On Restricted Disjunctive Temporal Problems: Faster Algorithms and Tractability Frontier

Carlo Comin

Department of Computer Science, University of Verona, Italy
carlo.comin.86@gmail.com
 https://orcid.org/0000-0001-5748-2029

Romeo Rizzi

Department of Computer Science, University of Verona, Italy
romeo.rizzi@univr.it
 https://orcid.org/0000-0002-2387-0952

— Abstract —

In 2005 T.K.S. Kumar studied the Restricted Disjunctive Temporal Problem (RDTP), a restricted but very expressive class of Disjunctive Temporal Problems (DTPs). An RDTP comes with a finite set of temporal variables, and a finite set of temporal constraints each of which can be either one of the following three types: ($\mathtt{t}_1$) two-variable linear-difference simple constraint; ($\mathtt{t}_2$) single-variable disjunction of many interval constraints; ($\mathtt{t}_3$) two-variable disjunction of two interval constraints only. Kumar showed that RDTPs are solvable in deterministic strongly polynomial time by reducing them to the Connected Row-Convex (CRC) constraints satisfaction problem, also devising a faster randomized algorithm. Instead, the most general form of DTPs allows for multi-variable disjunctions of many interval constraints and it is NP-complete.

This work offers a deeper comprehension on the tractability of RDTPs, leading to an elementary deterministic strongly polynomial time algorithm for them, significantly improving the asymptotic running times of all the previous deterministic and randomized solutions. The result is obtained by reducing RDTPs to the Single-Source Shortest Paths (SSSP) and the 2-SAT problem (jointly), instead of reducing to CRCs. In passing, we obtain a faster (quadratic time) algorithm for RDTPs having only $\{\mathtt{t}_1, \mathtt{t}_2\}$-constraints and no $\mathtt{t}_3$-constraint. As a second main contribution, we study the tractability frontier of solving RDTPs blended with Hyper Temporal Networks (HyTNs), a disjunctive strict generalization of Simple Temporal Networks (STNs) based on hypergraphs: we prove that solving temporal problems having only $\mathtt{t}_2$-constraints and either only multi-tail or only multi-head hyperarc-constraints lies in NP ∩ co-NP and admits deterministic pseudo-polynomial time algorithms; on the other hand, problems having only $\mathtt{t}_3$-constraints and either only multi-tail or only multi-head hyperarc-constraints turns out strongly NP-complete.

1 Introduction

Expressive and efficient temporal reasoning is essential to a number of areas in Artificial Intelligence (AI) [8, 14, 15]. Over the past few years, many constraint-based formalisms have been developed to represent and reason about time in automated planning and temporal scheduling [4, 12]. We begin by recalling the Disjunctive Temporal Problem (DTP) [13, 16, 17]. The general form of a DTP being, given a finite set $\mathcal{T} = \{X_0, X_1, \ldots, X_N\}$ of temporal

variables (i.e., time-points), to schedule them on the real line in such a way as to satisfy a prescribed finite set $\mathcal{C}$ of temporal constraints over $\mathcal{T}$. Every constraint $c_i \in \mathcal{C}$ is a disjunction of the form $s_{(i,1)} \vee s_{(i,2)} \vee \cdots \vee s_{(i,T_i)}$, where every $s_{i,j}$ is a simple temporal constraint of the form $(l_{i,j} \leq X_{\beta_{i,j}} - X_{\alpha_{i,j}} \leq u_{i,j})$ for some integers $0 \leq \alpha_{i,j}, \beta_{i,j} \leq N$ and reals $l_{i,j}, u_{i,j}$.

Although DTPs are expressive enough to capture many tasks in automated planning and temporal scheduling, they are NP-complete [16]. The principal direct approach taken to solve DTPs has been to convert the original problem into one of selecting a disjunct from each constraint [16, 17], then to check whether the set of selected disjuncts forms a consistent Simple Temporal Problem (STP) [5]. This can be done in strongly polynomial time by computing single-source shortest paths (e.g., with the Bellman-Ford's algorithm [2]). Under this prospect, of course the prohibitive complexity of solving DTPs comes from the fact that there are exponentially many disjunct combinations possible.

In [9, 10], T.K.S. Kumar studied the Restricted Disjunctive Temporal Problem (RDTP), a tractable subclass of DTPs strictly including the classical and well established STPs [5]. In RDTPs, each constraint can be either one of the following three types: $(\mathtt{t_1})$ $(Y - X \leq w)$, for w real (a simple temporal difference-constraint); $(\mathtt{t_2})$ $(l_1 \leq X \leq u_1) \vee \cdots \vee (l_k \leq X \leq u_k)$, for l_i, u_i reals (a single-variable disjunction of many interval-constraints); $(\mathtt{t_3})$ $(l_1 \leq X \leq u_1) \vee (l_2 \leq Y \leq u_2)$, for l_i, u_i reals (a two-variable disjunction of two interval-constraints).

It was shown in [10] that RDTPs are solvable in deterministic strongly polynomial time by reducing them to the Connected Row-Convex (CRC) [6] constraint satisfaction problem, faster randomized algorithms were also proposed. CRC constraints generalize many other known tractable classes of constraints like 2-SAT, implicational, and binary integer-weighted linear constraints [6]. Particularly, Kumar's deterministic algorithm for solving RDTPs works by reducing them into binary Constraint Satisfiability Problems (CSPs) over meta-variables representing $\mathtt{t_2}$ or $\mathtt{t_3}$ constraints, meanwhile showing that such binary constraints are indeed CRC constraints, finally exploiting the algorithmic tractability of CRC constraints.

An instantiation of a consistency checking algorithm (e.g., [6]) that further exploits the structure of CRC constraints leads to a time complexity of $O\big((|\mathcal{C}_{\mathtt{t_2}}| + |\mathcal{C}_{\mathtt{t_3}}|)^3 \cdot d_{\max}^2 + |\mathcal{T}| \cdot |\mathcal{C}_{\mathtt{t_1}}| \cdot (|\mathcal{C}_{\mathtt{t_2}}| + |\mathcal{C}_{\mathtt{t_3}}|)^2\big)$, where $\mathcal{C}_{\mathtt{t_1},\mathtt{t_2},\mathtt{t_3}}$ is the set of $\mathtt{t_1}, \mathtt{t_2}, \mathtt{t_3}$ constraints (respectively), and $d_{\max}$ is the maximum number of disjuncts possible per single constraint [10]. Randomization reduces the running time to $O\big((|\mathcal{C}_{\mathtt{t_2}}| + |\mathcal{C}_{\mathtt{t_3}}|)^2 \cdot d_{\max}^2 \cdot \delta + |\mathcal{T}| \cdot |\mathcal{C}_{\mathtt{t_1}}| \cdot (|\mathcal{C}_{\mathtt{t_2}}| + |\mathcal{C}_{\mathtt{t_3}}|)^2\big)$, where δ is the degree of the CRC network (i.e., the maximum number of constraints any variable participates into) [10].

Notable applications of RDTPs include solving STPs with Taboo Regions, cfr. [11].

Contributions.   This work offers a deeper comprehension on the tractability of RDTPs, leading to elementary deterministic strongly polynomial time algorithms, significantly improving the asymptotic running times of both the Kumar's deterministic and randomized solutions. Our time complexity is $O\big(|\mathcal{T}| \cdot |\mathcal{C}_{\mathtt{t_1}}| + |\mathcal{C}_{\mathtt{t_2}}| \cdot (|\mathcal{C}_{\mathtt{t_1}}| + |\mathcal{T}| \cdot \log |\mathcal{T}|) + |\mathcal{T}| \cdot d_{\mathcal{C}_{\mathtt{t_2}}} \cdot |\mathcal{C}_{\mathtt{t_3}}| + |\mathcal{C}_{\mathtt{t_3}}|^2\big)$, where $d_{\mathcal{C}_{\mathtt{t_2}}}$ is the total number of disjuncts counting over all $\mathtt{t_2}$-constraints. Since $d_{\mathcal{C}_{\mathtt{t_2}}} \leq d_{\max} \cdot |\mathcal{C}_{\mathtt{t_2}}|$, this improves over all of the previous solutions. The result is obtained by reducing RDTPs to the Single-Source Shortest Paths (SSSP) and the 2-SAT problem (jointly), instead of reducing to CRCs. So the full expressive power of CRCs is not needed, binary linear and 2-SAT constraints are enough. In passing, we obtain a faster (quadratic time) deterministic algorithm for solving temporal problems having only $\{\mathtt{t_1}, \mathtt{t_2}\}$-constraints and no $\mathtt{t_3}$-constraint.

As a second main contribution, we study the tractability frontier of RDTPs widened with another kind of restricted disjunctive constraints, i.e., Hyper Temporal Networks (HyTNs) [3], a strict generalization of STNs grounded on directed hypergraphs and introduced to overcome

🟨 **Table 1** Summary of main results.

Problem	Complexity	Improved Time Bound	Cfr.
$\mathtt{t}_2$DTPs	P	$O\big(|\mathcal{T}| \cdot |\mathcal{C}_{\mathtt{t}_1}| + |\mathcal{C}_{\mathtt{t}_2}| \cdot (|\mathcal{C}_{\mathtt{t}_1}| + |\mathcal{T}| \cdot \log|\mathcal{T}|) + |\mathcal{T}| \cdot d_{\mathcal{C}_{\mathtt{t}_2}}\big)$	Sect. 3
RDTPs	P	$O\big(|\mathcal{T}| \cdot |\mathcal{C}_{\mathtt{t}_1}| + |\mathcal{C}_{\mathtt{t}_2}| \cdot (|\mathcal{C}_{\mathtt{t}_1}| + |\mathcal{T}| \cdot \log|\mathcal{T}|) +$ $+ |\mathcal{T}| \cdot d_{\mathcal{C}_{\mathtt{t}_2}} \cdot |\mathcal{C}_{\mathtt{t}_3}| + |\mathcal{C}_{\mathtt{t}_3}|^2\big)$	Sect. 4
$\mathtt{t}_2$HyTPs	NP ∩ co-NP	$O\big((|\mathcal{T}| + |\mathcal{A}|) \cdot m_{\mathcal{A}} \cdot W_{\mathcal{A},\mathcal{C}_{\mathtt{t}_2}}\big)$	Sect. 6
$\mathtt{t}_3$HyTPs	NP-complete	*n.a. (exponential time)*	Sect. 5

the limitation of considering only conjunctions of constraints but maintaining a practical efficiency in the consistency check of the instances. In a HyTN a single temporal multi-tail (or multi-head) hyperarc-constraint is defined as a set of two or more maximum delay (minimum anticipation, respectively) constraints which is satisfied when at least one of these delay constraints is so. We prove that solving temporal problems having only $\mathtt{t}_2$-constraints and either only multi-tail or only multi-head hyperarc-constraints lies in NP ∩ co-NP and admits deterministic pseudo-polynomial time algorithms; on the other hand, solving temporal problems having only $\mathtt{t}_3$-constraints and either only multi-tail or only multi-head hyperarc-constraints turns out strongly NP-complete. See Table 1 for a summary.

🟨 2   Background

This section offers the basic background notions that are assumed in the rest of the paper, let's start with Simple Temporal Networks (STNs) and related problems (STPs), cfr. [4, 5].

▶ **Definition 1** (STNs, STPs [4, 5]). A *Simple Temporal Network* (STN) is a pair $(\mathcal{T}, \mathcal{C})$, where $\mathcal{T}$ is a set of real-valued variables called *time-points,* and $\mathcal{C}$ is a set of linear real-weighted binary constraints over $\mathcal{T}$ called *simple (or $\mathtt{t}_1$)* temporal constraints, each having the form:

$$(Y - X \leq w_{X,Y}), \text{ where } X, Y \in \mathcal{T} \text{ and } w_{X,Y} \in \mathbb{R}.$$

An STN is *consistent* if it admits a *feasible schedule*, i.e., some $s : \mathcal{T} \mapsto \mathbb{R}$ such that $s(Y) \leq s(X) + w_{X,Y}$ for all $(Y - X \leq w_{X,Y}) \in \mathcal{C}$. So the *Simple Temporal Problem* (STP) is that of determining whether a given STN is consistent or not.

Any STN $\mathcal{N} = (\mathcal{T}, \mathcal{C})$ can be seen as a directed weighted graph with vertices $\mathcal{T}$ and arc set $A_{\mathcal{C}} \triangleq \{(X, Y, w_{X,Y}) \mid (Y - X \leq w_{X,Y}) \in \mathcal{C}\}$. So, a *path* p in $\mathcal{N}$ is any finite sequence of vertices $p = (v_0, v_1, \ldots, v_k)$ (some $k \geq 1$) such that $(v_i, v_{i+1}) \in A_{\mathcal{C}}$ for every $i \in [0, k) \cap \mathbb{Z}$; the total weight of p is then $w_p \triangleq \sum_{i=0}^{k-1} w_{v_i, v_{i+1}}$. A *cycle* C in $\mathcal{N}$ is any set of arcs $C \subseteq A_{\mathcal{C}}$ cyclically sequenced as $a_0, a_1, \ldots a_{\ell-1}$ where an head equals a tail, i.e., $h(a_i) = t(a_j)$, iff $j = i + 1 \mod \ell$; it is called a *negative cycle* if $w(C) \leq 0$, where $w(C)$ stands for $\sum_{a \in C} w_a$. A graph is called *conservative* when it contains no negative cycle. A *schedule* is any function $f : \mathcal{T} \mapsto \mathbb{R}$. So the *reduced weight* of an arc $a = (t, h, w_a)$ with respect to a schedule f is defined as $w_a^f \triangleq w_a - f(h) + f(t)$. A schedule f is *feasible* iff $w_a^f \geq 0$ for every $a \in A_{\mathcal{C}}$.

