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: (t₁) two-variable linear-difference simple constraint; (t₂) single-variable disjunction of many interval constraints; (t₃) 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 {t₁, t₂}-constraints and no t₃-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 t₂-constraints and either only multi-tail or only multi-head hyperarc-constraints lies in NP \cap co-NP and admits deterministic pseudo-polynomial time algorithms; on the other hand, problems having only t₃-constraints and either only multi-tail or only multi-head hyperarc-constraints turns out strongly NP-complete.

2012 ACM Subject Classification G.2.2 Graph Theory, I.2.8 Problem Solving, Control Methods, and Search

Keywords and phrases Restricted Disjunctive Temporal Problems, Simple Temporal Networks, Hyper Temporal Networks, Consistency Checking, Single-Source Shortest-Paths, 2-SAT.

Digital Object Identifier 10.4230/LIPIcs..2018.

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 $C$ of temporal constraints over $T$. Every constraint $c_i \in C$ is a disjunction
of the form $s_{i,1} \lor s_{i,2} \lor \cdots \lor s_{i,T}$, where every $s_{i,j}$ is a simple temporal constraint of the
form $(l_{i,j} \leq X_{h_{i,j}} - X_{h_{i,j}} \leq u_{i,j})$ 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 \[15\]. 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: $(t_1)$ $(Y - X \leq w)$, for $w$
real (a simple temporal difference-constraint); $(t_2) (l_1 \leq X \leq u_1) \lor \cdots \lor (l_k \leq X \leq u_k)$,
for $l_1, u_1$ reals (a single-variable disjunction of many interval-constraints); $(t_3) (l_i \leq X \leq u_i) \lor (l_2 \leq Y \leq u_2)$, 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,
and 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 \[3\]. Particularly, Kumar’s deterministic algorithm for solving RDTPs works
by reducing them into binary Constraint Satisfiability Problems (CSPs) over meta-variables
representing $t_2$ or $t_3$ 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 $O\left(|C_2| + |C_3|)^3 \cdot d_{\text{max}}^2 \cdot |T| \cdot |C_1| \cdot (|C_2| + |C_3|)^2\right)$, where $C_2, C_3, C_3$ is the set of $t_1, t_2, t_3$ constraints (respectively), and $d_{\text{max}}$ is the maximum number of disjuncts possible per single constraint \[10\]. Randomization
reduces the running time to $O\left(|C_2| + |C_3|)^2 \cdot d_{\text{max}}^2 \cdot \delta + |T| \cdot |C_1| \cdot (|C_2| + |C_3|)^2\right)$, where $\delta$
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 $O\left(|T| \cdot |C_1| + |C_2| \cdot (|C_1| + |T| \cdot \log |T|) + |T| \cdot |C_2| + |C_3|\right)$, where $d_{C_2}$ is the total number of disjuncts counting over all $t_2$-constraints. Since $d_{C_2} \leq d_{\text{max}} \cdot |C_2|$, 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 $\{t_1, t_2\}$-constraints and no $t_3$-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
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 $t_2$-constraints and either only multi-tail or only multi-head hyperarc-constraints lies in $NP \cap co-NP$ and admits deterministic pseudo-polynomial time algorithms; on the other hand, solving temporal problems having only $t_3$-constraints and either only multi-tail or only multi-head hyperarc-constraints turns out strongly $NP$-complete. See Table 1 below for a summary.

| Problem       | Complexity | Improved Time Bound                        | Cfr.     |
|---------------|------------|--------------------------------------------|----------|
| $t_2$DTPs     | $P$        | $O(|T| \cdot |C_t| + |C_{t_2}| \cdot ([|C_t| + |T| \cdot \log |T|) + |T| \cdot d_{C_{t_2}})$ | Sect. 3  |
| RDTPs         | $P$        | $O(|T| \cdot |C_t| + |C_{t_2}| \cdot ([|C_t| + |T| \cdot \log |T|) + $ + $|T| \cdot d_{C_{t_2}} \cdot |C_{t_3}| + |C_{t_3}^2|$) | Sect. 4  |
| $t_2$HyTPs    | $NP \cap co-NP$ | $O((|T| + |A|) \cdot m_A \cdot W_{A,t_{t_2}})$ | Sect. 6  |
| $t_3$HyTPs    | $NP$-complete | $n.a.$ (exponential time) | Sect. 5  |

Table 1: Summary of main results.

