UNIVERSITÀ di VERONA

Dipartimento di INFORMATICA

Rapporto di ricerca
Research report

103/2018

February, 14 2018

Dynamic-Consistency Checking for Conditional Simple Temporal Networks: Strengthening the Theoretical Foundations and Presenting a Faster Algorithm

Luke Hunsberger

Vassar College – Poughkeepsie, NY USA

Roberto Posenato

University of Verona – Verona, Italy

Abstract

Recent work on Conditional Simple Temporal Networks (CSTNs) has focused on checking the dynamic consistency (DC) property for the case where an execution strategy can react instantaneously to observations. Three alternative semantics for such strategies—IR-dynamic, 0-dynamic, and π-dynamic—have been presented. However, the most practical DC-checking algorithm has only been analyzed with respect to the IR semantics. Meanwhile, 0-dynamic strategies were shown to permit a kind of circular dependence among simultaneous observations, making them impossible to implement, whereas π-dynamic strategies prohibit this kind of circularity. Whether IR-dynamic strategies allow this kind of circularity and, if so, what the consequences would be for the above-mentioned DC-checking algorithm remained open questions.

This paper makes the following contributions: (1) it shows that IR-dynamic strategies *do* allow circular dependence and, thus, that the IR semantics does not properly capture instantaneous reactivity; (2) it shows that one of the constraint-propagation rules from the IR-DC-checking algorithm is *unsound* with respect to the IR semantics; (3) it presents a simpler DC-checking algorithm, called the π-DC-checking algorithm, that uses half of the rules from the earlier algorithm, and that it proves is sound and complete with respect to the π-DC semantics; (4) it empirically evaluates the new algorithm. Thus, the paper places practical DC checking for CSTNs in the case of instantaneous reaction on a solid theoretical foundation.

1 Overview

A Conditional Simple Temporal Network (CSTN) is a data structure for reasoning about time in domains where some constraints may apply only in certain scenarios. For example, a patient who tests positive for a certain disease may need to receive care more urgently than someone who tests negative. Conditions in a CSTN are represented by propositional letters whose truth values are not controlled, but instead *observed* in real time. Just as doing a blood test generates a positive or negative result that is only learned in real time, the execution of an *observation time-point* in a CSTN generates a truth value for its corresponding propositional letter. An execution strategy for a CSTN specifies when the time-points will be executed. A strategy can be *dynamic* in that its decisions can react to information from past observations. A CSTN is said to be dynamically consistent (DC) if it admits a dynamic strategy that guarantees the satisfaction of all relevant constraints no matter which outcomes are observed during execution.

Different varieties of the DC property have been defined that differ in how reactive a dynamic strategy can be. Originally, Tsamardinos *et al.* [1] stipulated that a strategy can react to an observation after any arbitrarily small, but positive delay. Comin *et al.* [2] defined ϵ-DC, which assumes that a strategy's reaction time is bounded below by some $\epsilon > 0$. Finally, three different versions of DC for strategies that can react instantaneously (i.e., after zero delay) have been defined: IR-DC [3]; 0-DC and π-DC [4].

Although several approaches to DC-checking algorithms have been presented to address the different flavors of DC [1, 5, 6, 2], the approach based on the propagation of labeled constraints is the only one that has been demonstrated to be practical [3, 7]. Three variations of their DC-checking algorithm have been presented: one for DC, one for ϵ-DC, and one for IR-DC. This paper focuses on their IR-DC-checking algorithm.

Subsequently, Cairo *et al.* [4] showed that the ϵ-DC semantics, in the case where $\epsilon = 0$, did not properly capture instantaneous reaction because it allows a kind of circular dependence among simultaneous observations. To correct this flaw, they presented the π-DC semantics that requires a strategy to specify an order-of-dependence among simultaneous observations. Whether the IR-DC semantics allows this kind of circularity and, if so, what the consequences would be for the IR-DC-checking algorithm, remained open questions.

This paper makes the following contributions: (1) it shows that IR-dynamic strategies *do* allow circular dependence and, thus, that the IR semantics does not properly capture instantaneous reactivity; (2) it shows that one of the propagation rules from the IR-DC-checking algorithm is *unsound* with respect to the IR semantics; (3) it presents a simpler DC-checking algorithm, called the π-DC-checking algorithm, that uses half of the rules from the earlier algorithm, and that it proves is sound and complete with respect to the π-DC semantics; (4) it empirically evaluates the new algorithm. Thus, the paper places practical DC checking for CSTNs in the case of instantaneous reaction on a solid theoretical foundation.

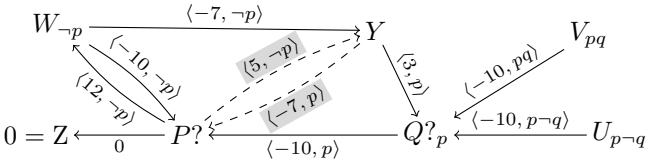


Figure 1: A sample CSTN

2 Background

Dechter *et al.* [8] introduced Simple Temporal Networks (STNs) to facilitate representing and reasoning about temporal constraints. An STN comprises real-valued variables, called *time-points,* and binary difference constraints on those variables. Typically, an STN includes a special time-point, Z, whose value is fixed at zero. A *consistent* STN is one that has a solution as a constraint satisfaction problem.

Tsamardinos *et al.* [1] presented CSTNs, which augment STNs to include *observation time-points* and their associated *propositional letters.* In a CSTN, the execution of an observation time-point $P?$ generates a truth value for its associated propositional letter p. In addition, each time-point in a CSTN can be labeled by a conjunction of propositional literals that specifies the scenarios in which that time-point must be executed. Since constraints among labeled time-points may similarly be applicable only in certain scenarios, later work generalized CSTNs to also include labels on constraints [9, 10].

Fig. 1 shows a sample CSTN in its graphical form, where the nodes represent time-points, and the directed edges represent binary difference constraints. In the figure, Z is fixed at 0; and $P?$ and $Q?$ are observation time-points whose execution generates truth values for p and q, respectively. $Q?$ being labeled by p indicates that $Q?$ is executed only if p happens to be *true*. Similarly, the edge from U to $Q?$ being labeled by $p\neg q$ indicates that it applies only in scenarios where p is *true* and q is *false*. The dashed edges with shaded labels are generated by the IR-DC-checking algorithm [3], to be discussed later on.

Tsamardinos *et al.* [1] noted that for any reasonable CSTN, the propositional labels on its time-points must satisfy certain properties. Hunsberger *et al.* [9] later extended these properties to accommodate labels on constraints. Together, their properties formalized the notion of a *well-defined* CSTN. Recently, Cairo *et al.* [11] showed that for any well-defined CSTN, no loss of generality results from subsequently *removing* the labels from its time-points. Therefore, this paper restricts attention to CSTNs whose time-points do not have any propositional labels, what Cairo *et al.* [11] called *streamlined* CSTNs. Since streamlined CSTNs are necessarily well defined, and the applicability conditions of the constraint-propagation rules become simpler when there are no labels on time-points, the following presentation benefits dramatically from the restriction to streamlined CSTNs.

2.1 Streamlined CSTNs

The following definitions are from Hunsberger *et al.* [3], except that propositional labels appear only on constraints. Henceforth, the term CSTN shall refer to streamlined CSTNs.

Definition 1 (Labels)**.** Given a set $\mathcal{P}$ of propositional letters:

- a *label* is a (possibly empty) conjunction of (positive or negative) literals from $\mathcal{P}$. The empty label is notated $\boxdot$.
- for any label ℓ, and any $p \in \mathcal{P}$, if $\ell \models p$ or $\ell \models \neg p$, then we say that p *appears* in ℓ.
- for any labels ℓ_1 and ℓ_2, if $\ell_1 \models \ell_2$ then ℓ_1 is said to *entail* ℓ_2. If $\ell_1 \wedge \ell_2$ is satisfiable, ℓ_1 and ℓ_2 are called *consistent.*
- the *label universe* of $\mathcal{P}$, denoted by $\mathcal{P}^*$, is the set of all *consistent* labels whose literals are drawn from $\mathcal{P}$.

Definition 2 (CSTN)**.** A *Conditional Simple Temporal Network* (CSTN) is a tuple, $\langle \mathcal{T}, \mathcal{P}, \mathcal{C}, \mathcal{OT}, \mathcal{O} \rangle$, where:

- $\mathcal{T}$ is a finite set of real-valued time-points (i.e., variables);
- $\mathcal{P}$ is a finite set of propositional letters (or propositions);
- $\mathcal{C}$ is a set of *labeled* constraints, each having the form, $(Y - X \leq \delta,\ \ell)$, where $X, Y \in \mathcal{T}$, $\delta \in \mathbb{R}$, and $\ell \in \mathcal{P}^*$;
- $\mathcal{OT} \subseteq \mathcal{T}$ is a set of observation time-points (OTPs); and
- $\mathcal{O} : \mathcal{P} \rightarrow \mathcal{OT}$ is a bijection that associates a unique observation time-point to each propositional letter.