It is also worth noticing that, given two feasible schedules s_1, s_2 of any STN, the pointwise-minimum schedule, $s(u) \triangleq \min(s_1(u), s_2(u)) \; \forall u \in \mathcal{T}$, is also feasible. Indeed, among all of the possible feasible schedules of a given consistent STN, it is natural to consider the *least* feasible one; i.e., $\hat{s} : \mathcal{T} \to \mathbb{R}_{\geq 0}$ is the *least feasible schedule* of STN $\mathcal{N}$ if $\hat{s}$ is feasible for $\mathcal{N}$ and, for any other non-negative feasible schedule $s' \geq 0$ of $\mathcal{N}$, it holds $\hat{s}(u) \leq s'(u) \; \forall u \in \mathcal{T}$.

Remarkably, finding the least feasible schedule of an STN takes polynomial time [4].
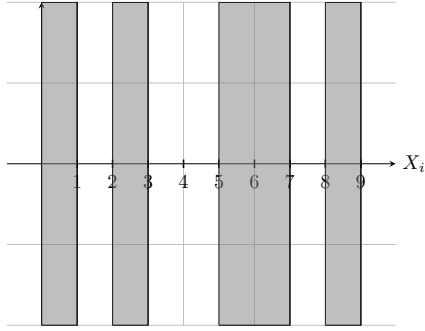
Figure 1 An example of a $\mathtt{t_2}$-constraint: $(0 \leq X_i \leq 1) \vee (2 \leq X_i \leq 3) \vee (5 \leq X_i \leq 7) \vee (8 \leq X_i \leq 9)$.
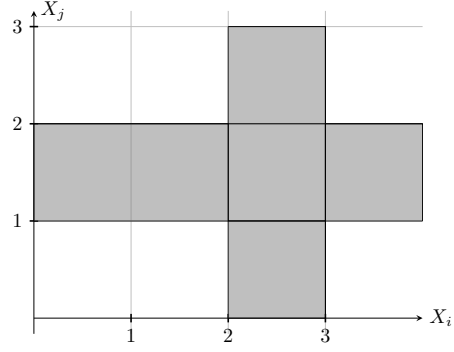
Figure 2 An example of a $\mathtt{t_3}$-constraint: $(2 \leq X_i \leq 3) \vee (1 \leq X_j \leq 2)$.

▶ **Theorem 2** ([4]). *Let $\mathcal{N} = (\mathcal{T}, \mathcal{C})$ be an STN. The* Bellman-Ford *(BF) algorithm (cfr. [2]) produces in $O(|\mathcal{T}| \cdot |\mathcal{C}|)$ time: either the least feasible schedule $\hat{s} : \mathcal{T} \to \mathbb{R}_{\geq 0}$, in case $\mathcal{N}$ is consistent; or a certificate that $\mathcal{N}$ is inconsistent in the form of a negative cycle. Moreover, if the weights of the arcs are all integers, then the scheduling values of $\hat{s}$ are all integers too.*

Concerning the BF algorithm itself, it's worth considering an improved variant of it that we call the *Bellman-Ford Value-Iteration* (BF-VI). The basic idea of BF-VI is the same as the original BF algorithm in that each vertex is used as a candidate to relax its adjacent vertices. The improvement is that instead of trying all vertices blindly, BF-VI maintains a queue Q of candidate vertices and adds a vertex to Q only if that vertex is relaxed. A candidate vertex v is extracted from Q according to a fixed policy (e.g., LIFO), and then the adjacent vertices of v are possibly relaxed as usual and added to Q (if they are not already in there, no repetitions are allowed). This process repeats until no more vertex can be relaxed.

BF-VI serves us as a basic model, to be leveraged to design faster algorithms for RDTPs.

2.1   Restricted Disjunctive Temporal Problems

Let us proceed by formally defining RDTNs and RDTPs. Fig. 1 and Fig. 2 (above) illustrate an example of a $\mathtt{t_2}$-constraint and $\mathtt{t_3}$-constraint (respectively).

▶ **Definition 3** (RDTNs, RDTPs [9, 10]). A *Restricted Disjunctive Temporal Network* (RDTN) $\mathcal{N}$ is a pair $(\mathcal{T}, \mathcal{C})$, where $\mathcal{T}$ is a set of time-points and $\mathcal{C} = \mathcal{C}_{\mathtt{t_1}} \cup \mathcal{C}_{\mathtt{t_2}} \cup \mathcal{C}_{\mathtt{t_3}}$ is a set of *restricted disjunctive temporal constraints* over $\mathcal{T}$, each being either one of the following three types:
($\mathtt{t_1}$) : $(Y - X \leq w_{X,Y})$, where $X, Y \in \mathcal{T}$ and $w_{X,Y} \in \mathbb{R}$;
($\mathtt{t_2}$) : $\bigvee_{i=1}^{k}(l_i \leq X \leq u_i)$, where $X \in \mathcal{T}$ and $l_i, u_i \in \mathbb{R}$ for every $i = 1, \ldots, k$;
($\mathtt{t_3}$) : $(l_1 \leq X \leq u_1) \vee (l_2 \leq Y \leq u_2)$, where $X, Y \in \mathcal{T}$ and $l_i, u_i \in \mathbb{R}$ for $i = 1, 2$.
An RDTN is *consistent* if it admits a *feasible schedule*, i.e., some $s : \mathcal{T} \mapsto \mathbb{R}$ satisfying all of the disjunctive temporal constraints in $\mathcal{C}$. The *Restricted Disjunctive Temporal Problem* (RDTP) is that of determining whether a given RDTN is consistent or not.

Notice that $\mathtt{t_1}$-constraints do coincide with simple temporal constraints of STNs.

We assume w.l.o.g. that the disjuncts of any $\mathtt{t_2}$-constraint are arranged in ascending order of the end points of their corresponding intervals, i.e., $l_i < l_{i+1} \wedge u_i < u_{i+1} \; \forall i$, whenever $\bigvee_{i=1}^{k}(l_i \leq X \leq u_i) \in \mathcal{C}_{\mathtt{t_2}}$; these natural orderings on the interval domains of the time-points will be referred to as as their *nominal ordering*. For any $\tau \in \{1, 2, 3\}$, $|\mathcal{C}_{\mathtt{t_\tau}}|$ denotes the number

(a) Multi-Head Hyperarc   **(b)** Multi-Tail Hyperarc
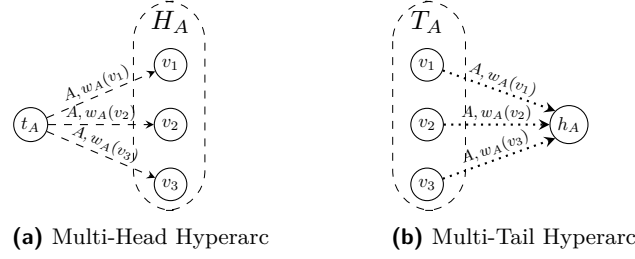
■ **Figure 3** Hyperarcs in Hyper Temporal Networks.

of $\mathtt{t}_\tau$-constraints (i.e., the cardinality of $\mathcal{C}_{\mathtt{t}_\tau}$, not the encoding length). Also, for any $c_X \in \mathcal{C}_{\mathtt{t}_2}$, $|c_X|$ denotes the number of disjuncts of c_X, and $d_{\mathcal{C}_{\mathtt{t}_2}} \triangleq \sum_{c_X \in \mathcal{C}_{\mathtt{t}_2}} |c_X|$. Finally, let us fix a total ordering on the time-points, i.e., $\mathcal{T} = \{T_1, \ldots, T_k\}$, this induces an ordering on the pair of disjuncts in any $\mathtt{t}_3$-constraint; so, provided, $c = (l_1 \leq X_i \leq u_1) \vee (l_2 \leq X_j \leq u_2) \in \mathcal{C}_{\mathtt{t}_3}$, for some $i < j$, then, $d' \triangleq (l_1 \leq X_i \leq u_1)$ and $d'' \triangleq (l_2 \leq X_j \leq u_2)$ are called the *first* and the *second* disjunct of c (respectively).

As mentioned in the introduction, Kumar showed in [10, 9] that RDTPs are solvable in deterministic strongly polynomial time by reducing them to CRCs [6].

2.2 Hyper Temporal Networks

In order to study the tractability frontier of RDTPs, we shall consider the HyTN model which is grounded on directed hypergraphs as defined next.

▶ **Definition 4** ([3]). A *directed hypergraph* $\mathcal{H}$ is a pair $(\mathcal{T}, \mathcal{A})$, where $\mathcal{T}$ is the set of nodes, and $\mathcal{A}$ is the set of *hyperarcs*. Each hyperarc $A \in \mathcal{A}$ is either *multi-head* or *multi-tail*:

- A *multi-head* hyperarc $A = (t_A, H_A, w_A)$ has a distinguished node t_A, called the *tail* of A, and a non-empty set $H_A \subseteq V \setminus \{t_A\}$ containing the *heads* of A; to each head $v \in H_A$, it is associated a *weight* $w_A(v) \in \mathbb{R}$, which is a real number (unless otherwise specified). Fig. 3a depicts a possible representation of a multi-head hyperarc: the tail is connected to each head by a dashed arc labeled by the name of the hyperarc and the weight associated to the considered head.
- A *multi-tail* hyperarc $A = (T_A, h_A, w_A)$ has a distinguished node h_A, called the *head* of A, and a non-empty set $T_A \subseteq V \setminus \{h_A\}$ containing the *tails* of A; to each tail $v \in T_A$, it is associated a *weight* $w_A(v) \in \mathbb{R}$, which is a real number (unless otherwise specified). Fig. 3b depicts a possible representation of a multi-tail hyperarc: the head is connected to each tail by a dotted arc labeled by the name of the hyperarc and weights.

The *cardinality* of a hyperarc $A \in \mathcal{A}$ is $|A| \triangleq |H_A \cup \{t_A\}|$ if A is multi-head, and $|A| \triangleq |T_A \cup \{h_A\}|$ if A is multi-tail; when $|A| = 2$, then $A = (u, v, w)$ is a standard arc. The *order* and *size* of a directed hypergraph $(\mathcal{T}, \mathcal{A})$ are $|\mathcal{T}|$ and $m_\mathcal{A} \triangleq \sum_{A \in \mathcal{A}} |A|$ (respectively).

▶ **Definition 5** (GENERAL-HYTN [3]). A *general-HyTN* is a directed hypergraph $\mathcal{H} = (\mathcal{T}, \mathcal{A})$ where each node $X \in \mathcal{T}$ represents a time-point, and each multi-head/multi-tail hyperarc stands for a set of temporal distance constraints between the tail/head and the heads/tails.

In general-HyTNs, an hyperarc is *satisfied* when at least one of its distance constraints is satisfied. Then, a HyTN is *consistent* when it is possible to assign a value to each time-point so that all of its hyperarcs are satisfied. More formally, in the HyTN model the consistency-checking problem is the following decision problem.

▶ **Definition 6** (GENERAL-HYTP [3]). Given a general-HyTN $\mathcal{H} = (\mathcal{T}, \mathcal{A})$, the *General Hyper Temporal Problem* (GENERAL-HYTP) is that of deciding whether or not there exists a schedule $s : \mathcal{T} \to \mathbb{R}$ such that, for every hyperarc $A \in \mathcal{A}$, the following hold:

- if $A = (t, h, w)$ is a standard arc, then: $s(h) - s(t) \leq w$;
- if $A = (t_A, H_A, w_A)$ is a multi-head hyperarc, then: $s(t_A) \geq \min_{v \in H_A} \{s(v) - w_A(v)\}$;
- if $A = (T_A, h_A, w_A)$ is a multi-tail hyperarc, then: $s(h_A) \leq \max_{v \in T_A} \{s(v) + w_A(v)\}$.

Any such schedule s is called *feasible*. A HyTN that admits at least one feasible schedule is called *consistent*.

Comparing the consistency of HyTNs with the consistency of STNs, the most important aspect of novelty is that, while in a distance graph of STNs each arc represents a distance constraint and all such constraints have to be satisfied by any feasible schedule, in a HyTN each hyperarc represents a disjunction of one or more distance constraints and a feasible schedule has to satisfy at least one of such distance constraints for each hyperarc.

Let us survey some interesting properties about the consistency-checking problem above. The first one is that any integer-weighted HyTN admits an integer-valued feasible schedule when it is consistent, as stated in the following proposition.

▶ **Proposition 7** ([3]). *Let $\mathcal{H} = (\mathcal{T}, \mathcal{A})$ be an integer-weighted[*] and consistent general-HyTN. Then $\mathcal{H}$ admits an integer feasible schedule $s : \mathcal{T} \to \{-T, -T+1, \ldots, T-1, T\}$, where $T = \sum_{A \in \mathcal{A}, v \in \mathcal{T}} |w_A(v)|$.*

The following theorem states that GENERAL-HYTP is NP-complete, in a strong sense.

▶ **Theorem 8** ([3]). GENERAL-HYTP *is an NP-complete problem even if the input instances $\mathcal{H} = (V, \mathcal{A})$ are restricted to satisfy $w_A(\cdot) \in \{-1, 0, 1\}$ and $|H_A|, |T_A| \leq 2$ for every $A \in \mathcal{A}$.*

As observed in [3], Theorem 8 motivates the study of consistency problems on HyTNs having either only multi-head or only multi-tail hyperarcs. In the former case, the consistency-checking problem is called HEAD-HYTP, while in the latter it is TAIL-HYTP; as stated in Theorem 10, the complexity of checking these two problems turns out to be lower than that for DTPs, i.e., both HEAD-HYTP, TAIL-HYTP $\in$ NP$\cap$co-NP, instead of being NP-complete.

So it's worth considering the following specialized notion of consistency for HyTNs.