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 $t_1$) 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 $s : \mathcal{T} \rightarrow \mathbb{R}$ such that $s(Y) \leq s(X) + w_{X,Y}$ for all $(Y - X \leq w_{X,Y}) \in \mathcal{C}$. 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 $a_0, a_1, \ldots, a_{t-1}$ where an head equals a tail, i.e., $h(a_i) = t(a_{i+1})$, iff $j = i + 1 \mod t$; 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 $f : \mathcal{T} \rightarrow \mathbb{R}$; the reduced weight of an arc $a = (t,h,w_a)$ with respect to a schedule $f$ is defined as $w'_a \triangleq w_a - f(h) + f(t)$; A schedule $f$ is feasible iff $w'_a \geq 0$ 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., $\hat{s} : \mathcal{T} \rightarrow \mathbb{R}_{\geq 0}$ 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].

Theorem 1 (4). Let $\mathcal{N} = (\mathcal{T}, \mathcal{C})$ be an STN. The Bellman-Ford (BF) algorithm (cfr. [2]) produces in $O(|T| \cdot |C|)$ time: either the least feasible schedule $\hat{s} : \mathcal{T} \rightarrow \mathbb{R}_{\geq 0}$, 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 $\tau_2$-constraint and $\tau_3$-constraint (respectively).

▶ Definition 2 (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 $\mathcal{C} = \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2} \cup \mathcal{C}_{t_3}$ is a set of restricted disjunctive temporal constraints over $\mathcal{T}$, each being either one of the following three types:

$t_1$ : $(Y - X \leq w_{X,Y})$, where $X, Y \in \mathcal{T}$ and $w_{X,Y} \in \mathbb{R}$;
$t_2$ : $\bigvee_{i=1}^{k}(l_i \leq X \leq u_i)$, where $X \in \mathcal{T}$ and $l_i, u_i \in \mathbb{R}$ for every $i = 1, \ldots, k$;
$t_3$ : $(l_1 \leq X \leq u_1) \lor (l_2 \leq Y \leq u_2)$, 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 $s : \mathcal{T} \rightarrow \mathbb{R}$ 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 $\tau_1$-constraints do coincide with simple temporal constraints of STNs.

We assume w.l.o.g. that the disjuncts of any $\tau_2$-constraint are arranged in ascending order of the end points of their corresponding intervals, i.e., $l_i < l_{i+1} \land u_i < u_{i+1}$ for every $i = 1, \ldots, k$, whenever $\bigvee_{i=1}^{k}(l_i \leq X \leq u_i) \in \mathcal{C}_{t_2}$; these natural orderings on the interval domains of the time-points will be referred to as their nominal ordering. For any $\tau \in \{1, 2, 3\}$, $|\mathcal{C}_\tau|$ denotes the number of $\tau$-constraints (i.e., the cardinality of $\mathcal{C}_\tau$, not the encoding length). Also, for any $c_X \in \mathcal{C}_{t_2}$, $|c_X|$ denotes the number of disjuncts of $c_X$, and $d_{\mathcal{C}_{t_2}} \triangleq \sum_{c_X \in \mathcal{C}_{t_2}} |c_X|$. 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 $t_{<3}$-constraint; so, provided, $c = (l_1 \leq X_i \leq u_1) \lor (l_2 \leq X_j \leq u_2) \in C_{t_{<3}}$, 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 3** (General-HyTN [3]). A directed hypergraph $H$ is a pair $(T, A)$, where $T$ is the set of nodes, and $A$ is the set of hyperarcs. Each hyperarc $A \in A$ 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).
- 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).

![Figure 3 Hyperarcs in Hyper Temporal Networks.](image)

**Figure 3** Hyperarcs in Hyper Temporal Networks.

The cardinality of a hyperarc $A \in A$ 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 $(T, A)$ are $|T|$ and $m_A \triangleq \sum_{A \in A} |A|$ (respectively).

**Definition 4** (General-HyTN [3]). A general-HyTN is a directed hypergraph $H = (T, A)$ where each node $X \in T$ 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, a 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 5** (General-HyTP [3]). Given a general-HyTN $H = (T, A)$, the General Hyper Temporal Problem (General-HyTP) is that of deciding whether or not there exists a schedule $s : T \rightarrow \mathbb{R}$ such that, for every hyperarc $A \in A$, the following hold:
On RDTPs: Faster Algorithms and Tractability Frontier

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

The following theorem states that General-HyTN is NP-complete, in a strong sense.

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

As observed in [3], Theorem 2 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 3, the complexity of checking these two problems turns out to be lower than that for DTPs, i.e., both head-HyTP, Tail-HyTP $\in \text{NP} \cap \text{co-NP}$, instead of being NP-complete.

