the amount of volume lost by shrinking the contingent intervals.

Since Ω and Ω' are the Cartesian products of elements in $\mathcal{U}$ and $\mathcal{U}'$ in this case, the volumes of these regions are just the product of lengths of the intervals in $\mathcal{U}$ and $\mathcal{U}'$ respectively. Our LP yields a choice of $\mathcal{U}'$, so the solution to our LP provides an approximate value for DSC. Moreover, the values taken on by t^+ variables, for $t \in T^c$, in the optimal solution of DSC-LP determine an admissible decision for the network that is a strong decision for the S'. We call a decision returned

by our LP in this way a **strong-LP decision** for the network. If a network has high DSC, the strong-LP decision is likely a good strategy for this network.

There are other natural choices of LPs one could use to approximate DSC. For example, we could maximize the sum of the size of subintervals captured by the strong decision [18], minimize the maximum amount of that a single contingent interval is squeezed, or maximize the minimum length over all contingent subintervals. An empirical comparison of these methods, included in Subsection 7.1.2, shows that the DSC-LP provides better approximations (i.e., higher success rates) than these other approaches. In particular, the use of volume as probability, motivated by the relationship between Ω, $\mathcal{P}$, and $\mathcal{P}'$, seems more effective than looser heuristics and metrics like flexibility for estimating likelihood of successful dispatch. In Section 6, we discuss how these ideas can be extended to PSTNs, which allows us to take advantage of probability density information to find strong decisions that can be dispatched with high likelihood of success.

5. Degree of Dynamic Controllability

Although strong controllability is valuable because it allows an agent to rely on a static schedule, dynamic controllability applies to a wider range of STNUs (since all strongly controllable networks are dynamically controllable, but not vice-versa). In cases where an STNU is not dynamically controllable, an agent may still wish to attempt to dispatch the schedule. For, even when success is not guaranteed, if the probability of successful execution is high enough, dispatching on an uncontrollable network may be worthwhile. To assess how safe it is to attempt dispatch, the agent will need some measure of *how* dynamically controllable the network is. This motivates a metric analogous to DSC in the context of dynamic controllability.

Definition. Let S be an STNU with realization space Ω. For a given subset Ω' of Ω, we let S' denote the network with the same required constraints as S, but realization space Ω'. Then the **Degree of Dynamic Controllability (DDC)** for S is defined to be the maximum possible value of

$$\frac{\text{Vol}\left(\Omega'\right)}{\text{Vol}\left(\Omega\right)}$$

taken over *all* $\Omega' \subseteq \Omega$ such that S' is dynamically controllable and where $\text{Vol}(\cdot)$ measures the volume of a region.

The key distinction between the DDC and the DSC is that Ω' is allowed to be any measurable subset of Ω. In particular, Ω' is not constrained to be an axis-parallel hyperrectangle. This means that the network S' is not necessarily an STNU (which always has a hyperrectangular realization space) in the traditional sense, but rather an *STNU with Generalized Realization Space (STNU-GR)*. We define this new type of network in Subsection 5.2, where we offer some intuition for why these objects are a natural framework for studying dynamic dispatch.

In the remainder of this section, we show how we can lower bound the DDC by solving a particular type of temporal relaxation problem. This is done using our Optimal DC Relax (ODC-Relax) strategy introduced in Subsection 5.1, which optimally relaxes a network's constraints until it is dynamically controllable. From here, we compute the volume of Ω' simply by multiplying the weights of the relaxed contingent intervals, as we did when computing the DSC. This is equivalent to approximating the volume of the optimal Ω' using an inscribed hyperrectangle. In general this method underestimates the true maximum volume, since it ignores the possibilities captured by a more general view of an STNU's realization space.

To get a better approximation, we formally define STNU-GRs in Subsection 5.2 and employ them in Subsection 5.3, where we demonstrate how removing the condition that Ω' be a hyperrectangle yields significantly more accurate estimates of dispatch success. Here, we improve the predicted DDC probability via a normal approximation technique, and then conclude by discussing related approaches. The examples and approximation method introduced in Subsection 5.3 demonstrate why allowing for non-hyperrectangular DC realization space captured by the STNU-GR is actually more useful for dispatch.

5.1. Optimal dynamic controllability relaxations

Dynamic controllability of STNUs can be checked in polynomial time [19]. The fastest current algorithms for determining DC search for a *conflict*: a series of constraints that, when taken together, cannot simultaneously be satisfied by any dynamic execution strategy. As mentioned above, previous work characterized conflicts as a special type of negative cycle occurring in the labeled distance graph of an STNU [6].

Bhargava et al. [20] build off this work and introduce a cubic time algorithm to extract conflicts from networks. They identify conflicts with the aim of solving *temporal relaxation problems*. A relaxation problem involves taking an uncontrollable network and determining how to relax the network's constraints in a minimal fashion to produce a new, controllable network.

In our context, where relaxing requirement constraints is not permitted, an STNU conflict consists of a set of contingent edges, together with a *resolution constant* κ that indicates the total amount the contingent edges must be shrunk by to resolve the conflict (the latter can be obtained from an identified conflict). We now present a solution (presented as Algorithm 1) to our version of the relaxation problem, where only contingents can be relaxed (by shrinking their intervals).

If we have an STNU with only one conflict, our solution is optimal in the sense that it shrinks intervals to minimize the amount of volume of Ω lost while still resolving the conflict.

Suppose the conflict consists of intervals $I_1, I_2, \ldots, I_p$ with respective lengths $\ell_1 \leq \ell_2 \leq \cdots \leq \ell_p$. We want to shrink each interval I_j to an interval of length ℓ'_j such that the sum of the ℓ'_js is less than or equal to the quantity

$$\ell' = \left(\sum_{j=1}^{p} \ell_j\right) - \kappa$$

since this is exactly what it means for the conflict to be resolved. Thus, resolving the conflict while maximizing the volume of the reduced realization space is equivalent to finding values $\ell'_j \in [0, \ell_j]$ with $\sum_{j=1}^{p} \ell'_j = \ell'$ that maximize the product $\prod_{j=1}^{p} \ell'_j$. This optimization problem has an analytic solution, which we now describe. Let q be the smallest index such that

$$\ell' \leq \ell_1 + \ell_2 + \cdots + \ell_{q-1} + (p - q + 1)\ell_q. \tag{8}$$

Then taking $\ell'_j = \ell_j$ for $j < q$ and

$$\ell'_j = \frac{\ell' - \ell'_1 - \cdots - \ell'_{q-1}}{p - q + 1}$$

for $j \geq q$ solves the maximization problem. In this solution, the largest contingent intervals are all shrunk to the same size, while smaller intervals are left alone. Intuitively, the solution reduces the realization space to make it look more like a hypercube.