▶ **Definition 9** (HEAD-HYTP). Given a multi-head HyTN $\mathcal{H} = (\mathcal{T}, \mathcal{A})$, the HEAD-HYTP problem is that of deciding whether or not there exists a schedule $s : \mathcal{T} \to \mathbb{R}$ such that:

$$s(t_A) \geq \min_{v \in H_A} \{s(v) - w_A(v)\}, \quad \forall A \in \mathcal{A}.$$

The tightest currently known worst-case time complexity upper-bound for solving (integer-weighted) HEAD-HYTPs was established in [3] and it is expressed in the following theorem.

▶ **Theorem 10** ([3]). *The following proposition holds on (integer-weighted, multi-head) HyTNs. There exists an $O\big((|\mathcal{T}| + |\mathcal{A}|) \cdot m_{\mathcal{A}} \cdot W\big)$ pseudo-polynomial time algorithm for checking HEAD-HYTP; given any HyTN $\mathcal{H} = (\mathcal{T}, \mathcal{A})$, if $\mathcal{H}$ is consistent the same algorithm also returns an integer-valued feasible schedule $s : \mathcal{T} \to \mathbb{Z}$ of $\mathcal{H}$; otherwise, it returns a negative certificate in the form of a negative hypercycle (cfr. Appendix A for more details on that).*

Above, $W \triangleq \max_{A \in \mathcal{A}, v \in H_A} |w_A(v)|$ is the maximum absolute value among the weights.

---

[*] Integer-weighted $HyTN$ means that $w_A(v) \in \mathbb{Z}$ for every $A \in \mathcal{A}$ and $v \in \mathcal{T}$ for which $w_A(v)$ is defined.

Concluding this section we recall that the two problems HEAD-HYTP and TAIL-HYTP are actually inter-reducible, i.e., one can check any one of the two models in $f(m, n, W)$-time whenever there's an $f(m, n, W)$-time procedure for checking the consistency of the other one.

▶ **Theorem 11** ([3]). HEAD-HYTP *and* TAIL-HYTP *are inter-reducible by means of* log-*space, linear-time, local-replacement reductions.*

Thus, Theorem 10 extends to multi-tail HyTNs (i.e., they're checkable in pseudo-poly time).

3   Faster Deterministic Algorithm for $\mathtt{t_2}$DTPs

This section offers a deterministic quadratic time algorithm for solving temporal problems having only $\{\mathtt{t_1}, \mathtt{t_2}\}$-constraints, as defined below.

The same algorithm will be leveraged to solve RDTPs fastly, later on in Section 4.

▶ **Definition 12.** Any RDTN $\mathcal{N} = (\mathcal{T}, \mathcal{C}_{\mathtt{t_1}} \cup \mathcal{C}_{\mathtt{t_2}} \cup \mathcal{C}_{\mathtt{t_3}})$ having $\mathcal{C}_{\mathtt{t_3}} = \emptyset$ is called $\mathtt{t_2}DTN$.

So, $\mathtt{t_2}$DTNs are denoted simply as $(\mathcal{T}, \mathcal{C}_{\mathtt{t_1}} \cup \mathcal{C}_{\mathtt{t_2}})$. The corresponding temporal problem, i.e., $\mathtt{t_2}DTP$, is that of determining whether a given $\mathtt{t_2}$DTN is consistent or not.

One possible solution to $\mathtt{t_2}$DTPs is Kumar's reduction from RDTPs to CRCs [10]. Our solution, named $\mathtt{t_2DTP()}$, employs kind of a value-iteration approach in which all are initially set to zero and then updated monotonically upwards by necessary arc relaxations – this is somehow reminiscent of the BF-VI algorithm for STPs mentioned in Section 2. Indeed, given a $\mathtt{t_2}$DTN $\mathcal{N}_{\mathtt{t_2}} = (\mathcal{T}, \mathcal{C}_{\mathtt{t_1}} \cup \mathcal{C}_{\mathtt{t_2}})$, we firstly solve the STP $\mathcal{N}_{\mathtt{t_1}} = (\mathcal{T}, \mathcal{C}_{\mathtt{t_1}})$ (e.g., with BF-VI). If $\mathcal{N}_{\mathtt{t_1}}$ is consistent, the returned least feasible schedule $\hat{\varphi}_{\mathcal{N}}$ provides an initial candidate, the next step in mind being that of satisfying all the $\mathtt{t_2}$-constraints. For this, recall that the disjuncts of any $c_{\mathtt{t_2}} \in \mathcal{C}_{\mathtt{t_2}}$ are arranged according to their nominal ordering, so that we can try to satisfy any given $c_{\mathtt{t_2}}$ by iteratively picking the next (i.e., in ascending order) unsatisfied disjunct of $c_{\mathtt{t_2}}$ and by enforcing its lower-bound constraint in an auxiliary STN as if it were a $\mathtt{t_1}$-constraint. While there's an unsatisfied $\mathtt{t_2}$-constraint $c_{\mathtt{t_2}}$, the current candidate schedule is thus increased by the least necessary amount satisfying both $c_{\mathtt{t_2}}$ and the whole $\mathcal{C}_{\mathtt{t_1}}$. It turns out that this can be done efficiently by performing $|\mathcal{C}_{\mathtt{t_2}}|$ calls to the Dijkstra shortest paths algorithm [7]. In order to show this, let us point out two key facts (i.e., Lemma 13 and 14).

▶ **Lemma 13.** *Let $\mathcal{N} = (\mathcal{T}, \mathcal{C}_{\mathtt{t_1}})$ be any STN, and let φ, φ' be any pair of schedules of $\mathcal{N}$.*

Let $\mathcal{N}^{\varphi}$ be the STN reweighted according to the reduced-costs weight transformation w^{φ} (i.e., each weight $w_{X,Y}$ in $\mathcal{N}$ is simply replaced by $w^{\varphi}_{X,Y}$), let $\mathcal{N}^{\varphi'}$ be the same w.r.t. $w^{\varphi'}$. Let $\delta^{\varphi}_X : \mathcal{T} \to \mathbb{Z} \cup \{+\infty\}$ be the length of the shortest path in $\mathcal{N}^{\varphi}$ from any $T \in \mathcal{T}$ to $X \in \mathcal{T}$, and let $\delta^{\varphi'}_X$ be the same w.r.t. $\mathcal{N}^{\varphi'}$.

Then for every $T, X \in \mathcal{T}$, either $\delta^{\varphi'}_X(T)$ and $\delta^{\varphi}_X(T)$ are both $+\infty$, or the following holds:

$$\delta^{\varphi'}_X(T) - \delta^{\varphi}_X(T) = \big(\varphi'(T) - \varphi(T)\big) - \big(\varphi'(X) - \varphi(X)\big).$$

Proof. Let $T, X \in \mathcal{T}$ (arbitrarily), w.l.o.g. X is reachable from T in $\mathcal{N}$ (otherwise, $\delta^{\varphi'}_X(T)$ and $\delta^{\varphi}_X(T)$ are both $+\infty$). Consider any path $p_{T,X}$ from T to X in $\mathcal{N}^{\varphi}$, i.e., for some $k \geq 0$:

$$p_{T,X} \triangleq (T = T_0, T_1, T_2, \dots, T_k = X), \text{ having total weight } w^{\varphi}_{p_{T,X}} \triangleq \sum_{i=0}^{k-1} w^{\varphi}_{T_i, T_{i+1}} \text{ in } \mathcal{N}^{\varphi}.$$

Then, the following holds by telescoping:

$$w_{p_{T,X}}^{\varphi} = \big(w_{T_0,T_1} - \varphi(T_1) + \varphi(T_0)\big) + \big(w_{T_1,T_2} - \varphi(T_2) + \varphi(T_1)\big) + \dots$$
$$\dots + \big(w_{T_{k-1},T_k} - \varphi(T_k) + \varphi(T_{k-1})\big)$$
$$= \varphi(T_0) - \varphi(T_k) + \sum_{i=0}^{k-1} w_{T_i,T_{i+1}} = \varphi(T) - \varphi(X) + w_{p_{T,X}}. \tag{1}$$

Thus, provided $\delta_X(T)$ is the shortest path distance from T to X in the original network $\mathcal{N}$, we have:

$$\delta_X^{\varphi}(T) = \min \big\{ w_{p_{T,X}}^{\varphi} \mid p_{T,X} \text{ is a path from } T \text{ to } X \text{ in } \mathcal{N} \big\} \qquad \text{(by def. of } \delta_X^{\varphi})$$
$$= \min \big\{ \varphi(T) - \varphi(X) + w_{p_{T,X}} \mid p_{T,X} \text{ is any path from } T \text{ to } X \text{ in } \mathcal{N} \big\} \qquad \text{(by (1))}$$
$$= \varphi(T) - \varphi(X) + \min \big\{ w_{p_{T,X}} \mid p_{T,X} \text{ is any path from } T \text{ to } X \text{ in } \mathcal{N} \big\} \quad (\varphi \text{ is constant here})$$
$$= \varphi(T) - \varphi(X) + \delta_X(T). \qquad \text{(by def. of } \delta_X)$$

For the same reason, $\delta_X^{\varphi'}(T) = \varphi'(T) - \varphi'(X) + \delta_X(T)$. Therefore,

$$\delta_X^{\varphi'}(T) - \delta_X^{\varphi}(T) = \big(\varphi'(T) - \varphi'(X) + \delta_X(T)\big) - \big(\varphi(T) - \varphi(X) + \delta_X(T)\big)$$
$$= \big(\varphi'(T) - \varphi(T)\big) - \big(\varphi'(X) - \varphi(X)\big).$$

This concludes the proof. ◀

▶ **Lemma 14.** *Let $\mathcal{N} = (\mathcal{T}, \mathcal{C}_{t_1})$ be any STN, and let $\hat{\varphi}$ be the least feasible schedule of $\mathcal{N}$. Fix some $X \in \mathcal{T}$ and some real value $l_X \geq \hat{\varphi}(X)$. Let $\mathcal{N}' = (\mathcal{T}', \mathcal{C}'_{t_1})$ be the auxiliary STN obtained by introducing a corresponding lower-bound t_1-constraint over X, i.e.,*

$$\mathcal{T}' \triangleq \mathcal{T} \cup \{z\}, \qquad \mathcal{C}'_{t_1} \triangleq \mathcal{C}_{t_1} \cup \big\{(z - T \leq 0) \mid T \in \mathcal{T}\big\} \cup \big\{(z - X \leq -l_X)\big\}.$$

Let $\mathcal{N}^{\hat{\varphi}}$ be the STN reweighted according to the reduced-costs weight transformation $w^{\hat{\varphi}}$, and let $\delta_X^{\hat{\varphi}}(T)$ be the length of the shortest path in $\mathcal{N}^{\hat{\varphi}}$ from (any) $T \in \mathcal{T}$ to X.
Then, for every $T \in \mathcal{T}$, the least feasible schedule $\hat{\varphi}'$ of $\mathcal{N}'$ is given by:

$$\hat{\varphi}'(T) = \hat{\varphi}(T) + \max \big(0, l_X - \hat{\varphi}(X) - \delta_X^{\hat{\varphi}}(T)\big).$$

Proof. Let w.l.o.g. $\hat{\varphi}'(z) = 0$. In order to become feasible for $\mathcal{N}'$ we claim, for every $T \in \mathcal{T}$, that the least feasible schedule $\hat{\varphi}(T)$ must be increased by at least $\max\big(0, l_X - \hat{\varphi}(X) - \delta_X^{\hat{\varphi}}(T)\big)$ time units (because of the lower-bound constraint $(z - X \leq -l_X) \in \mathcal{C}'_{t_1}$). Indeed, for any $T \in \mathcal{T}$ that reaches X in $\mathcal{N}$, the t_1-constraint $(X - T \leq \delta_X(T))$ (which is induced by telescoping all of the t_1-constraints along any shortest path from T to X) must be satisfied. On the other hand, by Lemma 13 (applied to $\hat{\varphi}$ and to the anywhere-zero[†] schedule), it holds $\hat{\varphi}(X) - \hat{\varphi}(T) = \delta_X(T) - \delta_X^{\hat{\varphi}}(T)$. This can be seen as follows: if $\hat{\varphi}(T)$ is kept fixed, then $\hat{\varphi}(X)$ can be increased by at most $\delta_X^{\hat{\varphi}}(T)$ time units without breaking the induced constraint $(X - T \leq \delta_X(T))$. Here, $\hat{\varphi}(X)$ must be increased by at least $l_X - \hat{\varphi}(X)$ time units in order to satisfy $(z - X \leq -l_X) \in \mathcal{C}'_{t_1}$, so $\hat{\varphi}(T)$ must be increased by at least the amount said above.
Next, we claim this increase also preserves feasability, i.e., it is the *least feasible increase*. For ease of notation, let $f(z) \triangleq 0$ and $f(T) \triangleq \hat{\varphi}(T) + \max\big(0, l_X - \hat{\varphi}(X) - \delta_X^{\hat{\varphi}}(T)\big) \; \forall T \in \mathcal{T}$.

[†] The anywhere-zero schedule ζ is that defined as, $\zeta(T) = 0$ for every $T \in \mathcal{T}$.

In order to prove that f satifies all the constraints in $\mathcal{C}_{\mathsf{t}_1}$, pick any $(B - A \leq w_{A,B}) \in \mathcal{C}_{\mathsf{t}_1}$. By hypothesis, it holds:

$$\hat{\varphi}(B) - \hat{\varphi}(A) \leq w_{A,B}. \tag{1}$$

For the sake of the argument, let us define: $\Delta_{A,B} \triangleq \big(f(B) - \hat{\varphi}(B)\big) - \big(f(A) - \hat{\varphi}(A)\big)$.