In a CSTN graph, the observation time-point for p (i.e., $\mathcal{O}(p)$) may be denoted by $P?$; and each labeled constraint, $(Y - X \leq \delta, \ell)$, is representedd by an arrow from X to Y annotated by the *labeled value,* $\langle \delta, \ell \rangle$: $X \xrightarrow{\langle \delta, \ell \rangle} Y$. Since any time-points X and Y may participate in multiple constraints of the form, $(Y - X \leq \delta_i, \ell_i)$, the corresponding edge from X to Y may have multiple labeled values of the form, $\langle \delta_i, \ell_i \rangle$.

Definition 3 (Scenario)**.** A *scenario* over a set $\mathcal{P}$ of propositional letters is a function, $s : \mathcal{P} \rightarrow \{true, false\}$, that assigns a truth value to each letter in $\mathcal{P}$. Any such function also provides the truth value for any label $\ell \in \mathcal{P}^*$, denoted by $s(\ell)$. The set of all scenarios over $\mathcal{P}$ is denoted by $\mathcal{I}$.

Definition 4 (Schedule)**.** A *schedule* for a set of time-points $\mathcal{T}$ is a mapping, $\psi : \mathcal{T} \rightarrow \mathbb{R}$. The set of all schedules *for any subset of* $\mathcal{T}$ is denoted by Ψ.

The projection of a CSTN onto a scenario, s, is the STN obtained by restricting attention to the constraints whose labels are true under s (i.e., must be satisfied in scenario s).

Definition 5 (Projection)**.** Let $\mathcal{S} = \langle \mathcal{T}, \mathcal{P}, \mathcal{C}, \mathcal{OT}, \mathcal{O} \rangle$ be any CSTN, and s any scenario over $\mathcal{P}$. The *projection* of $\mathcal{S}$ onto s—notated $\mathcal{S}(s)$—is the STN, $(\mathcal{T}, \mathcal{C}_s^+)$, where:

$$\mathcal{C}_s^+ = \{(Y - X \leq \delta) \mid \exists \ell, (Y - X \leq \delta, \ell) \in \mathcal{C} \wedge s(\ell) = true\}$$

Definition 6 (Execution Strategy)**.** An *execution strategy* for a CSTN $\mathcal{S} = \langle \mathcal{T}, \mathcal{P}, \mathcal{C}, \mathcal{OT}, \mathcal{O} \rangle$ is a mapping, $\sigma : \mathcal{I} \rightarrow \Psi$, from scenarios to schedules. The execution time for the time-point X in the schedule $\sigma(s)$ is denoted by $[\sigma(s)]_X$.

Definition 7 (Viable Strategy)**.** An execution strategy σ for a CSTN $\mathcal{S}$ is *viable* if for each scenario s, the schedule $\sigma(s)$ is a solution to the projection $\mathcal{S}(s)$.

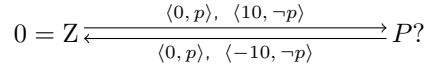


Figure 2: An absurd 0-DC CSTN

3 Strategies with Instantaneous Reaction

The truth values of propositions in a CSTN are not known in advance, but a *dynamic* execution strategy can *react* to observations in real time. Three *distinct* semantics for strategies that can react *instantaneously* to observations (i.e., after zero delay) have been defined: IR-dynamic [3]; and 0-dynamic and π-dynamic [4]. Since a CSTN is DC iff it has a viable and dynamic execution strategy, each distinct version of dynamic strategy gives rise to a distinct version of DC.

3.1 0-Dynamic Strategies

In an ϵ-dynamic strategy, $\epsilon > 0$ represents a fixed lower bound on the strategy's reaction times [2]. Cairo *et al.* [4] then considered the ramifications of setting $\epsilon = 0$, leading to the following definition.

Definition 8 (0-Dynamic Execution Strategy)**.** An execution strategy σ for a CSTN $\mathcal{S} = \langle \mathcal{T}, \mathcal{P}, \mathcal{C}, \mathcal{OT}, \mathcal{O} \rangle$ is 0-*dynamic* if for any scenarios s_1 and s_2, and any time-point $X \in \mathcal{T}$:

$$\begin{array}{rl} \text{either:} & [\sigma(s_1)]_X \geq [\sigma(s_2)]_X \\ \text{or:} & [\sigma(s_1)]_X \geq \min\{[\sigma(s_1)]_{P?} \mid s_1(p) \neq s_2(p)\}. \end{array}$$

I.e., if $\sigma(s_1)$ executes X *before* $\sigma(s_2)$ does, then $\sigma(s_1)$ must have observed some p *at or before* its execution of X, obtaining a different outcome than $\sigma(s_2)$'s observation of p.

Cairo *et al.* [4] showed that a 0-dynamic strategy can exhibit a kind of circular dependence among multiple simultaneous observations, making them impossible to implement. Even worse, the CSTN in Fig. 2 has a 0-dynamic strategy that exhibits a circular dependence involving just one observation. The labeled constraints stipulate that $P? = 0$ in scenarios where $p = \mathit{true}$; and $P? = 10$ where $p = \mathit{false}$[1] (i.e., the execution time of $P?$ must depend on the observation obtained by executing $P?$). Nonetheless, the CSTN has a viable and 0-dynamic strategy σ_0, where: $[\sigma_0(p)]_{\mathrm{Z}} = [\sigma_0(p)]_{P?} = [\sigma_0(\neg p)]_{\mathrm{Z}} = 0$ and $[\sigma_0(\neg p)]_{P?} = 10$. Clearly, σ_0 is viable. In addition, it is trivially 0-dynamic since $[\sigma_0(s_p)]_{P?} = 0 = \min\{[\sigma_0(s_p)]_{Q?} \mid s_p(q) \neq s_{\neg p}(q)\}$ and $[\sigma_0(s_{\neg p})]_{P?} = 10 > 0 = [\sigma_0(s_p)]_{P?}$.

3.2 IR-Dynamic Strategies

Hunsberger *et al.* [3] defined IR-dynamic strategies to support an analysis of their propagation-based DC-checking algorithm in the case of instantaneous reaction. This section recalls the definition of IR-dynamic strategies and then compares them to 0-dynamic strategies.

Definition 9 (History)**.** Let $\mathcal{S} = \langle \mathcal{T}, \mathcal{P}, \mathcal{C}, \mathcal{OT}, \mathcal{O} \rangle$ be any CSTN, s any scenario, σ any execution strategy for $\mathcal{S}$, and t any real number. The *history* of t in the

[1] $(P? = \delta)$ abbreviates $\{(P? - \mathrm{Z} \leq \delta), (\mathrm{Z} - P? \leq -\delta)\}$.

scenario s, for the strategy σ—notated $\mathit{Hist}(t, s, \sigma)$—is the set of observations made before time t according to the schedule $\sigma(s)$:

$$\mathit{Hist}(t, s, \sigma) = \{(p, s(p)) \mid P? \in \mathcal{OT} \wedge [\sigma(s)]_{P?} < t\}$$

Definition 10 (IR-Dynamic Strategy)**.** An execution strategy σ for a CSTN $\mathcal{S} = \langle \mathcal{T}, \mathcal{P}, \mathcal{C}, \mathcal{OT}, \mathcal{O} \rangle$ is called *IR-dynamic* if for any scenarios s_1 and s_2, and any time-point X:

let:	$t = [\sigma(s_1)]_X$
if:	$\mathit{Hist}(t, s_1, \sigma) = \mathit{Hist}(t, s_2, \sigma)$
then:	$[\sigma(s_2)]_X = t$
unless:	$[\sigma(s_1)]_{P?} = [\sigma(s_2)]_{P?} = t$ and $s_1(p) \neq s_2(p)$ for some $p \in \mathcal{P}$, where $P? \not\equiv X$ (i.e., $P?$ and X are distinct time-points).

In other words, if an IR-dynamic strategy σ executes X at time t in scenario s_1, and the schedules $\sigma(s_1)$ and $\sigma(s_2)$ have the same history of observations, then σ must also execute X at time t in s_2—unless some observation at time t (where $P? \not\equiv X$) yielded different results in the two scenarios.

The condition, $P? \not\equiv X$, prohibits the circular dependence exhibited by the 0-dynamic strategy σ_0. In particular, σ_0 is not IR-dynamic, since $[\sigma_0(p)]_{P?} = 0$ and $\mathit{Hist}(0, p, \sigma_0) = \mathit{Hist}(0, \neg p, \sigma_0) = \emptyset$, but $[\sigma_0(\neg p)]_{P?} \neq 0$, even though there is no $Q?$ different from $P?$ that $\sigma_0(p)$ and $\sigma_0(\neg p)$ both executed at 0. Thus, IR-dynamic and 0-dynamic are different.