Algorithm 1: Optimal Relax (ODC-RELAX) Strategy.

```
Input : conflicts, a sorted list of contingent intervals in the detected STNU conflict
Input : κ, resolution constant
Var   : p, length of conflicts
Var   : ℓ, a list in which ℓ[j] is the length of the contingent interval, conflicts[j]
Initialization :
1   relaxations ← {};
2   S ← sum(ℓ[1...p]) − κ ;
3   q ← None;
    OptimalRelax
4   for j ← 1 to p do
5       s ← sum(ℓ[1..j]) + (p − j + 1) ∗ ℓ[j];
6       if s ≥ S then
7           q = j;
8           break;
9   A ← (S − sum(ℓ[1..q]))/(p − q + 1);
10  for j ← 1 to p do
11      edge ← conflicts[j];
12      relaxations[edge] ← j < q? ℓ[j] : A;
13  return relaxations;
```

We prove optimality by induction on q. When $q = 1$, all of the ℓ_j are less than or equal to ℓ'/p by definition of q. Now, the Arithmetic Mean-Geometric Mean (AM-GM) inequality asserts that for a list of nonnegative reals with a fixed sum, the product is maximized when all the numbers are equal. So, setting $\ell'_j = \ell'/p$ satisfies the constraints and maximizes the product in this scenario.

Now take some fixed $q > 1$, and suppose the result is correct for all integers less than q. We claim there exists an optimal choice of the ℓ'_j with $\ell'_1 = \ell_1$. Suppose to the contrary this were not true. Then, take an optimal selection of values $\ell'_1, \ell'_2, \ldots, \ell'_p$ (this exists because the domain of the ℓ_js is compact). If $\ell'_1 < \ell_1$ in this assignment, then Equation (8) implies that there must exist an index k with $\ell'_k \geq \ell_1$. It follows that there exists a small positive ϵ for which replacing ℓ'_1 with $\ell'_1 + \epsilon$ and ℓ'_k with $\ell'_k - \epsilon$ preserves the sum of the ℓ'_j while increasing their product. This contradicts the optimality of our initial choice, so $\ell'_1 = \ell_1$ as claimed. Now the problem reduces to maximizing $\prod_{j=2}^{p} \ell'_j$ subject to the constraint that $\sum_{j=2}^{p} \ell'_j = \ell' - \ell_1$. Finally, using the inductive hypothesis proves that our claimed solution is optimal.

This algorithm yields an optimal relaxation for networks with a single conflict. In the case of multiple conflicts, our approach lends itself to a greedy algorithm, where we pick conflicts and relax them one at a time using the above procedure. The resulting algorithm might not be optimal, but functions as a simple yet useful heuristic for resolving conflicts while minimizing probability mass lost through relaxation.

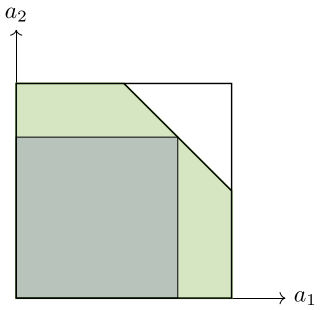

Fig. 3. The realization space for $\mathcal{S}'$. The inner box, the realization space of the relaxed network, underestimates the volume of realizations that lead to valid schedules.

5.2. STNU with Generalized Realizations

A *Simple Temporal Network with Uncertainty and Generalized Realization Space* (STNU-GR) is an STNU that permits dependence between contingent constraints. Just like in an ordinary STNU, an STNU-GR has a set of temporal events T partitioned into a set of controllables T^c and a set of uncontrollables T^u. As usual, we may have some set of binary requirement constraints C_r between events in T.

This generalized network differs in that times for uncontrollable events are determined in a more complicated fashion. Suppose T^u consists of m uncontrollable events $t_1, t_2, \ldots, t_m$. Then each $t_j \in T^u$ has a unique controllable $t'_j \in T^c$ associated with it. We let $\delta_j = t_j - t'_j$ denote the time elapsed between t_j and t'_j. The values of the δ_j are determined not by contingent intervals, but instead by a realization space Ω that is part of the data of an STNU-GR. The space Ω is a measurable subset of $\mathbb{R}^m_{\geq 0}$ with finite volume, where $\mathbb{R}_{\geq 0}$ denotes the set of nonnegative reals. Nature samples a point uniformly at random from Ω, and then sets δ_j equal to the jth coordinate of the point sampled, for $j = 1, 2, \ldots, m$. This then determines when the uncontrollable t_j occurs, according to the equation $t_j = t'_j + \delta_j$. Combining this information altogether, an STNU-GR is defined as the quadruple $\langle T^c, T^u, C_r, \Omega \rangle$.

Although the details of this definition are more involved than the definition for a normal STNU, making Ω part of the explicit data for the network (rather than something that indirectly comes out of a product of contingent intervals) affords us more power in modeling dispatch situations. During dispatch, a scheduler is interested in the space of realizations that would lead to success, given the uncontrollable timepoints that have already been realized. As we will see in Subsection 5.3, this space does not have to be a hyperrectangle, which makes it difficult to work with in the usual STNU framework. However, when viewed in the context of STNU-GRs, we can easily keep track of this information by viewing it as a realization space corresponding to a subnetwork. Consequently, STNU-GRs are a useful extension of STNUs that helps schedulers identify which realizations lead to success during dispatch.

5.3. Estimating DDC

Although solving relaxation problems is useful, we are also interested in accurately computing the probability of successful dispatch with a dynamic execution strategy. Previous research has attempted to measure this likelihood by first relaxing to strongly/dynamically controllable networks. As noted in [21] however, the volume of the resulting hyperrectangular realization space in a relaxed network does not accurately estimate the relevant probability; it merely bounds it from below. Indeed, this relaxation approach underestimates the probability too severely to estimate DDC.

For example, consider an STNU $\mathcal{S}'$ with two uncontrollables t_1 and t_3, and one controllable event t_2. This network has requirement constraints $t_1 - t_2 \leq 0$ and $t_3 - t_0 \leq 3$, along with contingent constraints $t_1 - t_0 \in [0, 2]$ and $t_3 - t_2 \in [0, 2]$. Network $\mathcal{S}'$ is not dynamically controllable, but by inspection we see that the optimal execution strategy schedules $t_2 = t_1$. In this case, if $a_1 = t_1 - t_0$ and $a_2 = t_3 - t_0$ denote the realized lengths of the contingent edges, then as depicted in Fig. 3 our schedule succeeds exactly when $a_1 + a_2 \leq 3$, which happens 87.5% of the time. In contrast, if we wanted to relax $\mathcal{S}'$ to get a dynamically controllable network while maximizing volume, both intervals for the contingent edges would shrink from $[0, 2]$ to $[0, 3/2]$. This shrunk realization space takes up 56.25% of the original volume, which is significantly smaller than the true probability of succeeding during dispatch.

Interestingly enough, this underestimate still holds in higher dimensions. We can generalize our construction of $\mathcal{S}'$ by considering the "contingent-chain" STNU $\mathcal{S}^{(k)}$ which has $k \geq 2$ uncontrollables $t_1, t_3, \ldots, t_{2k-1}$ along with $(k - 1)$ controllable events $t_2, t_4, \ldots, t_{2k-2}$. This network has $(k - 1)$ requirement constraints $t_{2j-1} - t_{2j} \leq 0$ for $j = 1, 2, \ldots, k - 1$ that order the events, k contingent constraints $t_{2j-1} - t_{2j-2} \in [0, 2]$ for $j = 1, 2, \ldots, k$, and one final requirement constraint $t_{2k-1} - t_0 \leq 2k - 1$ that sets a makespan of $2k - 1$ for the network. Note that the network $\mathcal{S}^{(2)}$ under this definition is precisely the previous example $\mathcal{S}'$.