So, the following holds:

$$f(B) - f(A) = \hat{\varphi}(B) - \hat{\varphi}(A) + \Delta_{A,B}. \tag{2}$$

Then, either one of the following two cases holds:

- If $l_X - \hat{\varphi}(X) \leq \delta_X^{\hat{\varphi}}(B)$, then $f(B) = \hat{\varphi}(B)$, so $\Delta_{A,B} \leq 0$. Therefore,

$$
\begin{aligned}
f(B) - f(A) &\leq \hat{\varphi}(B) - \hat{\varphi}(A) && \text{(by (2))} \\
&\leq w_{A,B}. && \text{(by (1))}
\end{aligned}
$$

- If $l_X - \hat{\varphi}(X) > \delta_X^{\hat{\varphi}}(B)$, it is easy to check that $\Delta_{A,B} \leq \delta_X^{\hat{\varphi}}(A) - \delta_X^{\hat{\varphi}}(B)$.
  By definition of $\delta_X^{\hat{\varphi}}$ and since $(B - A \leq w_{A,B}) \in \mathcal{C}_{\mathsf{t}_1}$, then $\delta_X^{\hat{\varphi}}(A) \leq \delta_X^{\hat{\varphi}}(B) + w_{A,B}^{\hat{\varphi}}$. Therefore,

$$
\begin{aligned}
f(B) - f(A) &\leq \hat{\varphi}(B) - \hat{\varphi}(A) + \delta_X^{\hat{\varphi}}(A) - \delta_X^{\hat{\varphi}}(B) \\
&\leq \hat{\varphi}(B) - \hat{\varphi}(A) + w_{A,B}^{\hat{\varphi}} = w_{A,B}.
\end{aligned}
$$

So, in either case, $f(B) - f(A) \leq w_{A,B}$.

Finally, clearly $f(X) = l_X$, so $(z - X \leq -l_X) \in \mathcal{C}'_{\mathsf{t}_1}$ is also satisfied.

This proves f is a feasible schedule of $\mathcal{N}'$. All in, it is the least feasible, i.e., $f = \hat{\varphi}'$. ◀

With this two facts in mind, the description of $\mathsf{t}_2\mathtt{DTP}()$ can now proceed more smoothly.

Recall that, firstly, the STN $\mathcal{N}_{\mathsf{t}_1}$ is checked. If $\mathcal{N}_{\mathsf{t}_1}$ is already inconsistent, so it is $\mathcal{N}_{\mathsf{t}_2}$; otherwise, $\hat{\varphi}_{\mathcal{N}}$ is the least feasible schedule of $\mathcal{N}_{\mathsf{t}_1}$. So, $w^{\hat{\varphi}_{\mathcal{N}}} \geq 0$ for every constraint in $\mathcal{C}_{\mathsf{t}_1}$. Now, for each target node $X \in \mathcal{T}$, the Dijkstra algorithm on input $(\mathcal{N}^{\hat{\varphi}_{\mathcal{N}}}, X)$ computes $\delta_X^{\hat{\varphi}_{\mathcal{N}}}(T)$. The whole distance matrix $\{\delta_X^{\hat{\varphi}_{\mathcal{N}}}(T)\}_{T \in \mathcal{T}, X \in \mathcal{T}}$ is computed here, and kept stored in memory. Multiple-sources single-target shortest paths are needed, actually, but these can be easily computed with the traditional Dijkstra's algorithm, e.g., just reverse the direction of all arcs in the input network and treat the single-target node as if it were a single-source. What follows aims, if there's still an unsatisfied t_2-constraint $c_{\mathsf{t}_2} \in \mathcal{C}_{\mathsf{t}_2}$, at increasing the candidate schedule f by the least necessary amount satisfying both c_{t_2} and the whole $\mathcal{C}_{\mathsf{t}_1}$. So, let us initialize $f \leftarrow \hat{\varphi}_{\mathcal{N}}$. Then the following iterates.

While $\exists$ some $X \in \mathcal{T}$ and $c_X = \bigvee_{i=1}^{k}(l_i \leq X \leq u_i) \in \mathcal{C}_{\mathsf{t}_2}$ s.t. $f(X)$ doesn't satisfy c_X: if $f(X) > u_k (= \max_i u_i)$, then $\mathcal{N}_{\mathsf{t}_2}$ is inconsistent (see Theorem 15); otherwise, let i^* be the smallest $i \in [1, k]$ such that $f(X) < l_i$. By Lemma 13 and given f, then δ_X^f is given by:

$$\delta_X^f(T) \leftarrow \delta_X^{\hat{\varphi}_0}(T) + \big(f(T) - \hat{\varphi}_0(T)\big) - \big(f(X) - \hat{\varphi}_0(X)\big), \ \forall \, T \in \mathcal{T}. \tag{rule-δ}$$

So, by Lemma 14, the following updating rule:

$$f(T) \leftarrow f(T) + \max\big(0, l_{i^*} - f(X) - \delta_X^f(T)\big), \ \forall \, T \in \mathcal{T}. \tag{rule-f}$$

yelds the least feasible schedule for the next auxiliary STN $\mathcal{N}'_{\mathsf{t}_1}$ obtained by adding the new lower-bound t_1-constraint $(z - X \leq -l_{i^*})$. At each iteration of the while-loop $\mathcal{N}'_{\mathsf{t}_1}$ is enriched with an additional lower-bound t_1-constraint as above. So, $\mathcal{N}'_{\mathsf{t}_1}$ has $|\mathcal{T}'| = |\mathcal{T}| + 1$ time-points (z included) and at most $|\mathcal{C}'_{\mathsf{t}_1}| \leq |\mathcal{C}_{\mathsf{t}_1}| + |\mathcal{C}_{\mathsf{t}_2}|$ t_1-constraints (one t_1-constraint per $c_{\mathsf{t}_2} \in \mathcal{C}_{\mathsf{t}_2}$

is enough, as for each $c_{\mathtt{t}_2}$ only its greatest lower-bound counts). If the while-loop completes without ever finding $\mathcal{N}_{\mathtt{t}_2}$ to be inconsistent (because, eventually, $f(X) > u_k (= \max_i u_i)$ for some $X \in \mathcal{T}$ at some point), then the last updating of f yelds the least feasible schedule of $\mathcal{N}_{\mathtt{t}_2}$ (as shown below in Theorem 15). This concludes the description of $\mathtt{t}_2\mathtt{DTP()}$.

Notice that, during the whole computation, the scheduling values can only increase monotonically upwards – like in a value-iteration process.

▶ **Theorem 15.** $\mathit{t}_2 DTP()$ is correct, i.e., on any input $\mathit{t}_2 DTN$ $\mathcal{N}_{t_2} = (\mathcal{T}, \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2})$, it returns a feasible schedule $\varphi : \mathcal{T} \to \mathbb{R}$, if $\mathcal{N}_{t_2}$ is consistent; otherwise, it recognizes $\mathcal{N}_{t_2}$ as inconsistent.

Proof. Let $\iota = 0, 1, 2, \ldots, \iota_h$ be all the iterations of the while-loop of $\mathtt{t}_2\mathtt{DTP()}$, where ι_h is assumed to be the last iteration where the updating $\mathtt{rule\text{-}}f$ is applied.

For every iteration $\iota \in [1, \iota_h]$, the auxiliary STN $\mathcal{N}'^{(\iota)}_{\mathtt{t}_1}$ is formally defined as:

$$\mathcal{N}'^{(\iota)}_{\mathtt{t}_1} \triangleq (\mathcal{T} \cup \{z\}, \mathcal{C}'^{(\iota)}_{\mathtt{t}_1}), \text{ where } z \text{ is the } \textit{zero time-point}, \text{ and } \ldots$$

$$\mathcal{C}'^{(\iota)}_{\mathtt{t}_1} \triangleq \mathcal{C}_{\mathtt{t}_1} \cup \left\{(z - T \le 0) \mid T \in \mathcal{T}\right\} \cup \left\{(z - X^{(\gamma)} \le -l^{(\gamma)}_{i*}) \mid 1 \le \gamma \le \iota\right\},$$

where, for all $\gamma \le \iota$, $X^{(\gamma)}$ is the (unique) $X \in \mathcal{T}$ appearing in some $\mathtt{t}_2$-constraint that is considered at the while-loop's γ-th iteration, and $l^{(\gamma)}_{i*}$ is its corresponding lower-bound.

Also, let $f^{(\iota)}$ be the candidate schedule as updated by $\mathtt{rule\text{-}}f$ during the ι-th iteration.

By applying Lemma 13 and 14 repeatedly, for each iteration ι, it holds that $f^{(\iota)}$ is the least feasible schedule of $\mathcal{N}'^{(\iota)}_{\mathtt{t}_1}$. This is the key invariant at the heart of $\mathtt{t}_2\mathtt{DTP()}$.

Concerning actual correctness, firstly, assume that $\mathtt{t}_2\mathtt{DTP()}$ recognizes $\mathcal{N}_{\mathtt{t}_2}$ as inconsistent. If $\mathcal{N}_{\mathtt{t}_1}$ was already inconsistent (cfr. Theorem 2), so $\mathcal{N}_{\mathtt{t}_2}$ is too. Otherwise, the inconsistency of $\mathcal{N}_2$ really holds because of these two facts jointly: (i) the key invariant mentioned above; and, (ii) at the end of the while-loop, it must be $f(X) > u_k (= \max_i u_i)$ for some $\mathtt{t}_2$-constraint $c_X = \bigvee_{i=1}^{k}(l_i \le X \le u_i) \in \mathcal{C}_{\mathtt{t}_2}$. Indeed notice that, by (i), no possible feasible schedule $g < f$ can be neglected (discarded) during the upward monotone (value-iteration like) updates of the schedules; and, by (ii), no possible schedule $g \ge f$ can ever satisfy $c_X \in C_2$. So, $\mathcal{N}_{\mathtt{t}_2}$ is really inconsistent.

Secondly, assume that $\mathcal{N}_{\mathtt{t}_2}$ is recognized as consistent, by returning a schedule $f^{(\iota_h)}$.

Since $\mathtt{t}_2\mathtt{DTP()}$ can do that only after the above while-loop completes, the exit condition of the latter ensures that $f^{(\iota_h)}$ satisfies every constraint in $\mathcal{C}_{\mathtt{t}_2}$. Moreover, the key invariant implies that $f^{(\iota_h)}$ is the least feasible schedule of $\mathcal{N}'^{(\iota_h)}_{\mathtt{t}_1}$, so that $f^{(\iota_h)}$ satisfies all of the $\mathtt{t}_1$-constraints in $\mathcal{C}_{\mathtt{t}_1}$. These two combined, $f^{(\iota_h)}$ is the least feasible schedule of $\mathcal{N}_{\mathtt{t}_2}$. So, $\mathcal{N}_{\mathtt{t}_2}$ is indeed consistent.   ◀

The next result asserts that $\mathtt{t}_2\mathtt{DTP()}$ always halts in time polynomial in the input size.

▶ **Theorem 16.** Suppose that $\mathit{t}_2 DTP()$ runs on input $\mathit{t}_2 DTN$ $\mathcal{N}_{t_2} = (\mathcal{T}, \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2})$.
Then, $\mathit{t}_2 DTP()$ halts in time $O\big(|\mathcal{T}| \cdot |\mathcal{C}_{t_1}| + |\mathcal{C}_{t_2}| \cdot (|\mathcal{C}_{t_1}| + |\mathcal{T}| \cdot \log |\mathcal{T}|) + |\mathcal{T}| \cdot d_{\mathcal{C}_{t_2}}\big)$.

Proof. Solving the STP $\mathcal{N}_{\mathtt{t}_1} = (\mathcal{T}, \mathcal{C}_{\mathtt{t}_1})$ with BF-VI takes $O(|\mathcal{T}| \cdot |\mathcal{C}_{\mathtt{t}_1}|)$ time (cfr. Theorem 2). Computing the shortest paths distance matrix $\{\delta^{\hat{\varphi}_{\mathcal{N}}}_X(T)\}_{T \in \mathcal{T}, X \in \mathcal{T}}$ takes $|\mathcal{C}_{\mathtt{t}_2}|$ calls to the Dijkstra algorithm (one per $X \in \mathcal{T}$ participating in some $\mathtt{t}_2$-constraint), so, $O(|\mathcal{C}_{\mathtt{t}_2}| \cdot (|\mathcal{C}_{\mathtt{t}_1}| + |\mathcal{T}| \cdot \log |\mathcal{T}|))$ total time. Checking the while-loop exit condition (i.e., wether there exists some unsatisfied $c_X \in \mathcal{C}_{\mathtt{t}_2}$), can be done in $O(|\mathcal{T}| \cdot d_{\mathcal{C}_{\mathtt{t}_2}})$ total time (because there are at most $d_{\mathcal{C}_{\mathtt{t}_2}}$ iterations and each check can be done in $O(|\mathcal{T}|)$ time). At each iteration of the while-loop, applying $\mathtt{rule\text{-}}\delta$ and $\mathtt{rule\text{-}}f$ to all $T \in \mathcal{T}$ takes $O(|\mathcal{T}|)$ time per iteration, and

we have at most $d_{\mathcal{C}_{t_2}}$ of them; so, notice that it takes only $O(1)$ time per single application of the rules.