Theorem 1 (IR-dynamic ⇒ 0-dynamic)**.** *If an execution strategy for a CSTN is IR-dynamic, then it is also* 0*-dynamic.*

Proof. Let σ be any IR-dynamic strategy, but suppose that σ is not 0-dynamic. Then, for some scenarios s_1 and s_2, some time-point X, and $t = [\sigma(s_1)]_X$, it must be that:

$$t < [\sigma(s_2)]_X \text{ and } t < \min\{[\sigma(s_1)]_{P?} \mid s_1(p) \neq s_2(p)\}.$$

With no loss of generality, assume t is minimal for this circumstance. Consider the schedules $\sigma(s_1)$ and $\sigma(s_2)$, each annotated with the truth values of observations as they occur. Since $t < [\sigma(s_2)]_X$, these annotated schedules differ at t. Let t^* be the first time at which they differ. By construction, $t^* \leq t$, and $\mathit{Hist}(t^*, s_1, \sigma) = \mathit{Hist}(t^*, s_2, \sigma)$. Now, there are only two ways the annotated schedules can differ at t^*:

Case 1: For some $P?$, $[\sigma(s_1)]_{P?} = [\sigma(s_2)]_{P?} = t^*$, but $s_1(p) \neq s_2(p)$. But this contradicts that $t^* \leq t < \min\{[\sigma(s_1)]_{P?} \mid s_1(p) \neq s_2(p)\}$.

Case 2: For some Y, $[\sigma(s_i)]_Y = t^* < [\sigma(s_j)]_Y$, where $\{s_i, s_j\} = \{s_1, s_2\}$. Since $\mathit{Hist}(t^*, s_i, \sigma) = \mathit{Hist}(t^*, s_j, \sigma)$ and σ is IR-dynamic, the "unless" clause of Defn. 10 must hold. Thus, there must be some $P? \not\equiv Y$ such that $[\sigma(s_i)]_{P?} = [\sigma(s_j)]_{P?} = t^*$ and $s_i(p) \neq s_j(p)$, which is a contradiction by Case 1. □

Although Theorem 1 shows that IR-dynamicity is stronger than 0-dynamicity, IR-dynamicity is nonetheless flawed. For example, consider the network shown in Fig. 3, which is equivalent to a network from Cairo *et al.* [4]. It has three observation time-points, $A?$, $B?$ and $C?$, with corresponding propositional letters,

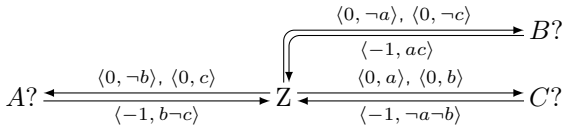


Figure 3: An absurd IR-DC CSTN

a, b and c. Each observation time-point is forced to execute at 0 in some scenarios, but at or above 1 in some other scenario. Thus, there is no single time-point that executes first, at or after Z, implying that no implementable, realistic strategy can exist. Yet, the strategy, σ', defined below, is viable and IR-dynamic (i.e., the network is IR-DC):

$$\begin{array}{lll} \text{if } s \models b\neg c, & [\sigma'(s)]_{A?} = 1, & \text{else } [\sigma'(s)]_{A?} = 0. \\ \text{if } s \models ac, & [\sigma'(s)]_{B?} = 1, & \text{else } [\sigma'(s)]_{B?} = 0. \\ \text{if } s \models \neg a\neg b, & [\sigma'(s)]_{C?} = 1, & \text{else } [\sigma'(s)]_{C?} = 0. \end{array}$$

Although tedious to check, it holds that for any scenarios, s_1 and s_2, and any observation time-point $X? \in \{A?, B?, C?\}$, where $[\sigma'(s_1)]_{X?} = 0$ and $[\sigma'(s_2)]_{X?} = 1$, there is always some *other* observation time-point $Y? \not\equiv X?$ that σ' executes at 0, but for which s_1 and s_2 generate different truth values. As a result, the "unless" clause of Defn. 10 invariably applies, allowing a circular dependence among $A?, B?$ and $C?$.

3.3 The IR-DC-checking Algorithm

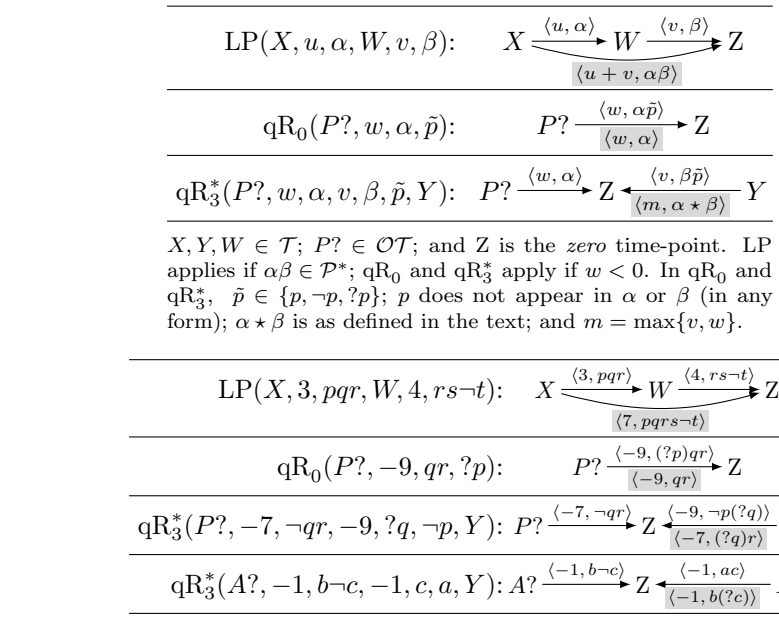
Hunsberger *et al.* [3] presented an algorithm based on the propagation of labeled constraints that they claimed was sound and complete for checking the IR-DC property. However, their algorithm says that the IR-DC network in Fig. 3 is *not* IR-DC. Therefore, something is wrong. This section will show that in the case of the IR-DC-checking algorithm, the fault lies with the IR-DC semantics, not the algorithm.

Table 1 lists three of the six constraint-propagation rules from the IR-DC-checking algorithm, along with sample applications that illustrate their use. Unlike the original, the LP rule shown in Table 1 focuses on generating edges terminating at Z, but that restriction does not affect what follows. These three rules will be sufficient to ensure completeness of the algorithm under the alternative semantics discussed later. The qR_3^* rule can generate a new kind of propositional label, called a *q-label*; and the qR_0 and qR_3^* rules can each be applied to q-labeled edges. Below, q-literals and q-labels are summarized, along with the $\star$ operator that extends ordinary conjunction to q-labels.

Whereas a constraint labeled by p must hold in all scenarios in which p is true, a constraint labeled by the q-literal $?p$ need only hold as long as the truth value of p is unknown (i.e., as long as $P?$ has not been executed).

Definition 11 (Q-literals, q-labels)**.**

- A *q-literal* is a literal of the form $?p$, where $p \in \mathcal{P}$.
- For convenience, if $p \in \mathcal{P}$, then $\tilde{p}, \tilde{p}_1$ or $\tilde{p}_2$ may be used to denote an arbitrary element of $\{p, \neg p, ?p\}$.



$X, Y, W \in \mathcal{T}$; $P? \in \mathcal{OT}$; and Z is the *zero* time-point. LP applies if $\alpha\beta \in \mathcal{P}^*$; qR$_0$ and qR$_3^*$ apply if $w < 0$. In qR$_0$ and qR$_3^*$, $\tilde{p} \in \{p, \neg p, ?p\}$; p does not appear in α or β (in any form); $\alpha \star \beta$ is as defined in the text; and $m = \max\{v, w\}$.

Table 1: The LP, qR$_0$ and qR$_3^*$ propagation rules for CSTNs (above) and instances of their application (below)

- For any scenario s, and any q-literal $?p$, it will be convenient to say that $s \not\models ?p$.
- A *q-label* is a conjunction of literals and/or q-literals.
- $\mathcal{Q}^*$ denotes the set of all q-labels.

For example, $p(?q)\neg r$ and $(?p)(?q)(?r)$ are both q-labels.

The $\star$ operator extends ordinary conjunction to accommodate q-labels. Intuitively, if C_1 is labeled by p, and C_2 is labeled by $\neg p$, then *both* constraints must hold as long as the value of p is unknown, which is represented by $p \star \neg p = ?p$.