If we let $a_j = t_{2j-1} - t_{2j-2}$ for $j = 1, 2, \ldots, k$ denote realized lengths of contingent edges (all nonnegative reals less than or equal to 2), then the schedule succeeds precisely when

$$a_1 + a_2 + \cdots + a_k \leq 2k - 1.$$

It follows, using the formula for the volume of a simplex, that the maximum probability of successful dispatch is

$$1 - \frac{1}{2^k k!}$$

which tends to 1 rapidly as k gets large. On the other hand, relaxing $\mathcal{S}^{(k)}$ and achieving dynamic controllability while minimizing volume lost corresponds to shrinking each contingent interval to have length $2 - (1/k)$. The resulting realization is a hypercube that takes up

$$\left(\frac{2 - \frac{1}{k}}{2}\right)^k = \left(1 - \frac{1}{2k}\right)^k$$

of the original volume. This quantity monotonically increases as k gets larger and tends to $e^{-1/2} \approx 0.6$. Again, the relaxation yields an estimate close to 60%, a dramatic underestimate of the true probability of success, which approaches 100% rapidly as k increases.

For this particular family of STNUs, we can explain this phenomenon by a simple geometric reason: inscribed hyperrectangles are bad at approximating volumes of simplices. This observation is in turn related to "concentration of measure" results in high dimensions, which suggests another avenue in which geometry could be helpful for proving results in scheduling.

Motivated by these examples, we leverage the notion of conflicts to directly approximate the probability of successful dispatch using the **normal approximation to DDC**. Suppose we have an STNU with a single conflict preventing it from being dynamically controllable. Carry over the notation from the previous section and additionally let a_j denote the difference between the realized value during dispatch and the lower bound of the contingent interval I_j for $j = 1, 2, \ldots, p$. Then if the inequality

$$\sum_{i=1}^{p} a_i \leq \ell'$$

holds, the conflict is "avoided" in the sense that the observed realization could have come from a controllable STNU.

This idea, that dynamic execution strategies succeed if conflicts are avoided during dispatch, is the crucial insight that allows us to estimate the DDC. We may view each a_j as an independent random variable drawn from some distribution over $[0, \ell_j]$. Then the Central Limit Theorem implies that the sum of the a_j (the left hand side of the above inequality) can be approximated as coming from a normal distribution $\mathcal{N}(\mu, \sigma^2)$. If we use our assumption from earlier that contingent edges have uniformly distributed uncertainty, then the a_js are independent uniform random variables and the parameters for the approximating normal distribution are given by $\mu = (\ell_1 + \cdots + \ell_p)/2$ and $\sigma^2 = (\ell_1^2 + \cdots + \ell_p^2)/12$, the sums of the means and variances of the a_js respectively. Thus the probability the above inequality holds can be computed using the standard normal CDF Φ. For uncontrollable STNUs with a single conflict, the normal approximation accurately estimates the probability of successful dispatch. In the case of multiple conflicts we use the normal approximation to estimate the probabilities of avoiding each conflict separately, and then multiply the obtained probabilities. Taking this product implicitly treats each conflict as if it were independent of the others (no sharing contingent edges among conflicts). Because conflicts are not generally independent, we expect this technique to behave worse in networks with many conflicts. However, as detailed in our empirical evaluation, the normal approximation is still useful in these cases.

5.4. Related approaches

The ODC-Relax algorithm builds on top of work that uses a mixed integer LP to solve a broad range of relaxation problems related to dynamic controllability [21]. Our approach tackles a specific variant of their general formulation, but in this specialized scenario obtains an analytic solution that is optimal in the case of a single conflict, and can be computed much more efficiently in $O(p \log p)$ time for an STNU with p conflicts. Moreover, ODC-Relax can naturally be used as a subroutine in the conflict-directed search methods in [8] and the RelaxSearch method of [20] to augment existing algorithms.

Although work done by [22] also evaluates the probability of successful dispatch in a dynamic controllability setting, to the best of our knowledge, our normal approximation for DDC is the first known approach for estimating the rate of online dispatch without appealing to relaxations and nonlinear solvers. Because our definition of DDC does not constrain the shape of Ω', it effectively captures the notion of robustness geometrically. Note that in our framework, we assume agents in networks are mandated to satisfy all requirement constraints, so that violating any one constraint is just as bad as violating all constraints simultaneously. This is consistent with the methodology of [22], but contrasts with previous work [21,8,20] that allows both requirement *and* contingent constraints to be relaxed, and research [23] that maximizes the number of constraints satisfied under preference schemes.

6. Generalizing degree of controllability to probabilistic simple temporal networks: the likelihood of controllability

In this section, we extend the concept of Degree of Controllability to PSTNs. This provides both opportunities and challenges. The main opportunity is that PSTN contingent edges have probability density functions (PDFs), which provide richer sources of information for understanding when events might happen. However, this also presents the first challenge, since we now have to worry about optimizing the total density captured along each contingent edge, rather than merely optimizing the volume. This makes the optimization problem much more complex, even in low-dimensional cases.

The second challenge that PSTNs introduce is that the PDFs governing contingent edges may be unbounded (e.g. if the realization of the edge is drawn from a normal distribution). For certain networks, this means it is inherently impossible to control for *all* uncertainty, regardless of how much slack is introduced into the network. As a simple example, consider a network with a single activity whose duration is dictated by a distribution with a long, unbounded tail. If this PSTN contains a makespan constraint imposing a deadline by which the activity must finish, there is no way to guarantee success, regardless of how generously the deadline is set—only removing the deadline altogether can guarantee success. This means that there are entire classes of PSTNs that are inherently uncontrollable in the traditional, discrete sense of controllability.

We now outline how we can extend the DSC and DDC definitions to PSTNs to create a more useful, fine-grained metric. At a high-level, the definitions translate in a relatively straight-forward way: we exchange the n-dimensional *volume* optimizations of STNUs (Sections 4 and 5) for n-dimensional *mass* optimizations in the case of PSTNs. Thus, a PSTN whose contingent edges only have bounded distributions would end up performing an equivalent optimization to an STNU that lacks information about the distribution of uncontrollable events.

6.1. Likelihood of strong controllability

PSTNs have richer realization spaces than STNUs. Recall that the realization space of an STNU can be viewed as the product of its contingent intervals. Consequently, this space is always a bounded, axis-parallel hyperrectangle. In contrast, each contingent constraint in a PSTN is determined by some PDF P_{ij}. We can associate an interval to such a constraint by taking the minimal interval that contains the support of P_{ij}. Then, as in the case of STNUs, we can define the realization space Ω for a PSTN S to be the product of these contingent intervals. However, now the set Ω comes with a density function μ, defined as the product measure of the P_{ij}s on the contingent intervals.

Consequently, the realization space (Ω, μ) of a PSTN is a possibly unbounded space, Ω, together with a density function, μ. The total mass of Ω with respect to μ is exactly 1, because μ is a PDF.

Definition. Let S be a PSTN with realization space (Ω, μ). For a given subset Ω' of Ω, we let S' denote the network with the same required constraints as S, but realization space Ω'. Then the **Likelihood of Strong Controllability (LSC)** for S is defined as the maximum possible value of

$$\int_{\Omega'} \mathrm{d}\mu$$

taken over all axis-parallel hyperrectangles $\Omega' \subseteq \Omega$ such that S' is strongly controllable.