Therefore, the overall time complexity of $\texttt{t}_2\texttt{DTP}()$ on any input $\mathcal{N}_{t_2} = (\mathcal{T}, \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2})$ is:

$$\texttt{Time}_{\texttt{t}_2\texttt{DTP}()}(\mathcal{N}_{t_2}) = O\big(|\mathcal{T}| \cdot |\mathcal{C}_{t_1}| + |\mathcal{C}_{t_2}| \cdot (|\mathcal{C}_{t_1}| + |\mathcal{T}| \cdot \log |\mathcal{T}|) + |\mathcal{T}| \cdot d_{\mathcal{C}_{t_2}}\big).$$

This is a strongly polynomial time, i.e., not depending on the magnitude of the arc weights. ◀

4 Faster Deterministic Algorithm for RDTPs

With our brand new $\texttt{t}_2\texttt{DTPs}$ algorithm in mind, let us now focus on solving RDTPs fastly. Given an input RDTP $\mathcal{N} = (\mathcal{T}, \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2} \cup \mathcal{C}_{t_3})$, we firstly solve the $\texttt{t}_2\texttt{DTP}$ $\mathcal{N}_{t_2} = (\mathcal{T}, \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2})$ with $\texttt{t}_2\texttt{DTP}()$ (cfr. Section 3). If $\mathcal{N}_{t_2}$ is already inconsistent, we're done as $\mathcal{N}$ is too. Otherwise, the key idea is that of checking the consistency of all the $\texttt{t}_3$-constraints by making one single reduction call to the 2-SAT problem (which can be solved in linear-time [1]).

For this reason, the universe of boolean variables is $\{x_c\}_{c \in \mathcal{C}_{t_3}}$, i.e., we have one variable per $c \in \mathcal{C}_{t_3}$. Let d', d'' be the first and second disjunct of any given $c \in \mathcal{C}_{t_3}$ (respectively), the intended interpretation being that x_c is $\texttt{true}$ iff d' is satisfied (and d'' can be anything), whereas x_c is $\texttt{false}$ iff d' is unsatisfied and d'' is satisfied.

The 2-CNF formula $\mathrm{CL}_{\mathcal{N}}$ is built as follows. Basically, for each $c \in \mathcal{C}_{t_3}$ and each disjunct d of c, we enforce the binding requirement of satisfying all the temporal constraints in $\{d\} \cup \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2}$, and we check whether this implies that some other disjunct $\tilde{d}$ of any other $\texttt{t}_3$-constraint $\tilde{c} \neq c$ becomes unsatisfiable as a consequence. More precisely, we check whether satisfying $\{d\} \cup \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2}$ implies that some weight $\tilde{u}$ must become a *strict* lower-bound for the scheduling value of some $\tilde{X} \in \mathcal{T}$ that appears in some other $\texttt{t}_3$-disjunct $\tilde{d} = (\tilde{l} \leq \tilde{X} \leq \tilde{u})$. This is formalized in Definition 17 (below). If that is the case, a binary clause asserting the above implication[‡] is added to $\mathrm{CL}_{\mathcal{N}}$. Let us formally describe the details of this construction.

▶ **Definition 17.** Given any RDTP $\mathcal{N} = (\mathcal{T}, \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2} \cup \mathcal{C}_{t_3})$, initially $\mathrm{CL}_{\mathcal{N}}$ is an empty set of binary clauses. For each $\texttt{t}_3$-constraint of $\mathcal{N}$, e.g., for each $c = d'_c \vee d''_c \in \mathcal{C}_{t_3}$ where $d'_c = (l_1 \leq X_i \leq u_1)$ and $d''_c = (l_2 \leq X_j \leq u_2)$, some $i < j$, $\mathrm{CL}_{\mathcal{N}}$ is populated as follows:

1. Consider the $\texttt{t}_2\texttt{DTP}$ $\mathcal{N}[d'_c]_{t_2}$ in which d'_c is added to $\mathcal{C}_{t_1}$ as a pair of $\texttt{t}_1$-constraints, i.e.,

$$\mathcal{N}[d'_c]_{t_2} \triangleq \Big(\mathcal{T} \cup \{z\}, \big(\mathcal{C}_{t_1} \cup \{(z - X_i \leq -l_1), (X_i - z \leq u_1)\}$$
$$\cup \{z - T \leq 0 \mid T \in \mathcal{T}\}\big) \cup \mathcal{C}_{t_2}\Big).$$

   If $\mathcal{N}[d'_c]_{t_2}$ is consistent, let $\hat{\varphi}[d'_c]$ be its least feasible schedule; otherwise, add the unary clause $\neg x_c$ to $\mathrm{CL}_{\mathcal{N}}$. For each $\tilde{c} \neq c$ in $\mathcal{C}_{t_3}$, e.g., $\tilde{c} = (\tilde{l}_1 \leq X_{\tilde{i}} \leq \tilde{u}_1) \vee (\tilde{l}_2 \leq X_{\tilde{j}} \leq \tilde{u}_2) \in \mathcal{C}_{t_3}$,
   - if $\hat{\varphi}[d'_c](X_{\tilde{i}}) > \tilde{u}_1$ then add the implication $x_c \Rightarrow \neg x_{\tilde{c}}$ (i.e., clause $\neg x_c \vee \neg x_{\tilde{c}}$) to $\mathrm{CL}_{\mathcal{N}}$;
   - if $\hat{\varphi}[d'_c](X_{\tilde{j}}) > \tilde{u}_2$ then add the implication $x_c \Rightarrow x_{\tilde{c}}$ (i.e., clause $\neg x_c \vee x_{\tilde{c}}$) to $\mathrm{CL}_{\mathcal{N}}$.
2. Consider the $\texttt{t}_2\texttt{DTP}$ $\mathcal{N}[d''_c]_{t_2}$ in which d''_c is added to $\mathcal{C}_{t_1}$ (similarly as above). If $\mathcal{N}[d''_c]_{t_2}$ is consistent, let $\hat{\varphi}[d''_c]$ be its least feasible schedule; otherwise, add the unary clause x_c to $\mathrm{CL}_{\mathcal{N}}$. Again, for each $\texttt{t}_3$-constraint $\tilde{c} \neq c$ of $\mathcal{N}$, e.g., $\tilde{c} = (\tilde{l}_1 \leq X_{\tilde{i}} \leq \tilde{u}_1) \vee (\tilde{l}_2 \leq X_{\tilde{j}} \leq \tilde{u}_2)$: if $\hat{\varphi}[d''_c](X_{\tilde{i}}) > \tilde{u}_1$ then add the implication $\neg x_c \Rightarrow \neg x_{\tilde{c}}$ (i.e., clause $x_c \vee \neg x_{\tilde{c}}$) to $\mathrm{CL}_{\mathcal{N}}$; and if $\hat{\varphi}[d''_c](X_{\tilde{j}}) > \tilde{u}_2$ then add the implication $\neg x_c \Rightarrow x_{\tilde{c}}$ (i.e., the clause $x_c \vee x_{\tilde{c}}$) instead.

[‡] Here, recall the rule of material implication $p \rightarrow q \leftrightarrow \neg p \vee q$.

So, if the 2-SAT problem instance $\mathrm{CL}_\mathcal{N}$ is unsatisfiable, the input RDTP $\mathcal{N}$ is inconsistent. Otherwise, for every $c = d' \vee d'' \in \mathcal{C}_{\mathtt{t}_3}$ we get at least one feasible $\mathtt{t}_2$DTP: either $\mathcal{N}[d'_c]_{\mathtt{t}_2}$, which is related to the first disjunct $\{d'\} \cup \mathcal{C}_{\mathtt{t}_1} \cup \mathcal{C}_{\mathtt{t}_2}$; or $\mathcal{N}[d''_c]_{\mathtt{t}_2}$, which is related to the second $\{d''\} \cup \mathcal{C}_{\mathtt{t}_1} \cup \mathcal{C}_{\mathtt{t}_2}$ (according to whether x_c is $\mathtt{true}$ or not in the satisfying assignment of $\mathrm{CL}_\mathcal{N}$). Then we compute the pointwise-maximum schedule taken among all of those. Formally,

▶ **Definition 18.** Let $\phi : \{x_c\}_{c \in \mathcal{C}_{\mathtt{t}_3}} \to \{\mathtt{true}, \mathtt{false}\}$ be any satisfying assignment of $\mathrm{CL}_\mathcal{N}$. For every $c = d'_c \vee d''_c \in \mathcal{C}_{\mathtt{t}_3}$, let us define:

$$d_c^\phi \triangleq \begin{cases} d'_c, & \text{if } \phi(x_c) = \mathtt{true}; \\ d''_c, & \text{otherwise.} \end{cases} \qquad \text{then,} \qquad \check{\varphi}_\mathcal{N}(T) \triangleq \max_{c \in \mathcal{C}_{\mathtt{t}_3}} \hat{\varphi}[d_c^\phi](T), \;\; \forall T \in \mathcal{T},$$

where $\hat{\varphi}[d_c^\phi]$ denotes the least feasible schedule of the consistent $\mathtt{t}_2$DTP $\mathcal{N}[d_c^\phi]_{\mathtt{t}_2}$.

The above pointwise-maximum schedule $\check{\varphi}_\mathcal{N}$ turns out to be feasible for the input RDTP $\mathcal{N}$, as we show next. It is assumed we are given an RDTP $\mathcal{N}$ for which $\mathrm{CL}_\mathcal{N}$ is satisfiable.

▶ **Proposition 19.** *Given $\mathcal{N}$ as above, the schedule $\check{\varphi}_\mathcal{N}$ satisfies every $c \in \mathcal{C}_{\mathtt{t}_1}$.*

Proof. Let $c_{\mathtt{t}_1} = (Y - X \leq w_{X,Y}) \in \mathcal{C}_{\mathtt{t}_1}$ be any $\mathtt{t}_1$-constraint, some $X, Y \in \mathcal{T}$ and $w \in \mathbb{R}$. Pick any $c_Y^* \in \arg\max_{c \in \mathcal{C}_{\mathtt{t}_3}} \hat{\varphi}[d_c^\phi](Y)$. Clearly, $\max_{c \in \mathcal{C}_{\mathtt{t}_3}} \hat{\varphi}[d_c^\phi](X) \geq \hat{\varphi}[d_{c_Y^*}^\phi](X)$. Therefore:

$$\check{\varphi}_\mathcal{N}(Y) - \check{\varphi}_\mathcal{N}(X) = \max_{c \in \mathcal{C}_{\mathtt{t}_3}} \hat{\varphi}[d_c^\phi](Y) - \max_{c \in \mathcal{C}_{\mathtt{t}_3}} \hat{\varphi}[d_c^\phi](X)$$

$$\leq \hat{\varphi}[d_{c_Y^*}^\phi](Y) - \hat{\varphi}[d_{c_Y^*}^\phi](X) \leq w,$$

where the very last inequality holds because $\hat{\varphi}[d_{c_Y^*}^\phi]$ is feasible for $(\mathcal{T}, \mathcal{C}_{\mathtt{t}_1})$. So, $\check{\varphi}_\mathcal{N}$ satisfies $c_{\mathtt{t}_1}$. ◀

▶ **Proposition 20.** *Given $\mathcal{N}$ as above, the schedule $\check{\varphi}_\mathcal{N}$ satisfies every $c \in \mathcal{C}_{\mathtt{t}_2}$.*

Proof. Let $c_{\mathtt{t}_2} = \bigvee_{i=1}^k (l_i \leq X \leq u_i) \in \mathcal{C}_{\mathtt{t}_2}$ be any $\mathtt{t}_2$-constraint, some $X \in \mathcal{T}$, $l_i, u_i \in \mathbb{R}$. Pick any $c_X^* \in \arg\max_{c \in \mathcal{C}_{\mathtt{t}_3}} \hat{\varphi}[d_c^\phi](X)$. By definition $\hat{\varphi}[d_{c_X^*}^\phi]$ is a feasible schedule of $\mathcal{N}[d_{c_X^*}^\phi]_{\mathtt{t}_2}$, thus it is feasible for $(\mathcal{T}, \mathcal{C}_{\mathtt{t}_2})$ too. Therefore,

$$l_q \leq \hat{\varphi}[d_{c_X^*}^\phi](X) \leq u_q, \text{ for some } q \in \{1, \dots, k\}.$$

Since, $\check{\varphi}_\mathcal{N}(X) = \varphi[d_{c_X^*}^\phi](X)$, then $\check{\varphi}_\mathcal{N}(X) \in [l_q, u_q]$ for the same q. So, $\check{\varphi}_\mathcal{N}$ satisfies $c_{\mathtt{t}_2}$. ◀

▶ **Proposition 21.** *Given $\mathcal{N}$ as above, the schedule $\check{\varphi}_\mathcal{N}$ satisfies every $c \in \mathcal{C}_{\mathtt{t}_3}$.*

Proof. Let $c_{\mathtt{t}_3} = (l_1 \leq X \leq u_1) \vee (l_2 \leq Y \leq u_2) \in \mathcal{C}_{\mathtt{t}_3}$ be any $\mathtt{t}_3$-constraint, some $X, Y \in \mathcal{T}$, $X < Y$ and $l_1, l_2, u_1, u_2 \in \mathbb{R}$. Assume w.l.o.g. $\phi(x_{c_{\mathtt{t}_3}}) = \mathtt{true}$. Then, $l_1 \leq \hat{\varphi}[d_{c_{\mathtt{t}_3}}^\phi](X) \leq u_1$.

If $c_{\mathtt{t}_3} \in \arg\max_{c \in \mathcal{C}_{\mathtt{t}_3}} \hat{\varphi}[d_c^\phi](X)$, then $\check{\varphi}_\mathcal{N}(X) = \hat{\varphi}[d_{c_{\mathtt{t}_3}}^\phi](X) \in [l_1, u_1]$; so, $\check{\varphi}_\mathcal{N}$ would satisfy $c_{\mathtt{t}_3}$. Otherwise, $c_{\mathtt{t}_3} \notin \arg\max_{c \in \mathcal{C}_{\mathtt{t}_3}} \hat{\varphi}[d_c^\phi](X)$, and assume $\check{\varphi}_\mathcal{N}(X) \notin [l_1, u_1]$ towards a contradiction. Pick any $c_X^* \in \arg\max_{c \in \mathcal{C}_{\mathtt{t}_3}} \hat{\varphi}[d_c^\phi](X)$. All these hypotheses combined:

$$\check{\varphi}_\mathcal{N}(X) = \hat{\varphi}[d_{c_X^*}^\phi](X) > u_1.$$

Therefore, ϕ must satisfy either $p \Rightarrow \neg x_{c_{\mathtt{t}_3}}$ or $\neg p \Rightarrow \neg x_{c_{\mathtt{t}_3}}$, for some boolean variable p (where the actual case depends on the actual value of $d_{c_X^*}^\phi$). Since ϕ satisfies either p or $\neg p$, then ϕ must satisfy $\neg x_{c_{\mathtt{t}_3}}$; i.e., $\phi(x_{c_{\mathtt{t}_3}}) = \mathtt{false}$. This is absurd, as we assumed $\phi(x_{c_{\mathtt{t}_3}}) = \mathtt{true}$.