Definition 12 ($\star$)**.** The operator, $\star : \mathcal{Q}^* \times \mathcal{Q}^* \rightarrow \mathcal{Q}^\star$, is defined in two steps. First, for any $p \in \mathcal{P}$, $p \star p = p$ and $\neg p \star \neg p = \neg p$; otherwise, $\tilde{p}_1 \star \tilde{p}_2 = ?p$. Next, for any q-labels, $\ell_1, \ell_2 \in \mathcal{Q}^*$, $\ell_1 \star \ell_2 \in \mathcal{Q}^*$ denotes the conjunction obtained by applying the $\star$ operator in pairwise fashion to corresponding literals from ℓ_1 and ℓ_2, as follows.

- If $\tilde{p}_1, \tilde{p}_2$ are literals in ℓ_1 and ℓ_2, respectively, then $\tilde{p}_1 \star \tilde{p}_2$ is contained in the conjunction, $\ell_1 \star \ell_2$.
- If $\tilde{p}$ is in ℓ_1, but proposition p does not appear in ℓ_2, then $\tilde{p}$ is contained in the conjunction, $\ell_1 \star \ell_2$.
- If $\tilde{p}$ is in ℓ_2, but proposition p does not appear in ℓ_1, then $\tilde{p}$ is contained in the conjunction $\ell_1 \star \ell_2$.

For example: $(p\neg q(?r)) \star (q\neg s) = p(?q)(?r)\neg s$.

Lemma 1. *The IR-DC-checking algorithm is not sound with respect to the IR-DC semantics.*

Proof. The CSTN from Fig. 3 is IR-DC, given the valid and IR-dynamic strategy σ'. But the IR-DC-checking algorithm declares it to be *not* IR-DC, as follows. First, applying the qR_3^* rule to the edges shown at the bottom of Table 1 generates the constraint $(B? \geq 1, b(?c))$.[2] Next, applying the qR_0 rule to this constraint yields: $(B? \geq 1, (?c))$ (i.e., B must be at least 1 as long as c is unknown). But this constraint is not satisfied by σ'. For example, in the scenario $\neg a \neg bc$, σ' executes $B?$ at 0, but $C?$ at 1. Furthermore, the *Spreading Lemma* [3] ensures that continued application of the qR_3^* and qR_0 rules will generate the constraints, $(A? \geq 1, \boxdot), (B? \geq 1, \boxdot)$ and $(C? \geq 1, \boxdot)$ (i.e., all three time-points must be executed at or after 1 in *all* scenarios), which is inconsistent, for example, with the constraint $(A? \leq 0, c)$. The inconsistency is detected by the LP rule, which generates a negative self-loop with a consistent label, causing the IR-DC-checking algorithm to declare that the network is *not* IR-DC, which is wrong. □

3.4 π-Dynamic Consistency

This section summarizes the π-DC semantics introduced by Cairo *et al.* [4] that forces a strategy to select an order of dependence among simultaneous observations. The three-rule version of the IR-DC-checking algorithm will be shown to be sound and complete with respect to the π-DC semantics.

Definition 13 (Order of dependence)**.** For any scenario s, let $(P_1?, \ldots, P_k?)$ be some ordering of the observation time-points in $\mathcal{OT}$, where $k = |\mathcal{OT}|$. An *order of dependence* is any permutation π over $(1, 2, \ldots, k)$; and for each $P? \in \mathcal{OT}$, $\pi(P?) \in \{1, 2, ..., k\}$ denotes the (integer) position of $P?$ in that order. In addition, it is convenient to set $\pi(X) = \infty$ for any *non*-observation time-point X. Finally, let Π_k denote the set of all permutations over $(1, 2, \ldots, k)$.

Definition 14 (π-Execution Strategy)**.** Given any CSTN $\mathcal{S} = \langle \mathcal{T}, \mathcal{P}, \mathcal{C}, \mathcal{OT}, \mathcal{O} \rangle$, let $k = |\mathcal{OT}|$. A *π-execution strategy* for $\mathcal{S}$ is a mapping, $\sigma : \mathcal{I} \to (\Psi \times \Pi_k)$, such that for each scenario s, $\sigma(s)$ is a pair (ψ, π) where $\psi : \mathcal{T} \to \mathbb{R}$ is a schedule for the time-points in $\mathcal{T}$; and $\pi \in \Pi_k$ is a permutation that determines an order of dependence among the time-points in $\mathcal{OT}$. For any time-point X, $[\sigma(s)]_X$ denotes the execution time of X (i.e., $\psi(X)$); for any observation time-point $P?$, $[\sigma(s)]^{\pi}_{P?}$ denotes the position of $P?$ in the order of dependence (i.e., $\pi(P?)$); and for any non-observation time-point X, $[\sigma(s)]^{\pi}_X = \infty$. Finally, a π-dynamic strategy must be *coherent:* for any scenario s, and any $P?, Q? \in \mathcal{OT}$, $[\sigma(s)]_{P?} < [\sigma(s)]_{Q?}$ implies $[\sigma(s)]^{\pi}_{P?} < [\sigma(s)]^{\pi}_{Q?}$ (i.e., if $\sigma(s)$ schedules $P?$ *before* $Q?$, then it orders $P?$ before $Q?$).

Definition 15 (Viability)**.** The π-execution strategy $\sigma = (\psi, \pi)$ is called *viable* for the CSTN $\mathcal{S}$ if for each scenario s, the schedule $\psi(s)$ is a solution to the projection $\mathcal{S}(s)$.

Definition 16 (π-History)**.** Let σ be any π-execution strategy for some CSTN $\mathcal{S} = \langle \mathcal{T}, \mathcal{P}, \mathcal{C}, \mathcal{OT}, \mathcal{O} \rangle$, s any scenario, t any real number, and $d \in \{1, 2, \ldots, |\mathcal{OT}|\} \cup \{\infty\}$ any integer position (or infinity). The *π-history* of (t, d) for the scenario s and strategy σ—denoted by $\pi Hist(t, d, s, \sigma)$—is the set

$$\{(p, s(p)) \mid P? \in \mathcal{OT},\ [\sigma(s)]_{P?} \leq t, \pi(P?) < d\}.$$

[2] $(B? \geq 1)$ is an abbreviation for $(\text{Z} - B? \leq -1)$.

Thus, the π-history specifies the truth values of each proposition p that is observed *before* time t in the schedule ψ, or observed *at* time t if its corresponding observation time-point $P?$ is ordered *before* position d by the permutation π.

The following definition is equivalent to that given by Cairo *et al.* [4]. (Proof in the Appendix.) This form, and the notation it uses, was chosen to facilitate comparison with the definition of an IR-dynamic strategy (Defn. 10).

Definition 17 (π-Dynamic Strategy)**.** A π-execution strategy, σ, for a CSTN is called π-*dynamic* if for every pair of scenarios, s_1 and s_2, and every time-point $X \in \mathcal{T}$:

$$\begin{array}{ll} \text{let:} & t = [\sigma(s_1)]_X, \text{ and } d = [\sigma(s_1)]^{\pi}_X. \\ \text{if:} & \pi Hist(t, d, s_1, \sigma) = \pi Hist(t, d, s_2, \sigma) \\ \text{then:} & [\sigma(s_2)]_X = t \text{ and } [\sigma(s_2)]^{\pi}_X = d. \end{array}$$

Thus, if X is some observation time-point $P?$ that, in the scenario s_1, σ executes at time t and position d, and the histories, $\pi Hist(t, d, s_1, \sigma)$ and $\pi Hist(t, d, s_2, \sigma)$, are the same, then in the scenario s_2, σ must execute $P?$ at the same time t, and in the same position d. The requirement for a non-observation time-point X is weaker because executing X at time t does not generate any information; therefore, it can be presumed to be ordered after any observation time-points that are executed at that same time.

Definition 18 (π-Dynamic Consistency)**.** A CSTN, $\mathcal{S}$, is π-*dynamically consistent* (π-DC) if there exists a π-execution strategy for $\mathcal{S}$ that is both viable and π-dynamic.

4 Sound and Complete π-DC Checking

This section proves that the constraint-propagation rules in Table 1 are sound and complete for the π-DC semantics.

Lemma 2 (Soundness of the LP rule)**.** *If σ is a valid and π-dynamic strategy for a CSTN $\mathcal{S}$ that includes the constraints $(W - X \leq u, \alpha)$ and $(Y - W \leq v, \beta)$, where $\alpha\beta \in \mathcal{P}^*$, then σ also satisfies the constraint $(Y - X \leq u + v, \alpha\beta)$ (i.e., for each scenario s, if $s \models \alpha\beta$, then $[\sigma(s)]_Y - [\sigma(s)]_X \leq u + v$).*

Proof. Let s be any scenario for which $s \models \alpha\beta$. Since σ is valid and $s \models \alpha$, $[\sigma(s)]_W - [\sigma(s)]_X \leq u$. Similarly, since σ is valid and $s \models \beta$, $[\sigma(s)]_Y - [\sigma(s)]_W \leq v$. Summing these inequalities yields: $[\sigma(s)]_Y - [\sigma(s)]_X \leq u + v$. □

Definition 19 (π-before)**.** For a given scenario s, we say that an execution strategy σ observes p π-*before* executing Y if $[\sigma(s)]_{P?} \leq [\sigma(s)]_Y$ and $[\sigma(s)]^{\pi}_{P?} < [\sigma(s)]^{\pi}_Y$.