Note that the integral in the above definition is precisely the probability mass of Ω', so that LSC really is the natural generalization of DSC for PSTNs. While the mass of the realization space is 1 by definition, finding the axis-parallel hyperrectangle that captures the greatest total probability mass is significantly more difficult. Calculating the mass of even a single hyperrectangle requires computing a definite integral for each component PDF, and then taking the product of the probabilities captured along each dimension. Solving the corresponding optimization problem is generally intractable in practice [11,12]. However, like DSC, LSC also provides a useful tool for predicting the likelihood of successful execution once a strongly controllable strategy has been identified.

6.1.1. Estimating LSC

Interestingly, although no formal, mathematically-precise definition of the Likelihood of Strong Controllability previously existed to our knowledge, all known prior approaches for finding robust execution strategy within a PSTN [8–10,12,13] happen to offer approximate approaches for finding the LSC. This provides external validation that the LSC is a useful and natural metric for scheduling. It also points to the benefit of framing this problem in terms of controllability and points to future potential for providing a unifying theory and new insights that may help us better dispatch PSTNs.

As discussed in Section 2.4, these previous approaches work by finding a strong decision schedule using an STNU that approximates the input PSTN. In a similar spirit, we can use the DSC-LP to find strongly controllable schedules that approximate LSC. The approach is a simple, two step approach. First, approximate the input PSTN as an STNU using the same risk-based strategy as above using a pre-specified α. Second, if the STNU approximation of the PSTN is strongly controllable, we simply return it. If not, run DSC-LP on the resulting STNU to find the strongly controllably STNU that minimally squeezes the contingent edges if one exists (i.e., if the STNU is feasible). We call this the LSC-LP approach.

Algorithm 2: Min-Loss DC.

Input : A PSTN $S = \langle V, E^P \rangle$ and a risk tolerance α
Output: An approximately optimal DC STNU
$E^U \leftarrow$ Extract-STNU-Edges(E^P, α);
if not DC-Check$(\langle V, E^U \rangle)$ **then**
  $E^U \leftarrow$ ODC-Relax$(\langle V, E^U \rangle)$;
return $\langle V, E^U \rangle$;

6.2. Likelihood of Dynamic Controllability

As we did for DSC, we extend DDC to PSTNs with the following definition.

Definition. Let S be an PSTN with realization space (Ω, μ). For a given subset Ω' of Ω, we let S' denote the network with the same required constraints as S, but realization space Ω'. Then the **Likelihood of Dynamic Controllability (LDC)** for S is defined to be the maximum possible value of

$$\int_{\Omega'} \mathrm{d}\mu$$

taken over *all* $\Omega' \subseteq \Omega$ such that S' is dynamically controllable.

Unlike with LSC, we are unaware of any work that attempts to use dynamic controllability to estimate the maximum likelihood of successful online dispatch for PSTNs. This makes the LSC definition and the Min-Loss DC approach for approximating LSC the first attempts to define and solve this problem. However, like LSC, we can borrow insights from our work on applying DDC to STNUs to inform our estimates of LDC.

6.2.1. Optimal DC relaxations for PSTNs: Min-Loss DC

We start by adapting our earlier ODC-Relax approach to PSTNs. The Min-Loss DC algorithm, summarized in Algorithm 2, works in a two-step process. First, it extracts a bounded contingent edge from each probabilistic contingent edge in the original PSTN using a procedure we call Extract-STNU-Edges(E^P, α). Extract-STNU-Edges takes in a set of probabilistic edges and a risk budget α, and returns a set of STNU edges, where for each edge, the probability that the realized value will fall between the bounds is guaranteed to be at least $(1 - \alpha)$ by truncating $\frac{\alpha}{2}$ of the probability density from each tail of the distribution. Note that because we truncate equal mass from each tail, this procedure naturally somewhat accounts for distributions with various skews, kurtoses, or modalities.

Second, if the resulting STNU is DC, we return it. If it is not DC, then we apply the ODC-Relax algorithm described in Section 5.1, which returns the STNU that corresponds to minimally "relaxing" the problem, in this case by squeezing the intervals associated with the contingent edges involved in the conflict. The terminology "relax" is used to reflect that we reduce the amount of uncertainty we need to control (by contracting, rather than expanding, contingent interval bounds), which makes the controllability problem *easier* to solve. Min-Loss resolves conflicts one at a time, so it only sacrifices probability mass on the side of the uncertain distribution that is needed. The ODC-Relax algorithm is optimal in the case of a single relaxation for uniform distributions, but optimality is no longer guaranteed when we have different probability distributions or when multiple relaxations must be performed.

6.2.2. Estimating LDC

Similarly to how we adapted our ODC-Relax approach to work for PSTNs, we can adapt the insights that we used to improve our DDC estimates using STNUs with Generalized Realizations to also improve our LDC estimates for PSTNs. Our approach for estimating DDC for STNUs involves extracting conflicts. Because the realization space for PSTNs can be unbounded, it is not as obvious what qualifies as a "conflict" in the PSTN setting. Therefore, we adopt the same risk-based approximation strategy as we have elsewhere. We first approximate the input PSTN as an STNU, denoted S_α, with a pre-specified risk factor α. Then we extract all conflicts in S_α and estimate the maximum likelihood of successful execution (i.e. degree of controllability) p_α by applying our normal approximation method. Note that our normal approximation method can naturally extend to PSTNs since the form of the Central Limit Theorem we apply works as long as the distributions of different contingent edges are independent. Then the approximate LDC becomes $(1 - \alpha)^m p_\alpha$, where m is the number of uncontrollable events in the network.

6.3. Risk and likelihood of controllability

As mentioned, all current methods for estimating LSC and LDC currently approximate the probabilistic contingent edges of a PSTN using interval-based STNU-style contingent edges that each sacrifice some probability density independently according to a risk tolerance, α. By doing so, the total risk accrued across the network, α^S, is $1 - (1 - \alpha)^m$, where m

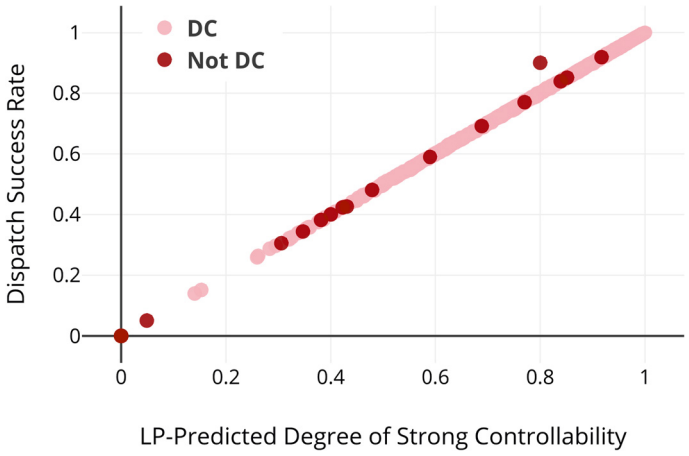

Fig. 4. Dispatch Success Rate *vs.* our DSC-LP Prediction.

is the number of contingent edges. Thus, an approach for deciding a risk budget that is independent of the number of uncontrollable events across an entire PSTN, α^S, is to set $\alpha = 1 - \sqrt[m]{1-\alpha^S}$, which distributes that risk budget equitably across uncontrollable events.