The proof of the other case, in which $\phi(x_{c_{\mathtt{t}_3}}) = \mathtt{false}$ is initially assumed, is symmetric. So, $\check{\varphi}_\mathcal{N}(X)$ satisfies $c_{\mathtt{t}_3}$. ◀

Let us mention that our algorithm is called `RDTP()`, basically, it aims at computing $\hat{\varphi}_{\mathcal{N}}$ as above; if it fails in that (either because $\mathcal{N}_{\mathsf{t}_2}$ is already inconsistent or $\mathrm{CL}_{\mathcal{N}}$ is unsatisfiable), it recognizes the input RDTP $\mathcal{N}$ as inconsistent. Now, we can prove this is correct and fast.

▶ **Theorem 22.** *`RDTP()` is correct, i.e., on any RDTN $\mathcal{N} = (\mathcal{T}, \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2} \cup \mathcal{C}_{t_3})$, it returns a feasible schedule $\hat{\varphi}_{\mathcal{N}} : \mathcal{T} \to \mathbb{R}$, if $\mathcal{N}$ is consistent; otherwise, $\mathcal{N}$ is recognized as inconsistent.*

Proof. Recall that $\mathcal{N}$ is recognized as inconsistent only if $(\mathcal{T}, \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2})$ is already inconsistent or if the 2-SAT problem instance $\mathrm{CL}_{\mathcal{N}}$ is unsatisfiable. In the former case, since $(\mathcal{T}, \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2})$ is inconsistent, so it is $\mathcal{N}$. In the latter, by construction of $\mathrm{CL}_{\mathcal{N}}$, it is not possible to satisfy all the constraints in $\mathcal{C}_{t_1} \cup \mathcal{C}_{t_2} \cup \mathcal{C}_{t_3}$ (otherwise, the reader can check, it would've been possible to construct a satisfying assignment for $\mathrm{CL}_{\mathcal{N}}$, straightforwardly); so, $\mathcal{N}$ is really inconsistent.

On the other side, by Propositions 19, 20 and 21, schedule $\hat{\varphi}_{\mathcal{N}}$ is really feasible for $\mathcal{N}$. ◀

The next result asserts that the halting time is strongly polynomial in the input size.

▶ **Theorem 23.** *Let `RDTP()` run on any input RDTP $\mathcal{N} = (\mathcal{T}, \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2} \cup C_3)$.*
Its always halts within time $O\big(|\mathcal{T}| \cdot |\mathcal{C}_{t_1}| + |\mathcal{C}_{t_2}| \cdot (|\mathcal{C}_{t_1}| + |\mathcal{T}| \cdot \log |\mathcal{T}|) + |\mathcal{T}| \cdot d_{\mathcal{C}_{t_2}} \cdot |\mathcal{C}_{t_3}| + |\mathcal{C}_{t_3}|^2\big)$.

Proof. By Theorem 16, $(\mathcal{T}, \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2})$ takes $O\big(|\mathcal{T}| \cdot |\mathcal{C}_{t_1}| + |\mathcal{C}_{t_2}| \cdot (|\mathcal{C}_{t_1}| + |\mathcal{T}| \cdot \log |\mathcal{T}|) + |\mathcal{T}| \cdot d_{\mathcal{C}_{t_2}}\big)$ time to be checked. Using that solution as an initial candidate, solving the two t_2DTPs $\mathcal{N}[d'_c]_{\mathsf{t}_2}$ and $\mathcal{N}[d''_c]_{\mathsf{t}_2}$, for each $c \in \mathcal{C}_{t_3}$ where $c = d'_c \vee d''_c$, it takes $O\big(|\mathcal{T}| \cdot |\mathcal{C}_{t_1}| + |\mathcal{C}_{t_2}| \cdot (|\mathcal{C}_{t_1}| + |\mathcal{T}| \cdot \log |\mathcal{T}|) + |\mathcal{T}| \cdot d_{\mathcal{C}_{t_2}} \cdot |\mathcal{C}_{t_3}|\big)$ total time. Next, for each $c, \tilde{c} \in \mathcal{C}_{t_3}$ such that $\tilde{c} \neq c$, eventually adding the corresponding clauses to $\mathrm{CL}_{\mathcal{N}}$ takes $O(1)$ time per clause; so, $\mathrm{CL}_{\mathcal{N}}$ is built in total time $O(|\mathcal{C}_{t_3}|^2)$. Since $|\mathrm{CL}_{\mathcal{N}}| = O(|\mathcal{C}_{t_3}|^2)$, solving the 2-SAT problem on input $\mathrm{CL}_{\mathcal{N}}$ takes time $O(|\mathcal{C}_{t_3}|^2)$ (e.g., with the algorithm of [1]). Finally, computing d_c^{ϕ} and $\check{\varphi}_{\mathcal{N}}$ takes $O(|\mathcal{T}| \cdot |\mathcal{C}_{t_3}|)$ time. All in, the above mentioned time complexity of `RDTP()` follows. ◀

5 NP-completeness of Multi-Tail & Multi-Head t_3HyTPs

This section enquiries the tractability frontier of RDTPs by considering HyTNs [3], where the basic idea is that of blending the two models together and see what happens to the complexity of the corresponding temporal problems. Two restricted kinds of disjunctive temporal problems, TAIL-t_3HyTP and HEAD-t_3HyTP, are both proven to be NP-complete. The former problem is that of deciding whether a multi-tail t_3HyTN (i.e., a temporal network in which the constraints can be modeled only by multi-tail hyperarcs and by t_3-constraints) is consistent or not. The latter, HEAD-t_3HyTP, is the same as the former but considers multi-head hyperarcs instead. Let us now focus on TAIL-t_3HyTP.

▶ **Theorem 24.** *TAIL-t_3HyTP is NP-complete in a strong sense, i.e., even if the input $(\mathcal{T}, \mathcal{A} \cup \mathcal{C}_{t_3})$ are restricted to satisfy $w_A(\cdot) \in \{-1, 0, 1\}$, $|T_A| \leq 2$ for every $A \in \mathcal{A}$, and every t_3-constraint $(l_i \leq X \leq u_i) \vee (l_j \leq Y \leq u_j) \in \mathcal{C}_{t_3}$ has all zero-valued lower/upper-bounds.*

Proof. We claim that if $\mathcal{H} = (\mathcal{T}, \mathcal{A} \cup \mathcal{C}_{t_3})$ is an integer-weighted and consistent multi-tail t_3HyTN, it admits an integer-valued feasible schedule $s : \mathcal{T} \to \{-T, \ldots, T\}$ where $T = \sum_{A \in \mathcal{A}, v \in V} |w_A(v)| + \sum_{c \in \mathcal{C}_{t_3}, c = (l_1 \leq X \leq u_1) \vee (l_2 \leq Y \leq u_2)} (|l_1| + |u_1| + |l_2| + |u_2|)$. Indeed let s be a feasible schedule (integer-valued or not) of $\mathcal{H}$, and consider the projection HyTN $\mathcal{H}^s \triangleq (\mathcal{T}, \mathcal{A}')$, for $\mathcal{A}' \triangleq \mathcal{A} \cup \bigcup_{c \in \mathcal{C}_{t_3}} A_c^s$, where for every $c = (l_1 \leq X \leq u_1) \vee (l_2 \leq Y \leq u_2) \in \mathcal{C}_{t_3}$ we pick the following pair of t_1-constraints:
$$A_c^s \triangleq \begin{cases} \{(Z - X \leq -l_1), (X - Z \leq u_1)\}, & \text{if } l_1 \leq s(X) \leq u_1; \\ \{(Z - Y \leq -l_2), (Y - Z \leq u_2)\}, & \text{otherwise.} \end{cases}$$
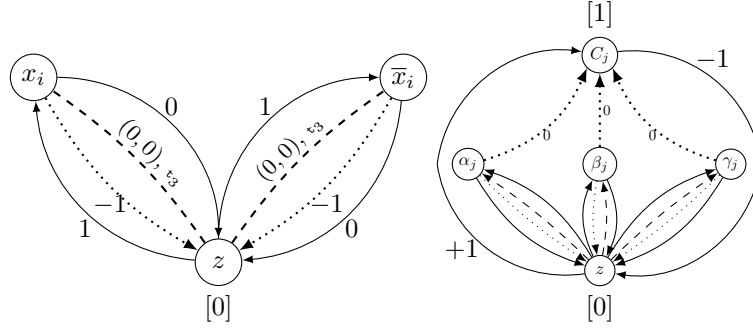
Figure 4 Variable and clause gadgets (at left and right, respectively) used in Theorem 24.

By construction of $\mathcal{H}^s$, s is a feasible for HyTN $\mathcal{H}^s$. So, by Proposition 7, $\mathcal{H}^s$ admits an integer-valued feasible schedule s' bounded by $-T$ and $+T$ as above. By contruction of $\mathcal{H}^s$, s' is feasible for $\mathcal{H}$ too.

Moreover, any such integer-valued feasible schedule can be verified in strongly polynomial time w.r.t. the size of the input; hence, TAIL-$\mathtt{t_3}$HyTP is in NP.

To show that the problem is NP-hard, we describe a reduction from 3-SAT.

Let us consider a boolean 3-CNF formula with $n \geq 1$ variables and $m \geq 1$ clauses:

$\varphi(x_1, \ldots, x_n) = \bigwedge_{i=1}^{m} (\alpha_i \vee \beta_i \vee \gamma_i)$, where $C_i = (\alpha_i \vee \beta_i \vee \gamma_i)$ is the i-th clause of φ and each $\alpha_i, \beta_i, \gamma_i \in \{x_j, \overline{x}_j \mid 1 \leq j \leq n\}$ is either a positive or a negative literal.

We associate to φ a multi-tail $\mathtt{t_3}$HyTN $\mathcal{H}_\varphi = (\mathcal{T}, \mathcal{A} \cup \mathcal{C}_{\mathtt{t_3}})$, where each boolean variable x_i occurring in φ gets represented by two time-points, x_i and $\overline{x}_i$. $\mathcal{T}$ also contains a time-point z that represents the reference initial time-point for $\mathcal{H}_\varphi$, i.e., the first time-point that has to be executed at time zero. Moreover, for each pair x_i and $\overline{x}_i$, $\mathcal{H}_\varphi$ contains:

a multi-tail hyperarc with tails $\{x_i, \overline{x}_i\}$, both weighted -1, and head in z.

a $\mathtt{t_3}$-constraint $\big((0 \leq x_i \leq 0) \vee (0 \leq \overline{x}_i \leq 0)\big) \in \mathcal{C}_{\mathtt{t_3}}$. If $\mathcal{H}_\varphi$ is consistent, the multi-tail hyperarc and the $\mathtt{t_3}$-constraint associated to $x, \neg x$ assures that $\mathcal{H}_\varphi$ admits an integer feasible schedule s (as we mentioned above) such that $s(x_i)$ and $s(\overline{x}_i)$ are coherently set with values in $\{0, 1\}$. In this way, s is forced to encode a truth assignment on the x_i's.

The HyTN $\mathcal{H}_\varphi$ contains also a time-point C_j for each clause C_j of φ; each C_j is connected by a multi-tail hyperarc with head in C_j and tails over the literals occurring in C_j and by two standard and opposite arcs with time-point z as displayed in Fig. 4 (right). This assures that if $\mathcal{H}_\varphi$ admits a feasible schedule s, then s assigns scheduling time 1 at least to one of the time-point representing the literals connected with the multi-tail hyperarc.

Fig. 4 depicts the gadgets. A more formal definition of $\mathcal{H}_\varphi$ is given in Appendix A.

The reader can check that $|\mathcal{T}| = 1 + 2n + m = O(m+n)$, $m_\mathcal{A} = O(m+n)$, $|\mathcal{C}_{\mathtt{t_3}}| = O(n)$; therefore, the transformation is linearly bounded.

We next show that φ is satisfiable if and only if $\mathcal{H}_\varphi$ is consistent.

Any truth assignment $\nu : \{x_1, \ldots, x_n\} \to \{\texttt{true}, \texttt{false}\}$ satisfying φ can be translated into a feasible schedule $s : \mathcal{T} \to \mathbb{Z}$ of $\mathcal{H}_\varphi$ as follows. For time-point z, let $s(z) = 0$, and let $s(C_j) = 1$ for each $j = 1, \ldots, m$; then, for each $i = 1, \ldots, n$, let $s(x_i) = 1$ and $s(\overline{x}_i) = 0$ if the truth value of x_i, $\nu(x_i)$, is $\texttt{true}$, otherwise let $s(x_i) = 0$ and $s(\overline{x}_i) = 1$. It is simple to verify that, using this schedule s, all the constraints comprising each single gadget are satisfied and, therefore, the network is consistent. So, $\mathcal{H}_\varphi$ is consistent.