Intuitively, if σ observes p π-before Y, then σ's decision to execute Y can depend on the value of p. In addition, note that if $P?$ and $Q?$ are distinct, then a coherent strategy σ observes p π-before $Q?$ if and only if $[\sigma(s)]^{\pi}_{P?} < [\sigma(s)]^{\pi}_{Q?}$.

Since the qR_0 and qR^*_3 rules involve lower-bound constraints whose labels may be q-labels, we first provide a semantics for such constraints. The following definition extends that of Hunsberger *et al.* [3] to accommodate the order of dependence among simultaneous observations.

Definition 20 (Satisfying a lower-bound constraint)**.** A strategy σ satisfies the lower-bound constraint $(Y \geq \delta, \alpha)$, where $\alpha \in \mathcal{Q}^*$, if and only if for each scenario s:

1. $[\sigma(s)]_Y \geq \delta$; or
2. $\exists a \in \mathcal{P}$, such that $A? \not\equiv Y$, $\tilde{a} \in \alpha$,
   σ observes a π-before Y, and $s \not\models \tilde{a}$

(i.e., the only way $\sigma(s)$ can execute Y *before* δ is if some observation was made π-before Y that ensured that $s \not\models \alpha$).

Lemma 3 (Soundness of qR_0)**.** *If σ is a valid and π-dynamic strategy that satisfies $(P? \geq \delta, \alpha\tilde{p})$, where $\alpha \in \mathcal{Q}^*$ and $\tilde{p} \in \{p, \neg p, ?p\}$, then σ also satisfies $(P? \geq \delta, \alpha)$.*

Proof. If σ satisfies $(P? \geq \delta, \alpha\tilde{p})$, then, for any scenario s, either: (1) $[\sigma(s)]_{P?} \geq \delta$; or (2) for some $\tilde{a} \in \alpha\tilde{p}$, such that $A? \not\equiv P?$, σ observes a π-before $P?$, and $s \not\models \tilde{a}$. Since qR_0 stipulates that p does not appear in α, (1) and (2) are equivalent to the satisfaction conditions for $(P? \geq \delta, \alpha)$. □

Lemma 4 (Soundness of the qR_3^* rule)**.** *Let σ be any viable and π-dynamic strategy for a CSTN $\mathcal{S}$. If σ satisfies the constraints $(P? \geq g, \alpha)$ and $(Y \geq h, \beta\tilde{p})$, where $\alpha, \beta \in \mathcal{Q}^*$, then σ also satisfies $(Y \geq x, \alpha \star \beta)$, where $x = \min\{g, h\}$.*[3]

Proof. Suppose that the conditions of the lemma hold, but that σ does not satisfy $(Y \geq x, \alpha \star \beta)$. Then for some scenario s, the *negation* of each condition from Defn. 20 holds:

(1′) $[\sigma(s)]_Y < x = \min\{g, h\}$; **and**

(2′) for each $\tilde{a} \in \alpha \star \beta$, where $A? \not\equiv Y$:

$$[\sigma(s)]_{A?} > [\sigma(s)]_Y, \text{ or } [\sigma(s)]^\pi_{A?} \geq [\sigma(s)]^\pi_Y, \text{ or } s \models \tilde{a}.$$

Since σ satisfies $(Y \geq h, \beta\tilde{p})$, one of the following holds:

(1) $[\sigma(s)]_Y \geq h \geq x$; **or**

(2) for some $\tilde{a} \in \beta\tilde{p}$, $A? \not\equiv Y$, σ observes a π-before Y, and $s \not\models \tilde{a}$.

Now, (1) contradicts (1′). Thus (2) must hold. Now, if $\tilde{a} \in \beta$, then either $\tilde{a}$ or $?a$ is in $\alpha\star\beta$, which would contradict (2′). Thus, it must be that $\tilde{a} = \tilde{p}$, in which case, (2) becomes: $P? \not\equiv Y$, σ observes p π-before Y (i.e., $[\sigma(s)]_{P?} \leq [\sigma(s)]_Y < x \leq g$ and $[\sigma]^\pi_{P?} < [\sigma(s)]^\pi_Y$), and $s \not\models \tilde{p}$. Next, since σ satisfies $(P? \geq g, \alpha)$, but $[\sigma(s)]_{P?} < g$, the following clause from Defn. 20 must hold:

($2^\dagger$) for some $\tilde{b} \in \alpha$, $[\sigma(s)]^\pi_{B?} < [\sigma(s)]^\pi_{P?}$ and $s \not\models \tilde{b}$.

Any instance of ($2^\dagger$) would imply that σ observes b π-before $P?$ which, together with σ observing p π-before Y, implies that σ observes b π-before Y. Now, $\tilde{b} \in \alpha$ implies that either $\tilde{b}$ or $?b$ is in $\alpha\star\beta$. But then a contradiction with (2′) can only be avoided if $B? \equiv Y$. But that would imply that $[\sigma(s)]^\pi_{A?} < [\sigma(s)]^\pi_{P?} < [\sigma(s)]^\pi_{A?}$, which is a contradiction. □

Algorithm 1: π-DC-Check

Input: $\mathcal{S} = \langle \mathcal{T}, \mathcal{P}, \mathcal{C}, \mathcal{OT}, \mathcal{O} \rangle$, any (streamlined) CSTN

1 **foreach** $(X \in \mathcal{T})$ **do**
2 Insert $(X \geq 0, \boxdot)$ and $(X \leq Mn, \boxdot)$ // Mn = horizon
3 $\mathcal{C}_{\text{new}} := \emptyset$; $\mathcal{C}_{\text{prev}} := \mathcal{C}$
4 **while** *(*$\mathcal{C}_{\text{prev}} \neq \emptyset$*)* **do**
5 **foreach** *(*$(\mathrm{Z} - X \leq v, \ell) \in \mathcal{C}_{\text{prev}}$*)* **do**
6 **if** *(*$(X \equiv P?)$ *and* $(\ell = \alpha\tilde{p})$*)* **then**
7 $\mathcal{C}_{\text{new}} := \mathcal{C}_{\text{new}} \cup \mathrm{qR}_0(P?, v, \alpha, \tilde{p})$
8 **foreach** *(*$(\mathrm{Z} - Q? \leq w, \alpha) \in \mathcal{C} \mid \ell = \beta\tilde{q}$*)* **do**
9 $\mathcal{C}_{\text{new}} := \mathcal{C}_{\text{new}} \cup \mathrm{qR}_3^*(Q?, w, \alpha, v, \beta, \tilde{q}, X)$
10 **foreach** *(*$(X - Y \leq w, \ell') \in \mathcal{C}$*)* **do**
11 $\mathcal{C}_{\text{new}} := \mathcal{C}_{\text{new}} \cup \mathrm{LP}(Y, v, \ell, X, w, \ell')$
12 **if** *(*$\exists$ *a negative self-loop in* $\mathcal{C}_{\text{new}}$ *with label in* $\mathcal{P}^*$*)* **then**
13 **return** $\mathcal{S}$ *is* not π-*DC*
14 $\mathcal{C} := \mathcal{C} \cup \mathcal{C}_{\text{new}}$; $\mathcal{C}_{\text{prev}} := \mathcal{C}_{\text{new}}$; $\mathcal{C}_{\text{new}} = \emptyset$
15 **return** $\mathcal{S}$ *is* π-*DC*

4.1 The π-DC-Checking Algorithm for CSTNs

Unlike the 6-rule IR-DC-checking algorithm, our new π-DC-checking algorithm, whose pseudo-code is shown in Algorithm 1, (1) uses only the three rules from Table 1; and (2) only generates edges terminating at Z. It begins (Lines 1–2) by constraining each time-point X thusly: $0 \leq X \leq h = Mn$, where M is the maximum absolute value of any negative weight in the CSTN, and $n = |\mathcal{T}|$. Cairo *et al.* [11] proved that such constraints preserve the DC property.

Each iteration of the main loop (Lines 5–13) processes each constraint $C \in \mathcal{C}_{\text{prev}}$ generated by the *previous* iteration, checking for possible applications of qR_0 (Lines 6–7), qR_3^* (Lines 8–9), and LP (Lines 10–11). Only constraints that are not entailed by constraints already in $\mathcal{C}$ are collected in $\mathcal{C}_{\text{new}}$. The main loop ends when: (1) a negative self-loop with a consistent label is found (Line 12); or (2) no new constraints were generated by the current iteration (Line 4). The algorithm reports *not* π-DC in the first case; π-DC in the second.