Our PSTN to STNU α-based approximation affords a natural parameterization based on the risk level α. For instance, we hypothesize that setting α too high will result in riskier edges and unnecessarily sacrifices too much likelihood, hurting overall performance of an approach like MinLoss DC. However, setting α too low might also hurt performance, since it will require more relaxations of the STNU edges. While the LSC-LP and ODC-Relax approaches attempt to relax the network optimally, they do so on STNU edges that are agnostic to the underlying probability density functions. We explore how α impacts the performance of a method like LSC-LP and Min-Loss DC in our empirical analysis.

Finally, both our LSC and LDC approaches rely on an α-based STNU approximation of a PSTN and then use DSC and DDC to yield an estimated degree of controllability, p_α. This yields an overall estimated likelihood of success of $(1-\alpha)^m p_\alpha$ in both cases. Thus, one could extend both our approaches so that they incorporate a search over α as an outer-loop that seeks to maximize the slightly more refined estimated likelihood of success $(1-\alpha)^m p_\alpha$ for each problem instance. However, doing so would introduce significant computational overhead, since it would require sweeping the discretized space of possible α values and solving the resulting degree of controllability problem for each α parameter setting. We show in our empirical evaluation that we are able to achieve state-of-the-art performance without requiring this additional computational overhead by empirically tuning α once for the entire benchmark of instances.

7. Empirical evaluation

In this section, we evaluate both the accuracy and the optimality of our degree of controllability and likelihood of controllability. We start by empirically validating our definitions of Degrees of Controllability as applied to STNUs.

7.1. Degree of controllability of STNUs

To evaluate our approaches[2] for assessing DSC and DDC, we ran experiments on STNUs derived from the PSTNs provided in the publicly available *ROVERS* and *CAR-SHARING* datasets [10]. STNU-style contingent edges (i.e. those with uniform distributions) were preserved. PSTN-style contingent edges in the *ROVERS* dataset were converted by replacing contingent probabilistic edges drawn from $\mathcal{N}(\mu, \sigma^2)$ with intervals $[\mu - 2\sigma, \mu + 2\sigma]$. STNUs derived from the *ROVERS* dataset were used to evaluate DSC. The PSTNs from *CAR-SHARING* were converted to STNUs by preserving the network structure, but varying the lengths of edges to keep the derived STNUs consistent yet not dynamically controllable.

7.1.1. Accuracy of the DSC-LP decision

We examine how well our DSC metric actually measures success rate for the strong-LP decision. For each STNU, we solved the DSC-LP for that network. We took the decision returned by the program and used it as a dispatch strategy. After sampling uniformly from the contingent edges, we picked random realizations 50,000 times for each STNU. Then, we recorded the success rate as the proportion of the time the strong-LP decision, together with a random realization, led to a valid schedule. This will determine if the strong-LP decision produces better-than-average success rates (compared to other decisions) when used during execution. In Fig. 4 this success rate is plotted against the approximate DSC value returned

[2] All code and problem instances available for download from https://github.com/HEATlab/Prob-in-Ctrl.

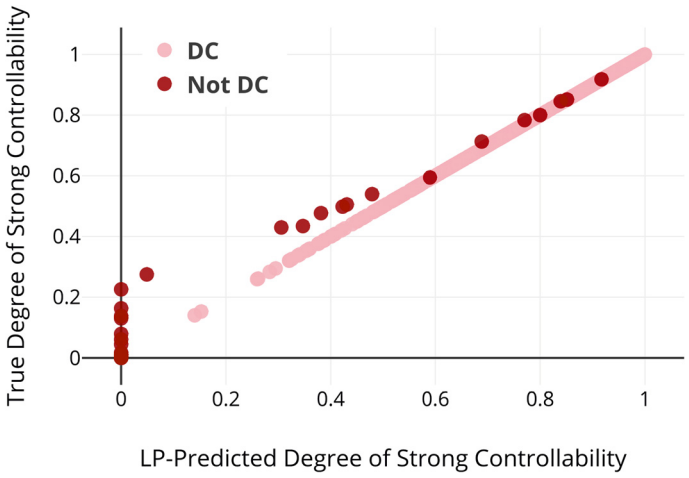

Fig. 5. True DSC *vs.* our DSC-LP Prediction.

by the DSC-LP. The graph displays a clear linear trend ($r = 0.999$), showing that the LP-predicted DSC of an STNU closely matches the rate of successfully executing the network using the strong-LP decision. This also validates the notion that the strong-decision produces better than average execution results when paired with a realization. Next, we discuss how our LP-predicted DSC compares with the true degree of strong controllability.

7.1.2. Optimality of DSC-LP

We now check how accurate our DSC-LP approximation is by comparing it to the true optimization problem of maximizing volume. Selecting subintervals to maximize volume is a nonlinear optimization problem. We solved these problems using Baron optimization software accessed through the NEOS server [24,25] to compute the true DSC for all problem instances. We compared this value against the approximate degree found using the LP. Fig. 5 demonstrates the correlation between our LP-predicted DSC approximation and the solution to the optimization problem. It demonstrates that at low degrees of strong controllability our LP may underestimate the degree, but at values greater than 0.5 the LP approximation is extremely accurate ($r = 0.996$). This result is consistent with our expectation that the approximation method works best for STNUs with high DSC values. Although DSC-LP is slightly less accurate at lower DSC values, when our approximation method returns a low value for the DSC we expect the true DSC to be low as well. In these cases where the DSC is quite low, committing to a fixed decision is never a good strategy to employ, so the lack of precision in these cases is perhaps not so concerning. We note that DSC-LP performs better on networks that are dynamically controllable than those that are not. However, given that instances in *ROVERS* were predominantly dynamically controllable whereas instances in *CAR-SHARING* were exclusively not dynamically controllable by design, we cannot rule out the possibility that this difference in performance might be due to structural differences between networks in *ROVERS versus CAR-SHARING*.

We also compared DSC-LP against other choices for objective functions that could be used in conjunction with our LP for approximating the DSC. Naturally, since DSC represents the optimal possible strongly controllable strategy, those methods that achieve higher empirical success rates are those that are doing a better job of approximating DSC. The formulations, which all enforce strong controllability, that we compared included:

- DSC-LP: our DSC-LP as defined in Section 4;
- Max subinterval: maximizes the sum of the length of each contingent subinterval [18];
- Minimax: minimize the maximum amount of that a single contingent interval is squeezed; and
- Maximin: maximize the minimum length over all contingent subintervals.

For each objective function, we generated simulated empirical success rate as we described in Section 7.1.1 for all problem instances we have (both from *ROVERS* and *CAR-SHARING* dataset) except 36 uncontrollable networks, for which the original LP from [18] failed to find a feasible solution. As discussed in Section 4 and shown in Table 1, our DSC-LP yields the highest success rate across all instances and thus offers the best approximation for DSC among the methods we tested.

7.1.3. Accuracy of approximate DDC

We now examine how well our estimate of DDC measures the actual success rate of STNU dispatch. For each STNU, we computed the estimate of DDC using the normal approximation method. To measure the dispatch rate empirically, we used the early-first dispatch strategy discussed in [26] on all uncontrollable STNUs derived from the datasets. This strategy is guaranteed to succeed on dynamically controllable networks, and thus was a natural choice for testing a network's DDC. In each trial, we simulated online dispatch $50,000$ times (with realizations chosen randomly). Success rate was recorded as

Table 1
Comparison of our DSC-LP with other LP formulations for establishing Strong Controllability.