Vice versa, assume that $\mathcal{H}_\varphi$ is consistent. Then, it admits an integer-valued feasible schedule s (as we mentioned above). After the translation $s(v) \triangleq s(v) - s(z)$, we can assume that $s(z) = 0$. Hence, $s(C_j) = 1$ for each $j = 1, \ldots, m$, as enforced by the two standard

arcs incident at C_j in the clause gadget, and $\{s(x_i), s(\overline{x}_i)\} = \{0, 1\}$ for each $i = 1, \ldots, n$, as enforced by the constraints comprising the variable gadgets. Therefore, the feasible schedule s can be translated into a truth assignment $\nu : \{x_1, \ldots, x_n\} \to \{\texttt{true}, \texttt{false}\}$ defined by $\nu(x_i) = \texttt{true}$ if $s(x_i) = 1$ (and $s(\overline{x}_i) = 0$); $\nu(x_i) = \texttt{false}$ if $s(x_i) = 0$ (and $s(\overline{x}_i) = 1$) for every $i = 1, \ldots, n$. So, φ is satisfiable.

To conclude, we observe that any hyperarc $A \in \mathcal{A}$ of $\mathcal{H}_\varphi$ has weights $w_A(\cdot) \in \{-1, 0, 1\}$, size $|A| \leq 3$, and any $\texttt{t}_3$-constraint $c = (l_i \leq X \leq u_i) \vee (l_j \leq Y \leq u_j) \in \mathcal{C}_{\texttt{t}_3}$ has zero lower and upper-bounds (i.e., $l_i = u_i = l_j = u_j = 0$). Since any hyperarc with three tails can be replaced by two hyperarcs each having at most two tails, the consistency problem remains NP-Complete even if $|A| \leq 2$ for every $A \in A$.   ◀

In order to prove that HEAD-$\texttt{t}_3$HYTP is also NP-complete, we could proceed with an argument similar to that of Theorem 24. However, we also observe that the same result follows as an immediate corollary of the following inter-reducibility between the two models.

▶ **Definition 25.** A multi-tail (multi-head) RHyTN is any temporal network in which the constraints can be modeled only by multi-tail (multi-head) hyperarcs and by $\{\texttt{t}_2, \texttt{t}_3\}$ disjunctive temporal constraints.

The problem of checking whether a given RHyTN is consistent is named RHyTP. Observe,

▶ **Proposition 26.** *Multi-head and multi-tail RHyTPs are inter-reducible by means of log-space, linear-time, local-replacement reductions. Particularly, multi-head and multi-tail t_3HyTPs are inter-reducible by such reductions. (The proof is in Appendix A)*

Therefore, by Proposition 26, it follows that HEAD-$\texttt{t}_3$HYTP is also strongly NP-complete.

6   Pseudo-Polynomial Time Algorithm for $\texttt{t}_2$HyTPs

We end by studying multi-tail and multi-head $\texttt{t}_2$HyTNs (i.e., temporal networks in which the temporal constraints can be only $\texttt{t}_2$ disjunctive temporal constraints and either only multi-tail or multi-head hyperarcs). It turns out that checking the corresponding temporal problems, TAIL-$\texttt{t}_2$HYTP and HEAD-$\texttt{t}_2$HYTP, lies in NP $\cap$ co-NP and admits pseudo-polynomial time algorithms. By Proposition 26, it is sufficient to focus on multi-head $\texttt{t}_2$HyTPs only. The corresponding pseudo-polynomial time algorithm is named $\texttt{t}_2\texttt{HyTP()}$, and described below – notice that it generalizes $\texttt{t}_2\texttt{DTP()}$. Given any integer-weighted multi-head $\texttt{t}_2$HyTPs $\mathcal{H}_{\texttt{t}_2} = (\mathcal{T}, \mathcal{A} \cup \mathcal{C}_{\texttt{t}_2})$ in input, we firstly solve the HyTP $\mathcal{H} = (\mathcal{T}, \mathcal{A})$ with the VI algorithm of Theorem 10. If $\mathcal{H}$ is recognized as inconsistent, the algorithm halts. Otherwise, let φ be the least feasible schedule of $\mathcal{H}$. Then proceed as follows:

While $\exists$ some $X \in \mathcal{T}$ and $c_X = \bigvee_{i=1}^{k}(l_i \leq X \leq u_i) \in \mathcal{C}_{\texttt{t}_2}$ s.t. $\varphi(X)$ doesn't satisfy c_X:

If $\varphi(X) > u_k(= \max_i u_i)$, then $\mathcal{H}_{\texttt{t}_2}$ is recognized as inconsistent; otherwise, let i^* be the smallest $i \in [1, k]$ such that $\varphi(X) < l_i$. Firstly, we increase the value of $\varphi(X)$ up to l_{i^*}, i.e., update $\varphi(X) \leftarrow l_{i^*}$. Secondly, the VI algorithm of Theorem 10 is invoked on input $(\mathcal{H}, \varphi)$, so, then, φ becomes the schedule returned by that run of VI. The process iterates so on and so forth, and if the while-loop completes without recognizing $\mathcal{H}_{\texttt{t}_2}$ as inconsistent, φ is returned. The correctness and the time complexity are asserted below. (The proof is in Appendix A)

▶ **Theorem 27.** *$t_2HyTP()$ is correct, i.e., running on any integer-weighted multi-head t_2HyTP $\mathcal{H}_{t_2} = (\mathcal{T}, \mathcal{A} \cup \mathcal{C}_{t_2})$, an integer-valued feasible schedule $\varphi : \mathcal{T} \to \mathbb{Z}$ is returned, in case $\mathcal{H}_{t_2}$ is consistent; otherwise, $\mathcal{H}_{t_2}$ is correctly recognized as inconsistent.*

Moreover, the corresponding time complexity is pseudo-polynomial, i.e.,

$$Time_{t_2HyTP()}(\mathcal{H}_{t_2}) = O\big((|\mathcal{T}| + |\mathcal{A}|) \cdot m_{\mathcal{A}} \cdot W_{\mathcal{A},\mathcal{C}_{t_2}}\big),$$

$$where\ W_{\mathcal{A},\mathcal{C}_{t_2}} \triangleq \max\Big(\max_{A \in \mathcal{A}} \max_{h \in A} |w_A(h)|, \max_{\substack{l_j\ appears\ in\ any \\ \vee_{i=1}^{k}(l_i \leq X \leq u_i) \in \mathcal{C}_{t_2}}} l_j\Big).$$

Finally, since `t`$_2$`HyTP()` is correct, it is possible to establish the following complexity result.

▶ **Theorem 28.** TAIL-T$_2$HYTP, HEAD-T$_2$HYTP $\in$ NP$\cap$co-NP. *(The proof is in Appendix A)*

7   Conclusions and Future Works

A deeper combinatorial comprehension on the algorithmics of RDTPs led to a new elementary deterministic strongly polynomial time procedure for solving them, significantly improving the asymptotic running times suggested by Kumar before. In future works we'd like to investigate further on possible generalizations/extensions of the proposed algorithms, aiming at covering some compatible (or even wider) subclasses of the disjunctive temporal constraints problem.

── **References** ──

1   Bengt Aspvall, Michael F. Plass, and Robert Endre Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Information Processing Letters*, 8(3):121–123, 1979. `doi:10.1016/0020-0190(79)90002-4`.

2   Richard Bellman. On a routing problem. *Quarterly of Applied Maths*, 16(1):87–90, 1958.

3   Carlo Comin, Roberto Posenato, and Romeo Rizzi. Hyper temporal networks - A tractable generalization of simple temporal networks and its relation to mean payoff games. *Constraints*, 22(2):152–190, 2017. `doi:10.1007/s10601-016-9243-0`.

4   Rina Dechter. *Constraint Processing.* Morgan Kaufmann, San Francisco, CA, US, 2003.

5   Rina Dechter, Itay Meiri, and Judea Pearl. Temporal constraint networks. *Artificial Intelligence*, 49(1-3):61–95, 1991. `doi:10.1016/0004-3702(91)90006-6`.

6   Yves Deville, Olivier Barette, and Pascal Van Hentenryck. Constraint satisfaction over connected row-convex constraints. *Artificial Intelligence*, 109(1):243–271, 1999. `doi:10.1016/S0004-3702(99)00012-0`.

7   E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, Dec 1959. `doi:10.1007/BF01386390`.

8   Manolis Koubarakis. Chapter 19 - temporal csps. In Francesca Rossi, Peter van Beek, and Toby Walsh, editors, *Handbook of Constraint Programming*, volume 2 of *Foundations of AI*, pages 665–697. Elsevier, 2006. `doi:10.1016/S1574-6526(06)80023-4`.

9   T. K. Sathish Kumar. Tractable classes of metric temporal problems with domain rules. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, AAAI'06, pages 847–852. AAAI Press, 2006.

10   T. K. Satish Kumar. On the tractability of restricted disjunctive temporal problems. In *Proceedings of the Fifteenth International Conference on Automated Planning and Scheduling (ICAPS 2005), June 5-10 2005, Monterey, California, USA*, pages 110–119, 2005.

11   T. K. Satish Kumar, Marcello Cirillo, and Sven Koenig. Simple temporal problems with taboo regions. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, AAAI'13, pages 548–554. AAAI Press, 2013.

12   Dana Nau, Malik Ghallab, and Paolo Traverso. *Automated Planning: Theory & Practice.* Morgan Kaufmann, San Francisco, CA, USA, 2004.

13 Angelo Oddi and Amedeo Cesta. Incremental forward checking for the disjunctive temporal problem. In *Proceedings of the 14th European Conference on Artificial Intelligence*, ECAI'00, pages 108–112, Amsterdam, The Netherlands, The Netherlands, 2000. IOS Press.

14 A.K. Pani and G.P. Bhattacharjee. Temporal representation and reasoning in artificial intelligence: A review. *Mathematical and Computer Modelling*, 34(1):55–80, 2001.

15 E. Schwalb and L. Vila. Temporal constraints: A survey. *Constraints*, 3(2):129–149, 1998.

16 Kostas Stergiou and Manolis Koubarakis. Backtracking algorithms for disjunctions of temporal constraints. *Artificial Intelligence*, 120(1):81–117, 2000.

17 Ioannis Tsamardinos and Martha E Pollack. Efficient solution techniques for disjunctive temporal reasoning problems. *Artificial Intelligence*, 151(1):43–89, 2003.

A Appendix: Omitted Proofs

The appendix proceeds by offering additional missing proofs.

Proof of Proposition 26. We show the reduction from multi-tail to multi-head hypergraphs; the converse direction is symmetric. Informally, all the arcs are reversed (so that what was multi-tail becomes multi-head), and, contextually, the time-axis is inverted (to account for the inversion of the direction of all arcs). Finally, all $\mathtt{t}_2$ and $\mathtt{t}_3$-constraints are also reversed.

Given a multi-tail RHyTN $\mathcal{H} = (\mathcal{T}, \mathcal{A} \cup \mathcal{C}_{\mathtt{t}_2} \cup \mathcal{C}_{\mathtt{t}_3})$, we associate to $\mathcal{H}$ a multi-head RHyTN $\mathcal{H}' = (\mathcal{T}, \mathcal{A}' \cup \mathcal{C}_2' \cup \mathcal{C}_3')$ by reversing all multi-tail hyperarcs, all $\mathtt{t}_2$ and $\mathtt{t}_3$-constraints. Formally,

$$\mathcal{A}' \triangleq \Big\{ (v, S, w) \mid (S, v, w) \in \mathcal{A} \Big\}, \quad \mathcal{C}_2' \triangleq \Big\{ \bigvee_{i=1}^{k} (-u_i \leq X \leq -l_i) \mid \bigvee_{i=1}^{k} (l_i \leq X \leq u_i) \in \mathcal{C}_{\mathtt{t}_2} \Big\},$$

$$\mathcal{C}_3' \triangleq \Big\{ \big((-u_1 \leq X \leq -l_1) \vee (-u_2 \leq Y \leq -l_2) \big) \mid \big((l_1 \leq X \leq u_1) \vee (l_2 \leq Y \leq u_2) \big) \in \mathcal{C}_{\mathtt{t}_3} \Big\}.$$

We claim that $\mathcal{H}$ is consistent if and only if $\mathcal{H}'$ is consistent. To prove it, we note that each schedule s for $\mathcal{H}$ can be associated, with a flip of the time direction, to the schedule $s' \triangleq -s$. Then, it holds that s is feasible for $\mathcal{H}$ if and only if s' is feasible for $\mathcal{H}'$. Indeed, s satisfies the constraint represented by an hyperarc $A = (T_A, h_A, w_A) \in \mathcal{A}$, i.e.,

$s(h_A) \leq \max_{v \in T_A} \{ s(v) + w_A(v) \}$,

or, equivalently, $-s(h_A) \geq \min_{v \in T_A} \{ -s(v) - w_A(v) \}$, if and only if s' (that is, $-s$) satisfies the constraint represented by the reversed hyperarc $A' = (h_A, T_A, w_A)$, i.e., if and only if:

$s'(h_A) \geq \min_{v \in T_A} \{ s'(v) - w_{A'}(v) \}$.

Next, s satisfies a $\mathtt{t}_2$-constraint $\bigvee_{i=1}^{k} (l_i \leq X \leq u_i)$ *iff* $l_i \leq s(X) \leq u_i$ holds for some $i \in [1, k]$, or equivalently, *iff* $-u_i \leq -s(X) \leq -l_i$; this happens *iff* s' satisfies the constraint represented by the reversed disjunct $(-u_i \leq X \leq -l_i)$, i.e., *iff* $-u_i \leq s'(X) \leq -l_i$.