4.2 Completeness of π-DC Checking

⇒ The approach followed below mirrors that in Hunsberger *et al.* [3], generalized to accommodate π-dynamicity.

At any point during the execution of a CSTN, the observations that have been made so far can be represented by a label $\ell \in \mathcal{P}^*$ that is equivalent to the π-history at that point. For convenience, such a label is called a *current partial scenario* (CPS). A q-label $\alpha \in \mathcal{Q}^*$ is called *applicable* in a given CPS ℓ, if the observations in ℓ are consistent with α.

[3] For convenience, the following substitutions have been made from Table 1: $g = -w$; $h = -v$; and $x = -m$.

Definition 21 (Applicable q-Label)**.** A q-label α is *applicable* with respect to a given CPS ℓ if the following condition holds: whenever any letter p appears in both ℓ and α, it appears in the same form (i.e., as p in both, or as $\neg p$ in both).

Note that if $?p \in \alpha$, then $appl(\alpha, \ell)$ only holds if p has not yet been observed (i.e., if p does not appear in ℓ).

Lemma 5. *If σ is a viable and π-dynamic strategy for some CSTN $\mathcal{S}$, then σ satisfies the constraint $(X \geq \delta, \alpha)$ if and only if for each scenario s, $[\sigma(s)]_X < \delta \Rightarrow \neg appl(\alpha, \ell)$, where $\ell = \pi Hist([\sigma(s)]_X, [\sigma(s)]^{\pi}_X, s, \sigma)$ is the relevant CPS.*

Proof. Suppose $[\sigma(s)]_X < \delta$. Since σ satisfies $(X \geq \delta, \alpha)$, there must be some a such that $\tilde{a} \in \alpha$, $A? \not\equiv X$, σ observes a π-before X, and $s \not\models \tilde{a}$. Since σ observes a π-before X, then either a or $\neg a$ appears in the CPS ℓ. Now, if $\tilde{a} = a$, then $a \in \alpha$, but $s \not\models a$ implies that $\neg a \in \ell$; and, hence, $\neg appl(\alpha, \ell)$. The case, $\tilde{a} = \neg a$, is similar. Finally, if $\tilde{a} = ?a$, then $\tilde{a} \in \alpha$ and a appearing in ℓ implies that $\neg appl(\alpha, \ell)$. □

Lemma 6 (Spreading Lemma)**.** *Let $\mathcal{S}$ be any CSTN whose set of constraints $\mathcal{C}$ is closed under the* qR_0 *and* qR^*_3 *rules. Let $\ell \in \mathcal{P}^*$ be any consistent label, representing a possible CPS. Let $\mathcal{T}_\ell = \{P? \mid p \text{ appears in } \ell\}$ be the observation time-points corresponding to the observations in ℓ. Let $\mathcal{T}_x$ be any set of time-points such that $\mathcal{T}_\ell \subseteq \mathcal{T}_x \subset \mathcal{T}$, and $\mathcal{T}_x \cap \mathcal{OT} = \mathcal{T}_\ell$. ($\mathcal{T}_x$ represents the time-points that have been executed so far. Note that the observation time-points in $\mathcal{T}_x$ are precisely those in $\mathcal{T}_\ell$; thus, ℓ is a possible CPS for the case where the observation time-points in $\mathcal{T}_x$ have been executed.) And let $\mathcal{T}_u = \mathcal{T} \setminus \mathcal{T}_x$ be the as-yet-unexecuted time-points.*

Now for each $X \in \mathcal{T}_u$, let:

- $ELB(X, \ell) = \max\{\delta \mid \exists \alpha \in \mathcal{Q}^*$ *and* $(X \geq \delta, \alpha) \in \mathcal{C}$ *such that* $appl(\alpha, \ell)\}$*; and*
- $\Lambda(\ell, \mathcal{T}_u) = \min\{ELB(X, \ell) \mid X \in \mathcal{T}_u\}$.

$ELB(X, \ell)$ is the effective lower bound *for X specified by constraints in $\mathcal{C}$ that are applicable in the CPS ℓ, and $\Lambda(\ell, \mathcal{T}_u)$ is the minimum ELB among unexecuted time-points.*

Given all of the above, for each $X \in \mathcal{T}_u$, the constraint $(X \geq \Lambda(\ell, \mathcal{T}_u), \ell)$ is entailed by the constraints in $\mathcal{C}$.

Proof. Given the premise, let Λ denote $\Lambda(\ell, \mathcal{T}_u)$, let $ELB(X)$ denote $ELB(X, \ell)$, and let $\mathcal{P}_u = \{p \in \mathcal{P} \mid P? \in \mathcal{T}_u\}$ be the propositional letters that have not yet been observed. By construction, for each $p \in \mathcal{P}_u$, $ELB(P?) \geq \Lambda$; hence there must be a constraint, $(P? \geq \lambda_p, \alpha_p) \in \mathcal{C}$, for some $\lambda_p \geq \Lambda$ and $\alpha_p \in \mathcal{Q}^*$ such that $appl(\alpha_p, \ell)$. Now, if p appears in α_p, then qR_0 can remove it from α_p; thus, since $\mathcal{C}$ is closed under qR_0, it can be assumed that p does not appear in α_p.

Next, let r be any other letter in $\mathcal{P}_u$. Again, there must be a constraint, $(R? \geq \lambda_r, \alpha_r)$, for some $\lambda_r \geq \Lambda$, and $r \in \mathcal{Q}^*$ such that $appl(\alpha_r, \ell)$. Now, if p appears in r, then it can be removed using qR^*_3, generating $(R? \geq \lambda_{pr}, \alpha_{pr})$, where $\lambda_{pr} = \min\{\lambda_p, \lambda_r\} \geq \Lambda$, and $\alpha_{pr} = \alpha_p \star \alpha'_r$, where α'_r is obtained by removing any $\tilde{p}$ from α_r. Since $appl(\alpha_p, \ell)$ and $appl(\alpha_r, \ell)$, it follows that $appl(\alpha_{pr}, \ell)$. As before, since $\mathcal{C}$ is closed under qR_0 and qR^*_3, this constraint must be entailed by constraints in $\mathcal{C}$. Thus, we can assume that p does not appear in α_r. Continuing

Algorithm 2: EarliestFirstExecutionStrategy($\mathcal{S}$)

Input: $\mathcal{S} = \langle \mathcal{T}, \mathcal{P}, \mathcal{C}, \mathcal{OT}, \mathcal{O} \rangle$, a CSTN
$t := 0, \quad d := 1$ // current time & dependency position
$\ell := \boxdot$ // current partial scenario
$[\sigma(s)]_{\mathrm{Z}} := 0$ // execute Z at 0
$\mathcal{T}_u := \mathcal{T} \setminus \{\mathrm{Z}\}$
while *($\mathcal{T}_u \neq \emptyset$)* **do**
  $t := \Lambda(\ell, \mathcal{T}_u)$ // as in Lemma 6
  $\chi := \{X \in \mathcal{T}_u \mid ELB(X, \ell) = t\}$ // as in Lemma 6
  foreach *($X \in \chi$)* **do**
    $[\sigma(s)]_X := t$ // execute X at t
    $\mathcal{T}_u := \mathcal{T}_u \setminus \{X\}$
    if *($X \in \mathcal{OT}$)* **then**
      $[\sigma(s)]^{\pi}_X := d$
      $d := d + 1$
      $\ell := \ell \wedge s(x)$ // record observation
return $\langle \sigma(s), s \rangle$ // $\ell = s$ at end

in this way, each observation time-point $T? \in \mathcal{T}_u$ must have a corresponding constraint $(T? \geq \lambda_t, \alpha_t)$ where $\lambda_t \geq \Lambda$ and α_t does not include p.