Approach	Dispatch success rate
DSC-LP	69.13%
Max subinterval [18]	66.26%
Minimax	36.21%
Maximin	30.31%

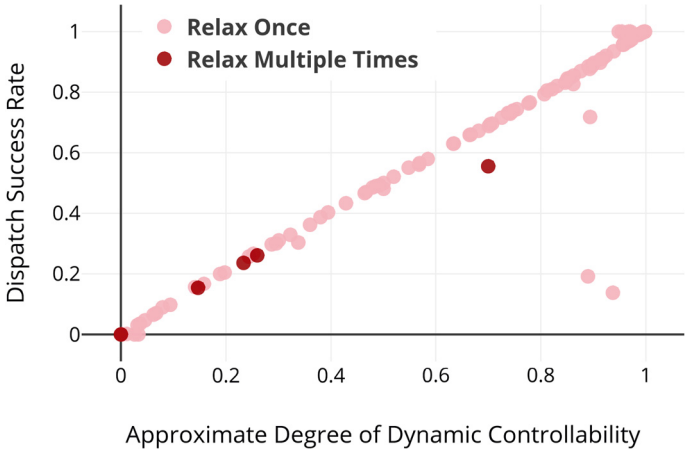

Fig. 6. Dispatch Success Rate *vs.* Approximate DDC.

the proportion of the time we obtained a valid schedule. In Fig. 6, success rate is plotted against the approximate DDC. The graph displays a clear linear trend ($r = 0.952$), indicating that the normal approximation to DDC accurately tracks the actual success rate. The approach also worked well for STNUs that had multiple separate conflicts, marked as darker points in the graph.

For two outliers in the bottom right corner of the plot, the normal approximation seems to drastically overestimate the success rate. In those two instances, after we employed a different dispatch strategy that waited longer before scheduling certain events, the success rates for the two STNUs were within 2% of the normal approximation. So even in these cases, the predicted DDC matches the *maximum* success rate achievable with dynamic strategies. In the few other cases where the normal approximation method overestimated the empirical success rate, it is possible that the variant of the early first strategy we used was not the optimal choice.

7.2. Accuracy of Likelihood of Controllability of PSTNs

Next, we evaluate how well our definitions of Degree of Controllability for STNUs can be translated into Likelihoods of Controllability for PSTNs. We test on the same benchmarks as before, but this time using the PSTN versions of each network. In this case, we preserve the original PSTN-style edges, and instead convert STNU-style contingent edges to probabilistic ones by essentially running our earlier conversion process (to get from probabilistic contingent edges to uniform contingent edges) in reverse. That is, we replace each contingent interval of the form $[\mu - 2\sigma, \mu + 2\sigma]$ in an STNU from the original benchmark with a contingent constraint with distribution $\mathcal{N}(\mu, \sigma^2)$ that is normally distributed about the interval's midpoint to obtain a PSTN. This reverse conversion process also allows us to maintain the aforementioned adjustments to the *CAR-SHARING* benchmarks so that they are likely to be consistent but not dynamically controllable.

7.2.1. Accuracy of the LSC-LP decision

LSC-LP begins by extracting an STNU that approximates the original PSTN. We do this by setting a risk $\alpha = 0.05$, which means we extract an STNU whose contingent edges each capture exactly two standard deviations (i.e. 95%) of the original PSTN contingent edge. We then use DSC-LP to find the strong decision that approximates the Degree of Strong Controllability. The DSC-LP will result in an output that approximately captures the largest volume of the hyperrectangular realization space. We can then use the original PSTN to calculate how much probability mass is captured along each dimension of this hyperrectangle. The product across all dimensions gives us the total probability mass captured by the strong decision, which we use as our approximate LSC.

As before, we took the decision returned by the program and used it as a dispatch strategy. We again select $50,000$ random realizations for each PSTN, but this time we use the sample from the Normal distribution associated with each contingent edge (rather than the uniform distribution of an STNU). Then, we recorded the success rate as the proportion

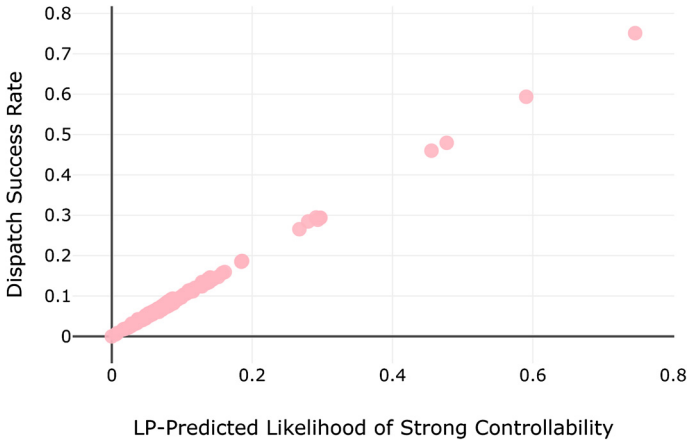

Fig. 7. Dispatch Success Rate vs. our LSC-LP Prediction.

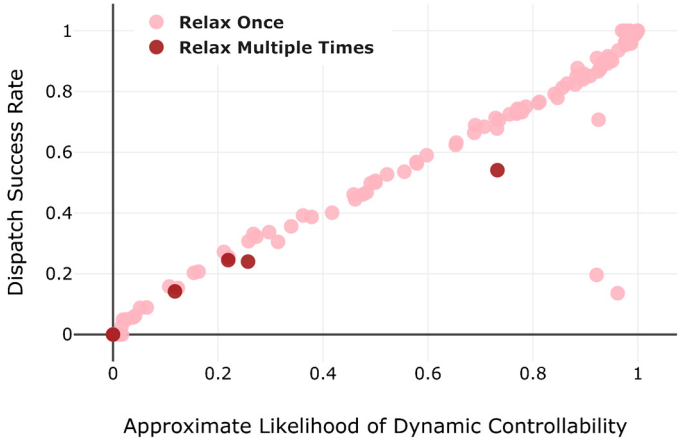

Fig. 8. Dispatch Success Rate vs. Approximate LDC.

of the time the strong-LP decision, together with a random realization, led to a valid schedule. In Fig. 7 this success rate is plotted against the approximate LSC value returned by the LSC-LP. Consistent with the earlier DSC results, the graph exhibits a clear linear trend ($r = 0.999$). Note that unlike the analogous evaluation for DSC depicted in Fig. 5, we do not distinguish between DC and non-DC networks, since all the PSTNs being tested are inherently not DC.

7.2.2. Accuracy of approximate LDC

We also evaluated the performance of our LDC approximation method. We again used the PSTN edge's distribution information to assist in calculating approximate LDC, this time using it in the Central Limit Theorem's normal approximation step. In order to identify potential conflicts, we extract an STNU that approximates the original PSTN with risk $\alpha = 0.05$, which results in an interval that captures two standard deviations per contingent edge. The edge distributions also were used to sample contingent edges when generating random realizations. The results of this evaluation are shown in Fig. 8. As with DDC, our LDC results exhibit a strong linear relationship, with $r = 0.949$.