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

**Definition 6** (head-HyTP). Given a multi-head HyTN $H = (T, A)$, the head-HyTP problem is that of deciding whether or not there exists a schedule $s : T \to \mathbb{R}$ such that:

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

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

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

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 4** ([3]). head-HyTP and Tail-HyTP are inter-reducible by means of log-space, linear-time, local-replacement reductions.

Thus, Theorem 3 extends to multi-tail HyTNs (i.e., they’re checkable in pseudo-poly time).
3 Faster Deterministic Algorithm for $t_2$DTPs

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

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

Definition 7. Any RDTN $N = (T, C_t \cup C_{\bar{t}} \cup C_{\bar{\bar{t}}})$ having $C_{\bar{\bar{t}}} = \emptyset$ is called $t_2$DTN.

So, $t_2$DTNs are denoted simply as $(T, C_t \cup C_{\bar{t}})$. The corresponding temporal problem, i.e., $t_2$DTP, is that of determining whether a given $t_2$DTN is consistent or not.

One possible solution to $t_2$DTPs is Kumar’s reduction from RDTPs to CRCs [10]. Our solution, named $t_2$DTP(), 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 $t_2$DTN $N_{t_2} = (T, C_t \cup C_{\bar{t}})$, we firstly solve the STP $N_{t_1} = (T, C_t)$ (e.g., with BF-VI). If $N_{t_1}$ is consistent, the returned least feasible schedule $\hat{\varphi}_{N}$ provides an initial candidate, the next step in mind being that of satisfying all the $t_2$-constraints. For this, recall that the disjuncts of any $c_{t_{2}} \in C_{t_{2}}$ are arranged according to their nominal ordering, so that we can try to satisfy any given $c_{t_{2}}$ by iteratively picking the next (i.e., in ascending order) unsatisfied disjunct of $c_{t_{2}}$ and by enforcing its lower-bound constraint in an auxiliary STN as if it were a $t_1$-constraint. While there’s an unsatisfied $t_2$-constraint $c_{t_{2}}$, the current candidate schedule is thus increased by the least necessary amount satisfying both $c_{t_{2}}$ and the whole $C_{\bar{t}_{2}}$. It turns out that this can be done efficiently by performing $|C_{t_{2}}|$ calls to the Dijkstra shortest paths algorithm [7]. In order to show this, let us point out two key facts (i.e., Lemma 1 and 2).

Lemma 1. Let $N = (T, C_t)$ be any STN, and let $\varphi, \varphi'$ be any pair of schedules of $N$.

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

Then for every $T, X \in T$, either $\delta^\varphi_X(T)$ and $\delta^\varphi_X(T)$ are both $+\infty$, or the following holds:

$$\delta^\varphi_X(T) - \delta^\varphi_X(T) = (\varphi'(T) - \varphi(T)) - (\varphi'(X) - \varphi(X)).$$

Proof. Let $T, X \in T$ (arbitrarily), w.l.o.g. $X$ is reachable from $T$ in $N$ (otherwise, $\delta_X^\varphi(T)$ and $\delta_X^\varphi(T)$ are both $+\infty$). Consider any path $P_{T,X}$ from $T$ to $X$ in $N^\varphi$, i.e., for some $k \geq 0$:

$$P_{T,X} \triangleq (T = T_0, T_1, T_2, \ldots, T_k = X),$$

having total weight $w_{P_{T,X}}^\varphi \triangleq \sum_{i=0}^{k-1} w_{T_i,T_{i+1}}^\varphi$ in $N^\varphi$.

Then, the following holds by telescoping:

$$w_{P_{T,X}}^\varphi = (w_{T_0,T_1} - \varphi(T_1)) + (w_{T_1,T_2} - \varphi(T_2)) + \varphi(T_1) + \ldots + (w_{T_{k-1},T_k} - \varphi(T_k)) + \varphi(T_{k-1})$$

$$= \varphi(T_0) - \varphi(T_k) + \sum_{i=0}^{k-1} w_{T_i,T_{i+1}} = \varphi(T) - \varphi(X) + w_{P_{T,X}}^\varphi.$$ (1)

Thus, provided $\delta_X(T)$ is the shortest path distance from $T$ to $X$ in the original network $N$,
On RDTPs: Faster Algorithms and Tractability Frontier

we have:

\[ \delta_X^\ell(T) = \min \left\{ w_{PT,X}^\ell \mid \text{path from } T \text{ to } X \text{ in } \mathcal{N} \right\} \]