Finally, s satisfies a $\mathtt{t}_3$-constraint $\big((l_1 \leq X \leq u_1) \vee (l_2 \leq Y \leq u_2) \big)$ *iff* either $l_1 \leq s(X) \leq u_1$ or $l_2 \leq s(Y) \leq u_2$, or equivalently, either $-u_1 \leq -s(X) \leq -l_1$ or $-u_2 \leq -s(Y) \leq -l_2$; this happens *iff* s' satisfies the constraint represented either by the reversed disjunct $(-u_1 \leq X \leq -l_1)$ or by $(-u_2 \leq Y \leq -l_2)$, i.e., *iff* either $-u_1 \leq s'(X) \leq -l_1$ or $-u_2 \leq s'(Y) \leq -l_2$. ◀

Formal definition of $\mathcal{H}_\varphi$ in the proof of Theorem 24. More formally, $\mathcal{H}_\varphi = (\mathcal{T}, \mathcal{A} \cup \mathcal{C}_{\mathtt{t}_3})$ is:

- $\mathcal{T} = \{z\} \cup \{x_i \mid 1 \leq i \leq n\} \cup \{\overline{x}_i \mid 1 \leq i \leq n\} \cup \{C_j \mid 1 \leq j \leq m\}$;
- $\mathcal{A} = \bigcup_{i=1}^{n} \mathrm{Var}_i \cup \bigcup_{j=1}^{m} \mathrm{Cla}_j$, where:

- $\mathrm{Var}_i = \Big\{(z, x_i, 1), (x_i, z, 0), (z, \overline{x}_i, 1), (\overline{x}_i, z, 0),$

  $\big(\{x_i, \overline{x}_i\}, z, [w(x_i), w(\overline{x}_i)] = [-1, -1]\big)\Big\}.$

  This is for the variable gadget of x_i as depicted in Fig. 4 (left);
- $\mathrm{Cla}_j = \Big\{(z, C_j, 1), (C_j, z, -1),$

  $\big(\{\alpha_j, \beta_j, \gamma_j\}, C_j, [w(\alpha_j), w(\beta_j), w(\gamma_j)] = [0, 0, 0]\big)\Big\}.$

  This defines the clause gadget for clause $C_j = (\alpha_i \vee \beta_i \vee \gamma_i)$ as in Fig. 4 (right).
- $\mathcal{C}_{\mathtt{t}_3} = \bigcup_{i=1}^n \mathrm{Var}_i'$, where:
  - $\mathrm{Var}_i' = \big\{\big((0 \le x_i \le 0) \vee (0 \le \overline{x}_i \le 0)\big)\big\}.$

    This completes the variable gadget of x_i as depicted in Fig. 4 (left); ◀

Proof of head-$\mathtt{t}_2$HyTP, tail-$\mathtt{t}_2$HyTP $\in$ NP. We claim that if $\mathcal{H} = (\mathcal{T}, \mathcal{A} \cup \mathcal{C}_{\mathtt{t}_2})$ is an integer-weighted and consistent multi-tail $\mathtt{t}_2$HyTN, it admits an integer-valued feasible schedule $s : \mathcal{T} \to \{-T, \dots, T\}$ where $T = \sum_{A \in \mathcal{A}, v \in V} |w_A(v)| + \sum_{c \in \mathcal{C}_{\mathtt{t}_2}, c = \bigvee_{i=1}^k (l_i \le X \le u_i)} (|l_i| + |u_i|)$. Indeed, let s be a feasible schedule (integer-valued or not) of $\mathcal{H}$, and consider the projection HyTN:

$$\mathcal{H}^s \triangleq (\mathcal{T} \cup \{z\}, \mathcal{A}'),$$
$$\mathcal{A}' \triangleq \mathcal{A} \cup \{(z - T \le 0) \mid T \in \mathcal{T}\} \cup \bigcup_{c \in \mathcal{C}_{\mathtt{t}_2}} A_c^s.$$

where for every $c = \bigvee_{i=1}^k (l_i \le X \le u_i) \in \mathcal{C}_{\mathtt{t}_2}$ this pair of $\mathtt{t}_1$-constraints is taken:

$$A_c^s \triangleq \big\{(z - X \le -l_i), (X - z \le u_i) \mid \text{ for the smallest } i \in \{1, \dots, k\} \text{ s.t. } l_i \le s(X) \le u_i\big\}.$$

By construction of $\mathcal{H}^s$, s is a feasible for HyTN $\mathcal{H}^s$. So, by Proposition 7, $\mathcal{H}^s$ admits an integer-valued feasible schedule s' bounded as above. By construction of $\mathcal{H}^s$, s' is feasible for $\mathcal{H}$ too.

Any such integer-valued feasible schedule can be verified in strongly polynomial time w.r.t. the size of the input, simply by checking the actual consistency of each constraint in $\mathcal{A} \cup \mathcal{C}_{\mathtt{t}_2}$; hence, HEAD-$\mathtt{T}_2$HYTP is in NP. Thus, by Proposition 26, TAIL-$\mathtt{T}_2$HYTP $\in$ NP. ◀

▶ **Definition 29** (Hypercycle). We recall from [3] that a *hypercycle* $\mathcal{C}_0$ in a HyTN $\mathcal{H}$ is actually a pair $(S, \mathcal{C}_0)$ with $S \subseteq \mathcal{T}$ and $\mathcal{C}_0 \subseteq \mathcal{A}$ such that:
1. $S = \cup_{A \in \mathcal{C}_0} A$ and $S \ne \emptyset$;
2. $\forall v \in S$ there exists an unique $A \in \mathcal{C}_0$ such that $t_A = v$.
Every infinite path in a cycle $(S, \mathcal{C})$ contains, at least, one *finite cyclic sequence* $v_i, v_{i+1}, \dots, v_{i+p}$, where $v_{i+p} = v_i$ is the only repeated node in the sequence. A cycle $(S, \mathcal{C}_0)$ is *negative* if for any finite cyclic sequence $v_1, v_2, \dots, v_p$, it holds that $\sum_{t=1}^{p-1} w_{a(v_t)}(v_{t+1}) < 0$, where $a(v)$ denotes the unique arc $A \in \mathcal{C}_0$ with $t_A = v$ as required in previous item 2.

▶ **Definition 30** (Certified Least Feasible Schedule (CLFS)). Given any integer-weighted multi-head HyTN $\mathcal{H} = (\mathcal{T}, \mathcal{A})$, a *certified least feasible schedule (CLFS)* for $\mathcal{H}$ is a pair $\varphi_{\mathrm{cert}} \triangleq \{\varphi, \mathcal{F}\}$, where $\varphi : \mathcal{T} \to \mathbb{Z}$ is a feasible schedule of $\mathcal{H}$, and $\mathcal{F} \triangleq \{\mathcal{C}_X\}_{X_{\mathcal{T}}}$ is a family of hypercycles of $\mathcal{H}$ (which works as a certificate of minimality for φ, as follows): for every $X \in \mathcal{T}$, $\mathcal{C}_X$ is a negative hypercycle of the auxiliary HyTN $\mathcal{H}_X$ obtained from $\mathcal{H}$ just by adding one $\mathtt{t}_1$-constraint requiring X to be scheduled strictly before time $\varphi(X)$,

$$\mathcal{H}_X \triangleq \big(\mathcal{T} \cup \{z\}, \mathcal{A} \cup \{(z - T \le 0) \mid T \in \mathcal{T}\} \cup \{X - z \le \varphi(X) - 1\}\big).$$

φ_{cert} can be verified in strongly polynomial time, because negative hypercycles can be checked so (as shown e.g., in Lemma 3 of [3]) and feasiblity of φ can be checked by inspection. The

soundness of CLFSs follows from the proof argument of Proposition 7 (i.e., the idea in this proof is – cfr. Lemma 2 in [3] – to project the feasible HyTN over a conservative graph and then, in that setting, to exploit the integrality properties of potentials as prescribed e.g., by the Bellman-Ford algorithm) plus the fact that the universe of feasible schedules of any given multi-head HyTN is closed under pointwise-minimum, i.e., given two feasible schedules s_1, s_2, the schedule $s(u) \triangleq \min(s_1(u), s_2(u)) \; \forall u \in \mathcal{T}$, is still feasible (also notice that, in multi-tail HyTNs, the pointwise-maximum works instead).

With Definitions 29 and 30 in mind, we can proceed with the following proof.

Proof of head-$\mathtt{t}_2$HyTP, tail-$\mathtt{t}_2$HyTP $\in$ co-NP. To show HEAD-$\mathtt{T}_2$HyTP $\in$ co-NP, we shall exhibit certificates (of inconsistent networks) which we can verify in strongly polynomial time.

The basic idea, in order to construct such certificates, is to consider what happens during the execution of algorithm $\mathtt{t_2HyTP()}$, assuming the input instance $\mathcal{H} = (\mathcal{T}, \mathcal{A} \cup \mathcal{C}_{\mathtt{t}_2})$ is inconsistent. If the HyTN $\mathcal{H}_0 \triangleq (\mathcal{T}, \mathcal{A})$ is already inconsistent, then it admits a negative hypercycle $\mathcal{C}_0$ (see e.g., Theorem 4 in [3]). Moreover, $\mathcal{C}_0$ can be checked in strongly polynomial time (see e.g., Lemma 3 in [3]), so $\mathcal{C}_0$ is already a valid certificate of inconsistency. Otherwise, let φ_0 be a CLFS of $\mathcal{H}_0$, then there must exist some $X_0 \in \mathcal{T}$ and $c_{X_0} = \bigvee_{i=1}^{k}(l_i \leq X_0 \leq u_i) \in \mathcal{C}_{\mathtt{t}_2}$ s.t. $\varphi_0(X_0)$ doesn't satisfy c_{X_0}. If $\varphi_0(X_0) > u_k(= \max_i u_i)$, then (φ_0, c_{X_0}) is a valid certificate of inconsistency; otherwise, let i_0^* be the smallest $i \in [1, k]$ such that $\varphi_0(X_0) < l_i$. Let $\mathcal{H}_1$ be the auxiliary HyTN obtained from $\mathcal{H}_0$ just by adding one $\mathtt{t}_1$-constraint requiring X_0 to be scheduled at or after time $l_{i_0^*}$:

$$\mathcal{H}_1 \triangleq \big(\mathcal{T} \cup \{z\}, \mathcal{A} \cup \{(z - T \leq 0) \mid T \in \mathcal{T}\} \cup \{z - X_0 \leq -l_{i_0^*}\}\big).$$

Then, let φ_1 be a CLFS of $\mathcal{H}_1$ (notice φ_1 exists because $\mathcal{H}_0$ was assumed to be feasible). Again, there must exist some $X_1 \in \mathcal{T}$ and $c_{X_1} = \bigvee_{i=1}^{k}(l_i \leq X_1 \leq u_i) \in \mathcal{C}_{\mathtt{t}_2}$ s.t. $\varphi_1(X_1)$ doesn't satisfy c_{X_1}. Again, if $\varphi_1(X_1) > u_k(= \max_i u_i)$, then $(\varphi_0, c_{X_0}, \varphi_1, c_{X_1})$ is a valid certificate of inconsistency; otherwise, we can construct yet another auxiliary HyTN $\mathcal{H}_2$ by adding one $\mathtt{t}_1$-constraint requiring X_1 to be scheduled at or after time $l_{i_1^*}$, for appropriate i_1^* defined similarly as before. The construction iterates inductively and, generally, it leads to a sequence of the following kind (where the $\{\varphi_i\}_{i=0}^{N}$ are all CLFSs of the auxiliary HyTNs):

$$\Big((\varphi_0, c_{X_0}), (\varphi_1, c_{X_1}), \ldots, (\varphi_N, c_{X_N})\Big),$$

where, notice, it's length is at most $N \leq d_{\mathcal{C}_{\mathtt{t}_2}}$ (because, for each iteration of the construction, one disjunct of some $\mathtt{t}_2$-constraint is ruled out forever). Each element of the sequence can be verified in strongly polynomial time, and the length of the same sequence is strongly polynomial; plus, Theorem 27 implies the correctness of these certificates. This proves that HEAD-$\mathtt{T}_2$HyTP $\in$ co-NP. Thus, by Proposition 26, TAIL-$\mathtt{T}_2$HyTP $\in$ co-NP too.   ◄

Proof of Theorem 27. The correctness argument is similar to that for proving correctness of $\mathtt{t_2DTP()}$, the details are simpler in this case because the only algorithm that is used to update the schedule φ monotonically is the VI algorithm of Theorem 10 (instead of Bellman-Ford and multiple calls to Dijkstra as it was for $\mathtt{t_2DTP()}$); indeed, the VI algorithm of Theorem 10 also provides the *least* feasible schedule in case the input HyTN is consistent, thus a similar (actually simpler) correctness argument still holds. Also the time complexity of $\mathtt{t_2HyTP()}$ is a direct consequence of the complexity of the VI algorithm of Theorem 10, where the maximal weight measure W is increased to $W_{\mathcal{A},\mathcal{C}_{\mathtt{t}_2}}$ (as defined above) in order to take into account the lower-bound constraints (i.e., those of type $(z - X \leq -l_{i^*})$) that are (implicitly) introduced in $\mathcal{A}$ during the main while-loop; notice that, during the computation, the VI algorithm

is invoked on input $(\mathcal{H}, \varphi)$ so that at each iteration the scheduling values are initialized to those of the previous iteration (this ensures that they are always updated monotonically upwards during the whole computation, thus amortizing the total cost among all iterations). Plus, at each iteration at least one scheduling value is increased (i.e., $\varphi(X)$ is increased to l_{i^*} to satisfy the last $(z - X \leq -l_{i^*})$). Finally, checking the while-loop's condition takes time $O(d_{\mathcal{C}_{t_2}} \cdot |\mathcal{T}|)$ total time, and since $d_{\mathcal{C}_{t_2}} \leq |\mathcal{T}| \cdot W_{\mathcal{A}, \mathcal{C}_{t_2}}$, then $O(d_{\mathcal{C}_{t_2}} \cdot |\mathcal{T}|) = O(|\mathcal{T}|^2 \cdot W_{\mathcal{A}, \mathcal{C}_{t_2}})$ (which, notice, it is not a bottleneck asymptotically). So, the $\texttt{Time}_{\texttt{t}_2\texttt{HyTP()}}$ bound holds. ◀