7.3. Optimality of likelihood of controllability of PSTNs

Since no tractable approaches for optimally solving likelihood of controllability problems currently exist, we compare both LSC-LP and Min-Loss DC to the current state-of-the-art approximate approaches. We focus our comparison on the Static Robust Execution Algorithm (SREA) [12] since it is the previous approach that most directly optimizes for LSC. We also compare against an online approach, the Dynamic Robust Execution Algorithm (DREA) [12] that reruns SREA every time new information is received and is thus the closest known approach to approximating LDC.

Throughout, we use the dynamic controllability dispatch algorithm taken from [26] to dispatch on the output of the methods described above. Since strong controllability is a special case of dynamic controllability, this algorithm can dispatch strongly controllable networks without requiring any adjustments. Across all approaches, we adopt the convention of [12, 13]. That is, rather than giving up if a realization falls outside of the STNU approximation, we attempt to dispatch the

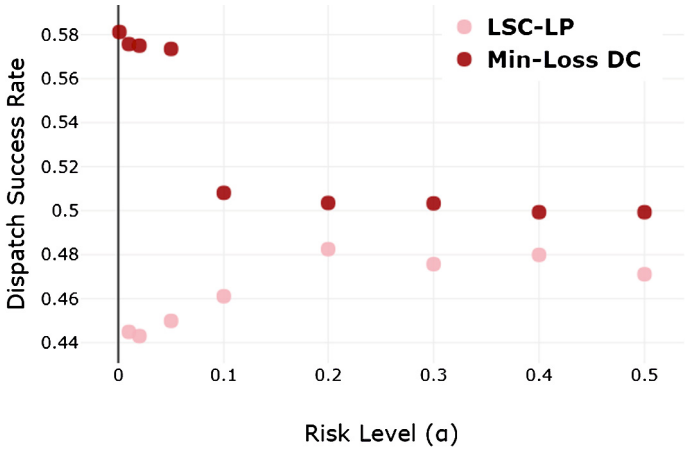

Fig. 9. LSC-LP and Min-Loss DC Dispatch Success Rates Across Risk Levels on *CAR-SHARING*.

Table 2
Performance of dispatch strategies for *CAR-SHARING* benchmark.

Approach	Dispatch success rate	Runtime (milliseconds)
Min-Loss DC	**57.05%**	161.2
DREA	54.85%	10911.4
LSC-LP	52.70%	256.7
SREA	51.92%	7430.1
DC-Dispatch	49.55%	**41.5**

network without controllability guidance, and execute the controllable timepoints as soon as all constraints corresponding to preceding events are satisfied. Finally, we also compare against DC-Dispatch, the dynamic controllability dispatch algorithm taken from [26], with STNU-controllability guidance incorporated.

We simulated real-time dispatch 200 times per problem instance, sampling randomly according to the probability distribution of each contingent edge. We focus these comparisons on the *CAR-SHARING* problem instances, because dispatch simulations coupled with the computationally intensive SREA and DREA methods struggled with the significantly larger *ROVERS* problem instances.

7.3.1. Finding optimal risk levels

In order to maximize the performance of LSC-LP and Min-Loss DC, we first need to determine what risk level (setting of α across all edges) would predict the best performance on the *CAR-SHARING* data set. The results are summarized in Fig. 9.

Interestingly, we found that Min-Loss performed best when set to the lowest risk level we tested, $\alpha = 0.001$, which corresponded to extracting an STNU where the bounds over uncertain edges each capture 99.9% of the likelihood. This demonstrates the ODC-Relax algorithm's ability to minimally relax uncertain intervals only when conflicts necessitate, and that it does so only for the interval involved in the conflict. This suggests that setting α to very small values allows ODC-Relax the flexibility in sacrificing likelihood only where it is needed.

LSC-LP, on the other hand, showed quite different trends that largely mirrored the Min-Loss DC results. Here, setting α too low hurt overall performance. We suspect that this was because the DSC-LP, unlike ODC-Relax, is not driven by conflicts and is agnostic of the underlying distribution. Thus, setting α too small provides the LP more opportunity to miss the densest portion of the probability mass in an attempt to generically optimize for volume. LSC-LP's dispatch success rate peaks at $\alpha = 0.2$ and drops slowly after that. Note that the results shown in Fig. 9 assume that the LSC-LP is used to guide the Dispatch-DC algorithm. The fact that Dispatch-DC attempts to guide execution even if the realizations fall outside the original STNU bounds mitigate how much performance drops off as α increases.

7.3.2. Optimality of LSC-LP and Min-Loss DC

The results of our empirical comparison across the five algorithms are shown in Table 2. Min-Loss DC demonstrates the highest overall success rate, a result that we confirmed across a wide diversity of problem instances. Min-Loss DC had statistically significant ($p < 0.05$) higher dispatch success rates than LSC-LP, SREA, and DC-Dispatch. The only other statistically significant difference was that DREA achieved higher dispatch rates than both SREA and DC-Dispatch, but not LSC-LP.

The most competitive approach with Min-Loss DC is DREA, which is an online algorithm that reruns SREA *every* time new information about the schedule is received. The fact that Min-Loss DC achieves a higher average success rate than DREA

is surprising and notable, because Min-Loss DC computes a dispatchable network *offline*, prior to execution and so incurs no additional computational overhead during execution compared to other dynamic dispatch methods. This is in stark contrast to DREA, whose advantage is that it gets to recompute a completely new, strongly controllable schedule that re-optimizes for new information every time new information arrives. As noted in Table 2, this leads to immense computational overhead, with DREA expending *two orders of magnitude* more computation than Min-Loss DC.

We now turn to comparing the two methods for approximately computing the LSC: LSC-LP and SREA. As shown in Table 2, both approaches end up having fairly comparable dispatch success rates, with LSC-LP slightly edging out SREA, but not by a significant margin. LSC-LP's main advantage over the SREA is that it is a single-shot LP rather than an LP that gets run many times as part of the binary search inner-loop. This results in an *order-of-magnitude* reduction in computation time. We note though, that most of this speedup can be attributed to the fact that SREA auto-tunes its α value for each particular problem instance, whereas LSC-LP does this only once across the entire benchmark offline and *a priori*.

DC-Dispatch had the least computational overhead, which is to be expected given that it does not attempt to optimize controllability of PSTNs. Note however that, except for DREA, the extra computational overhead associated with computing controllable network occurs offline *before* execution begins and, therefore, is unlikely to preclude its use in any situation where DC-Dispatch can be deployed.

8. Conclusion

In this paper, we applied a geometric view of STNUs and expanded the notion of controllability to a continuous measure. These new degrees of strong and dynamic controllability assess how far an STNU is from being controllable. We used this definition in the cases of strong and dynamic controllability to produce the Degree of Strong Controllability (DSC) and the Degree of Dynamic Controllability (DDC) metrics, which provide information on the maximum probability of success using certain types of offline and online strategies respectively. In doing so, we found an efficient LP for approximating DSC, presented a solution to a variant of the relaxation problem discussed by Bhargava et al. [20], and provided a normal approximation method for estimating DDC. We showed that DSC and DDC could be extended respectively to the new metrics Likelihood of Strong Controllability (LSC) and Likelihood of Dynamic Controllability) for PSTNs. Our empirical comparison demonstrated the usefulness of our metrics in capturing the likelihood of successful dispatch for both the strong and dynamic controllability cases and also for both STNUs and PSTNs. These contributions present a unified, geometric way of tracking the controllability of temporal networks.