Once p has been removed from all such labels, it follows that r can similarly be removed from all such labels, and so on, until each observation time-point $T? \in \mathcal{T}_u$ is seen to have a lower-bound constraint, $(T? \geq \lambda_t, \alpha_t)$, where $\lambda_t \geq \Lambda$ and α_t contains no propositional letters from $\mathcal{P}_u$. As a result, each label α_t can only have letters that appear in ℓ; and, since $appl(\alpha_t, \ell)$ holds, it follows that $\ell \models \alpha_t$. Finally, qR^*_3 can be used to similarly process the labels from lower-bound constraints on all $X \in \mathcal{T}_u$, removing any occurrences of letters in $\mathcal{P}_u$. Thus, for each $X \in \mathcal{T}_u$, the constraint $(X \geq \Lambda, \ell)$ must be entailed by constraints in $\mathcal{C}$. □

Lemma 7. *Let $\mathcal{S}$ be any CSTN, s any scenario, and $\mathcal{S}(s)$ the corresponding projection STN. If P is a path from X to Y of length δ in $\mathcal{S}(s)$, then there is a corresponding path P' in $\mathcal{S}$ from X to Y whose length is δ and whose edges have labels that are consistent with s. In addition, if the edges in $\mathcal{S}$ are closed under the* LP *rule, then there is a single edge in $\mathcal{S}$ from X to Y of length δ whose label is consistent with s.*

Proof. Follows from Defn. 5 and the LP rule. □

Theorem 2 (Correctness of π-DC checking)**.** *Algorithm 1 is sound and complete for π-DC checking for CSTNs with rational weights; and is guaranteed to terminate.*

Proof. Soundness follows from Lemmas 2–4. Termination is guaranteed by the horizon constraints: for each $X \in \mathcal{T}$, $0 \leq X \leq h$. A finite number of rational weights can be put over a common denominator D, yielding, in effect, an integer domain. Because there are n^2 edges, each with at most 3^k different q-labels and D different numerical weights, the algorithm generates at most $n^2(3^k)D$ incremental updates.

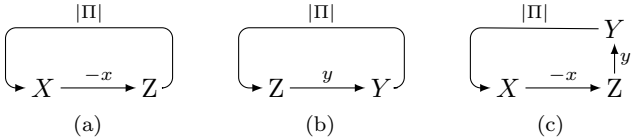


Figure 4: Loops discussed in proof of Theorem 2

Next, let $\mathcal{S}$ be any CSTN that the algorithm says is π-DC. Then the fully propagated CSTN must be closed under the LP, qR_0 and qR_3^* rules. Henceforth, $\mathcal{S}$ shall refer to the fully propagated CSTN.

Let σ be the earliest-first execution strategy (Algorithm 2). This strategy is computed in real time, as the execution of observation time-points incrementally reveals the scenario s.

• *σ is π-dynamic.* By construction, with each observation, the *ELB* values and, hence, $t = \Lambda(\ell, \mathcal{T}_u)$ can never decrease; and d never decreases. Thus, each execution decision depends only on the relevant π-history, as required by Defn. 17.

• *For any scenario s, the projection $\mathcal{S}(s)$ is consistent.* If $\mathcal{S}(s)$ contains a negative loop, then Lemma 7 implies there is a single-edge negative loop in $\mathcal{S}$ whose label is consistent with s, contradicting the algorithm's report that $\mathcal{S}$ is π-DC.

• *σ is valid.* Let s be any scenario, and $\mathcal{S}(s)$ the corresponding projection. We must show that the schedule $\sigma(s)$ is a solution to the STN $\mathcal{S}(s)$. Suppose not.

For each X, let $x = [\sigma(s)]_X$. The corresponding execution constraints are $(\text{Z} - X \leq x)$ and $(X - \text{Z} \leq x)$ (i.e., $X = x$). Since $\sigma(s)$ is not a solution, inserting these constraints into $\mathcal{S}(s)$ must create a negative loop—call it L. Without loss of generality, L has only one occurrence of Z.

Case 1. In this case, illustrated in Fig. 4a, L consists of the lower-bound execution constraint $(\text{Z} - X \leq -x)$ followed by a path Π from Z back to X, where: (1) $x = \Lambda(\ell, \mathcal{T}_u) = \mathit{ELB}(X, \ell)$, where ℓ is the CPS at the moment X was executed; (2) $|\Pi| < x$, and (3) the edges in Π are from $\mathcal{S}(s)$. By Lemma 7, there must be a single edge in $\mathcal{S}$ from Z to X of length $|\Pi|$ whose label is consistent with s. Next, the Spreading Lemma ensures that the constraint/edge, $(\text{Z} - X \leq -x, \ell)$, is entailed by constraints in the propagated CSTN. And ℓ is necessarily consistent with s. Applying the LP rule to these two edges would yield a single-edge negative loop in $\mathcal{S}$ with a consistent label, causing the algorithm to report that $\mathcal{S}$ was not π-DC, a contradiction.

Case 2. In this case, illustrated in Fig. 4b, L consists of the constraint $(Y - \text{Z} \leq y)$ followed by a path Π from Y back to Z, where: (1) $y = \Lambda(\ell, \mathcal{T}_u) = \mathit{ELB}(Y, \ell)$, where ℓ is the CPS when Y was executed; (2) the edges in Π are from $\mathcal{S}(s)$; and (3) $|\Pi| < -y$. By Lemma 7, there must be a single edge in $\mathcal{S}$ from Y to Z of length $|\Pi|$ whose label is consistent with s. But then $-|\Pi| \leq \mathit{ELB}(Y, \ell) = y$, by the definition of $\mathit{ELB}(Y, \ell)$, which contradicts that $|\Pi| < -y$.

Case 3. In this case, illustrated in Fig. 4c, L consists of a lower-bound edge $(\text{Z} - X \leq -x)$, followed by an upper-bound edge $(Y - \text{Z} \leq y)$, followed by a path Π from Y to X with edges in $\mathcal{S}(s)$. Here, $|\Pi| - x + y < 0$. By Lemma 7, there is a single edge Π' in $\mathcal{S}$ of length $|\Pi|$ whose label is consistent with s. By Case 1, the lower-bound constraint for X is entailed by constraints in $\mathcal{S}$. Thus, applying the LP rule to Π' and the lower-bound edge for X would have yielded

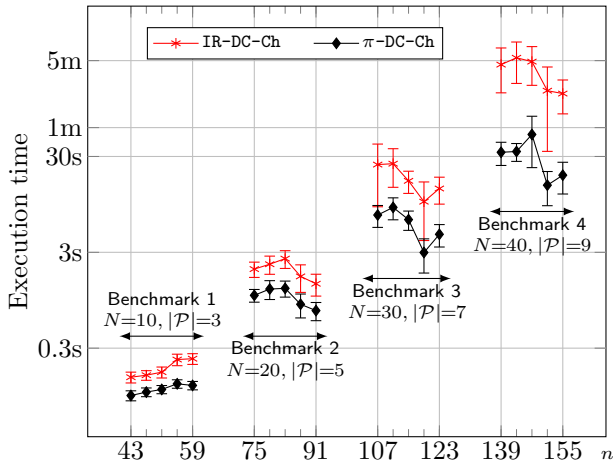


Figure 5: Execution time vs. number of time-points n

an edge, $(Z - Y \leq |\Pi| - x, \alpha)$ in $\mathcal{S}$, where α is consistent with s. So $-|\Pi| + x$ must be a relevant lower bound for Y. Hence, $ELB(Y, \ell_y) \geq -|\Pi| + x > y$, a contradiction. □

5 Empirical Evaluation

This section compares the performance of the π-DC-checking and IR-DC-checking algorithms. (The results in this paper imply that the 6-rule IR-DC algorithm is also sound and complete for π-DC checking, although not for IR-DC checking.) `π-DC-Ch` is our implementation of Algorithm 1; `IR-DC-Ch` is the freely available implementation of the IR-DC-checking algorithm [12]. Algorithms and procedures were implemented in Java and executed on a JVM 8 in a Linux box with two AMD Opteron 4334 CPUs and 64GB of RAM.

Both implementations were tested on instances of the four benchmarks from Hunsberger and Posenato [7]. Each benchmark has at least 60 DC and 60 non-DC CSTNs, obtained from random workflow schemata generated by the ATAPIS toolset [13]. The numbers of activites (N) and observations ($|\mathcal{P}|$) were varied, as shown in Fig. 5. Since non-DC networks were regularly solved one to two orders of magnitude faster than similarly sized DC networks, this section focuses on the results for the DC networks.

Fig. 5 displays the average execution times of the two algorithms over all four benchmarks, each point representing the average execution time for instances of the given size. We extended the benchmarks, adding up to 50 DC instances, to generate tight 95% confidence intervals. The results demonstrate that the 3-rule `π-DC-Ch` algorithm is much faster than the 6-rule `IR-DC-Ch` algorithm. Moreover, the better performance increases as the size of the instances increases.

6 Conclusions

This paper showed that the original analysis of the IR-DC-checking algorithm with respect to the IR-DC semantics was flawed. However, it showed that the

IR-DC-checking algorithm is sound-and-complete with respect to the π-DC semantics that properly captures instantaneous reaction. Furthermore, it proved that only three of the six propagation rules are needed to ensure completeness. And, since no proper subset of the three rules will suffice, the three rules represent a *minimal* sound-and-complete set of rules, an important theoretical result. The experimental evaluation indicated that the new algorithm is faster than the IR-DC-checking algorithm.

References

[1] I. Tsamardinos, T. Vidal, and M. E. Pollack, "CTP: A new constraint-based formalism for conditional, temporal planning," *Constraints*, vol. 8, pp. 365–388, 2003. doi:10.1023/A:1025894003623.