In the future, it would be interesting to define a continuous metric that interpolates between strong, dynamic, and weak controllability. Although these metrics extend the definition of controllability in useful ways, they currently draw no direct connection between strong and dynamic controllability. That is, even though strong controllability implies dynamic controllability, the DSC value does not provide any information about whether a network is dynamically controllable. This could yield an explicit way of classifying states intermediate between dynamic and strong controllability and offer a more refined way of comparing different networks. Lastly, our relaxation algorithm and normal approximation for the DDC currently do not handle STNUs with multiple conflicts well, as they overlook the possibility of conflicts sharing contingent edges. Further work could attempt to figure out how to contend with these dependencies between conflicts in a way that more accurately assesses the probability of successful dispatch.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

Funding for this work was graciously provided by The National Science Foundation under grant IIS-1651822 and also The Rose Hills Foundation. Thanks to the anonymous reviewers, HMC faculty, staff, and fellow HEATlab members for their support and constructive feedback. We thank Nikhil Bhargava for providing assistance with the implementation of algorithms presented in [20]. Finally, we thank Jordan Abrahams, Susan Martonosi, and Mohamed Omar for offering their expertise in temporal networks, optimization, and convex geometry respectively.

References

[1] S. Akmal, S. Ammons, H. Li, J.C. Boerkoel Jr., Quantifying degrees of controllability in temporal networks with uncertainty, in: Proc. of the 29th International Conference on Automated Planning and Scheduling, ICAPS-19, 2019, pp. 22–30.
[2] M. Gao, L. Popowski, J.C. Boerkoel Jr., Dynamic control of probabilistic simple temporal networks, in: Proc. of the 34st AAAI Conference on Artificial Intelligence, AAAI-20, 2020, pp. 22–30.
[3] R. Deichter, I. Meiri, J. Pearl, Temporal constraint networks, Artif. Intell. 49 (1991) 61–95.
[4] T. Vidal, H. Fargier, Handling contingency in temporal constraint networks: from consistency to controllabilities, J. Exp. Theor. Artif. Intell. (1999) 23–45.
[5] L. Hunsberger, Fixing the semantics for dynamic controllability and providing a more practical characterization of dynamic execution strategies, in: Proc. of the 16th International Symposium on Temporal Representation and Reasoning, TIME-09, 2009, pp. 155–162.

[6] P. Morris, A structural characterization of temporal dynamic controllability, in: Proc. of Principles and Practice of Constraint Programming, CP-06, 2006, pp. 375–389.
[7] I. Tsamardinos, A probabilistic approach to robust execution of temporal plans with uncertainty, Methods Appl. Artif. Intell. (2002) 97–108.
[8] P. Yu, C. Fang, B. Williams, Resolving uncontrollable conditional temporal problems using continuous relaxations, in: Proc. of 24th International Conference on Automated Planning and Scheduling, ICAPS-14, 2014, pp. 341–349.
[9] C. Fang, P. Yu, B.C. Williams, Chance-constrained probabilistic simple temporal problems, in: Proc. of the 28th AAAI Conference on Artificial Intelligence, AAAI-16, 2014, pp. 2264–2270.
[10] P. Santana, T. Vaquero, C. Toledo, A. Wang, C. Fang, B. Williams, Paris: a polynomial-time, risk-sensitive scheduling algorithm for probabilistic simple temporal networks with uncertainty, in: Proc. of the 26th International Conference on Automated Planning and Scheduling, ICAPS-16, 2016, pp. 267–275.
[11] J. Brooks, E. Reed, A. Gruver, J.C. Boerkoel, Robustness in probabilistic temporal planning, in: Proc. of the 29th AAAI Conference on Artificial Intelligence, AAAI-15, 2015, pp. 3239–3246.
[12] K. Lund, S. Dietrich, S. Chow, J. Boerkoel, Robust execution of probabilistic temporal plans, in: Proc. of the 31st AAAI Conference on Artificial Intelligence, AAAI-17, 2017, pp. 3597–3604.
[13] J.R. Abrahams, D.A. Chu, G. Diehl, M. Knittel, J. Lin, W. Lloyd, J.C. Boerkoel, J. Frank, Dream: an algorithm for mitigating the overhead of robust rescheduling, in: Proc. of the 29th International Conference on Automated Planning and Scheduling, ICAPS-19, 2019, pp. 3–12.
[14] A. Huang, L. Lloyd, M. Omar, J. Boerkoel, New perspectives on flexibility in simple temporal planning, in: Proc. of the 28th International Conference on Automated Planning and Scheduling, ICAPS-18, 2018, pp. 123–131.
[15] A. Cimatti, A. Micheli, M. Roveri, An SMT-based approach to weak controllability for disjunctive temporal problems with uncertainty, Artif. Intell. 224 (2015) 1–27.
[16] E.T. Jaynes, Probability Theory: The Logic of Science, Cambridge University Press, 2003.
[17] T.S. Motzkin, E.G. Straus, Maxima for graphs and a new proof of a theorem of Turán, Can. J. Math. 17 (1965) 533–540.
[18] M. Wilson, T. Klos, C. Witteveen, B. Huisman, Flexibility and decoupling in simple temporal networks, Artif. Intell. 214 (2014) 26–44.
[19] P. Morris, N. Muscettola, Temporal dynamic controllability revisited, in: Proc. of the 20th National Conference on Artificial Intelligence, AAAI-05, 2005, pp. 1193–1198.
[20] N. Bhargava, T. Vaquero, B. Williams, Faster conflict generation for dynamic controllability, in: Proc. of the 26th International Joint Conference on Artificial Intelligence, IJCAI-17, 2017, pp. 4280–4286.
[21] J. Cui, P. Yu, C. Fang, P. Haslum, B.C. Williams, Optimising bounds in simple temporal networks with uncertainty under dynamic controllability constraints, in: Proc. of the 25th International Conference on Automated Planning and Scheduling, ICAPS-15, 2015, pp. 52–60.
[22] P. Yu, C. Fang, B. Williams, Resolving over-constrained probabilistic temporal problems through chance constraint relaxation, in: Proc. of the 29th AAAI Conference on Artificial Intelligence, AAAI-15, 2015, pp. 3425–3431.
[23] F. Rossi, B. Venable, N. Yorke-Smith, Simple temporal problems with preferences and uncertainty, in: Proc. CP 2003 Workshop on Online Constraint Solving: Handling Change and Uncertainty, 2003.
[24] E.D. Dolan, NEOS server 4.0 administrative guide, CoRR, arXiv:cs.DC/0107034, 2001.
[25] M. Tawarmalani, N.V. Sahinidis, A polyhedral branch-and-cut approach to global optimization, Math. Program. 103 (2005) 225–249.
[26] M. Nilsson, J. Kvarnström, P. Doherty, Classical dynamic controllability revisited – a tighter bound on the classical algorithm, in: Proc. of 6th International Conference on Agents and Artificial Intelligence, ICAART-14, Angers, France, 6–8 March 2014, 2014, pp. 130–141.