[2] C. Comin and R. Rizzi, "Dynamic consistency of conditional simple temporal networks via mean payoff games: a singly-exponential time dc-checking," in *22st International Symposium on Temporal Representation and Reasoning (TIME 2015)*, pp. 19–28, IEEE, Sept. 2015. doi:10.1109/TIME.2015.18.

[3] L. Hunsberger, R. Posenato, and C. Combi, "A sound-and-complete propagation-based algorithm for checking the dynamic consistency of conditional simple temporal networks," in *22st International Symposium on Temporal Representation and Reasoning (TIME 2015)*, pp. 4–18, IEEE, Sept. 2015. doi:10.1109/TIME.2015.26.

[4] M. Cairo, C. Comin, and R. Rizzi, "Instantaneous reaction-time in dynamic-consistency checking of conditional simple temporal networks," in *TIME 2016*, Oct. 2016.

[5] A. Cimatti, L. Hunsberger, A. Micheli, R. Posenato, and M. Roveri, "Sound and complete algorithms for checking the dynamic controllability of temporal networks with uncertainty, disjunction and observation," in *21st International Symposium on Temporal Representation and Reasoning (TIME 2014)*, pp. 27–36, IEEE, Sept. 2014. doi:10.1109/TIME.2014.21.

[6] A. Cimatti, L. Hunsberger, A. Micheli, R. Posenato, and M. Roveri, "Dynamic controllability via Timed Game Automata," *Acta Informatica*, vol. 53, no. 6-8, pp. 681–722, 2016. doi:10.1007/s00236-016-0257-2.

[7] L. Hunsberger and R. Posenato, "Checking the Dynamic Consistency of Conditional Temporal Networks with Bounded Reaction Times," in *Proceedings of the 26th International Conference on Automated Planning and Scheduling, ICAPS 2016*, pp. 175–183, 2016. URL: http://www.aaai.org/ocs/index.php/ICAPS/ICAPS16/paper/view/13108.

[8] R. Dechter, I. Meiri, and J. Pearl, "Temporal constraint networks," *Artificial Intelligence*, vol. 49, no. 1-3, pp. 61–95, 1991. doi:10.1016/0004-3702(91)90006-6.

[9] L. Hunsberger, R. Posenato, and C. Combi, "The Dynamic Controllability of Conditional STNs with Uncertainty," in *PlanEx at ICAPS 2012*, pp. 1–8, June 2012.

[10] C. Combi, L. Hunsberger, and R. Posenato, "An algorithm for checking the dynamic controllability of a conditional simple temporal network with uncertainty," in *Proceedings of the 5th International Conference on Agents and Artificial Intelligence (ICAART-2013)*, vol. 2, pp. 144–156, Feb. 2013.

[11] M. Cairo, L. Hunsberger, R. Posenato, and R. Rizzi, "A Streamlined Model of Conditional Simple Temporal Networks - Semantics and Equivalence Results," in *24th International Symposium on Temporal Representation and Reasoning (TIME 2017)*, vol. 90 of *LIPIcs*, pp. 10:1–10:19, 2017. doi:10.4230/LIPIcs.TIME.2017.10.

[12] R. Posenato, "A CSTN(U) consistency check algorithm implementation in Java. version 1.22." http://profs.scienze.univr.it/∼posenato/software/cstnu, Nov. 2017.

[13] A. Lanz and M. Reichert, "Enabling time-aware process support with the atapis toolset," in *Proceedings of the BPM Demo Sessions 2014* (L. Limonad and B. Weber, eds.), vol. 1295 of *CEUR Workshop Proceedings*, pp. 41–45, 2014.

7 Appendix

Here is the definition of π-dynamic strategy given by Cairo *et al.* [4].

Definition 22 (π-Dynamic Strategy)**.** A π-execution strategy, σ, for a CSTN is called π-*dynamic* if for every pair of scenarios, s_1 and s_2, and every time-point $X \in \mathcal{T}$:

let: $t = [\sigma(s_1)]_X$, and $d = \begin{cases} [\sigma(s_1)]^\pi_X, & \text{if } X \in \mathcal{OT}; \\ \infty, & \text{otherwise.} \end{cases}$

if: $\text{Cons}(\pi(t, d, s_1, \sigma), s_2)$

then: $[\sigma(s_2)]_X = t$ and, if $X \in \mathcal{OT}$, $[\sigma(s_2)]^\pi_X = d$.

Note that for a scenario such as s_2, $\text{Cons}(\pi(t, d, s_1, \sigma), s_2)$ is equivalent to $s_2 \models \pi Hist(t, d, s_1, \sigma)$.

Theorem 3. *The definitions of π-dynamic strategy (i.e., Defns. 17 and 22) are equivalent.*

Proof. ($\Rightarrow$). Suppose that σ satisfies Defn. 17, but not Defn. 22. Then for some scenarios s_1 and s_2, and some time-point X, where $t = [\sigma(s_1)]_X$ and $d = [\sigma(s_1)]^\pi_X$, $s_2 \models \pi Hist(t, d, s_1, \sigma)$, but $[\sigma(s_2)]_X \neq t$ or $[\sigma(s_2)]^\pi_X \neq d$. With no loss of generality, choose the circumstance of this sort for which t is minimal and, for that t, d is minimal.

Next, for $\sigma(s_1)$ and $\sigma(s_2)$, consider the corresponding schedules of time-point executions, where each observation time-point is annotated with the boolean value specified by the given scenario, and where any simultaneous observations are listed in the order of dependence specified by $\sigma(s_1)$ or $\sigma(s_2)$, respectively. By construction, these two annotated schedules—call them $\sigma(s_1)^+$ and $\sigma(s_2)^+$, differ at time t. Let $t^* \leq t$ be the earliest time at which $\sigma(s_1)^+ \neq \sigma(s_2)^+$.

Case 1: The first difference involves an observation time-point $Q?$ for which $[\sigma(s_1)]_{Q?} = t^* = [\sigma(s_2)]_{Q?}$ and $[\sigma(s_1)]^\pi_{Q?} = d^* = [\sigma(s_2)]^\pi_{Q?}$, but but $s_1(q) \neq s_2(q)$. Now suppose that $t^* < t$. But then $(q, s_1(q)) \in \pi Hist(t, d, s_1, \sigma)$, in which case,

$s_2 \models \pi Hist(t, d, s_1, \sigma)$ would contradict that $s_1(q) \neq s_2(q)$. Thus, $t^* = t$. Similarly, given that $t^* = t$, if $d^* < d$, then $(q, s_1(q)) \in \pi Hist(t, d, s_1, \sigma)$, leading to the same contradiction. Thus, $d = d^*$. But then $\pi Hist(t, d, s_1, \sigma) = \pi Hist(t, d, s_2, \sigma)$ which, since σ satisfies Defn. 17, implies that $[\sigma(s_2)]_X = t$ and $[\sigma(s_2)]^\pi_X = d$, a contradiction.

Case 2: The first difference between $\sigma(s_1)^+$ and $\sigma(s_2)^+$ involves the execution of some time-point Y at time t^* and position d^* by $\sigma(s_i)$, but not by $\sigma(s_j)$, where $\{s_i, s_j\} = \{s_1, s_2\}$. But then $\pi Hist(t^*, d^*, s_1, \sigma) = \pi Hist(t^*, d^*, s_2, \sigma)$ and, since σ satisfies Defn. 17, $[\sigma(s_2)]_Y = t^*$ and $[\sigma(s_2)]_Y = d^*$, a contradiction.

($\Leftarrow$). Suppose that σ satisfies Defn. 22, but not Defn. 17. Thus, there must be some scenarios s_1 and s_2, and time-point X, such that $\pi Hist(t, d, s_1, \sigma) = \pi Hist(t, d, s_2, Hist)$, where $t = [\sigma(s_1)]_X$ and $d = [\sigma(s_1)]^\pi_X$, but $[\sigma(s_2)]_X \neq t$ or $[\sigma(s_2)]^\pi_X \neq d$. As above, choose t minimal with this property and, for that t, choose d minimal. Now, $s_2 \models \pi Hist(t, d, s_2, \sigma)$ since the history contains observations determined by s_2. But then $\pi Hist(t, d, s_1, \sigma) = \pi Hist(t, d, s_2, Hist)$ implies that $s_2 \models \pi Hist(t, d, s_1, \sigma)$ which, since σ satisfies Defn. 22, implies that $[\sigma(s_2)]_X = t$ and $[\sigma(s_2)]^\pi_X = d$, a contradiction. □

UNIVERSITÀ
di **VERONA**

Dipartimento
di **INFORMATICA**

University of Verona
Department of Computer Science
Strada Le Grazie, 15
I-37134 Verona
Italy

http://www.di.univr.it