(by def. of \( \delta_X^\ell \))

\[ = \min \left\{ \phi(T) - \phi(X) + w_{PT,X} \mid \text{path from } T \text{ to } X \text{ in } \mathcal{N} \right\} \]  

(by (1))

\[ = \phi(T) - \phi(X) + \min \left\{ w_{PT,X} \mid \text{path from } T \text{ to } X \text{ in } \mathcal{N} \right\} \]  

(\( \phi \) is constant here)

\[ = \phi(T) - \phi(X) + \delta_X(T). \]

(by def. of \( \delta_X \))

For the same reason, \( \delta_X^\ell(T) = \phi'(T) - \phi'(X) + \delta_X(T) \). Therefore,

\[ \delta_X^\ell(T) - \delta_X^\ell(T) = (\phi'(T) - \phi'(X) + \delta_X(T)) - (\phi(T) - \phi(X) + \delta_X(T)) \]

\[ = (\phi'(T) - \phi(T)) - (\phi'(X) - \phi(X)). \]

This concludes the proof. \( \square \)

Lemma 2. Let \( \mathcal{N} = (\mathcal{T}, \mathcal{C}_t) \) be any STN, and let \( \hat{\phi} \) be the least feasible schedule of \( \mathcal{N} \). Fix some \( X \in \mathcal{T} \) and some real value \( l_X \geq \hat{\phi}(X) \). Let \( \mathcal{N}' = (\mathcal{T}', \mathcal{C}'_t) \) be the auxiliary STN obtained by introducing a corresponding lower-bound \( t_1 \)-constraint over \( X \), i.e.,

\[ \mathcal{T}' \triangleq \mathcal{T} \cup \{ z \}, \quad \mathcal{C}'_t \triangleq \mathcal{C}_t \cup \{ (z - T \leq 0) \mid T \in \mathcal{T} \} \cup \{ (z \leq l_X) \}. \]

Let \( \mathcal{N}^{\hat{\phi}} \) be the STN reweighted according to the reduced-costs weight transformation \( w^{\hat{\phi}} \), and let \( \delta_X^\ell(T) \) be the length of the shortest path in \( \mathcal{N}^{\hat{\phi}} \) from (any) \( T \in \mathcal{T} \) to \( X \).

Then, for every \( T \in \mathcal{T} \), the least feasible schedule \( \phi' \) of \( \mathcal{N}' \) is given by:

\[ \phi'(T) = \hat{\phi}(T) + \max (0, l_X - \hat{\phi}(X) - \delta_X^\ell(T)). \]

**Proof.** Let w.l.o.g. \( \phi'(z) = 0 \). In order to become feasible for \( \mathcal{N}' \) we claim, for every \( T \in \mathcal{T} \), that the least feasible schedule \( \hat{\phi}(T) \) must be increased by at least \( \max (0, l_X - \hat{\phi}(X) - \delta_X^\ell(T)) \) time units (because of the lower-bound constraint \( (z - X \leq -l_X) \in \mathcal{C}'_t \)). 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 \ref{lemma:t1}\footnote{The anywhere-zero schedule \( \zeta \) is defined as, \( \zeta(T) = 0 \) for every \( T \in \mathcal{T} \).} (applied to \( \hat{\phi} \) and to the anywhere-zero[\ref{lemma:t1}] schedule), it holds \( \hat{\phi}(X) - \hat{\phi}(T) = \delta_X(T) - \delta_X^\ell(T) \). This can be seen as follows: if \( \hat{\phi}(T) \) is kept fixed, then \( \hat{\phi}(X) \) can be increased by at most \( \delta_X^\ell(T) \) time units without breaking the induced constraint \( (X - T \leq \delta_X(T)) \). Here, \( \hat{\phi}(X) \) must be increased by at least \( l_X - \hat{\phi}(X) \) time units in order to satisfy \( (z - X \leq -l_X) \in \mathcal{C}'_t \), so \( \hat{\phi}(T) \) must be increased by at least the amount said above.

Next, we claim this increase also preserves feasibility, i.e., it is the least feasible increase.

For ease of notation, let \( f(z) \triangleq 0 \) and \( f(T) \triangleq \hat{\phi}(T) + \max (0, l_X - \hat{\phi}(X) - \delta_X^\ell(T)) \forall T \in \mathcal{T} \).

In order to prove that \( f \) satifies all the constraints in \( \mathcal{C}'_t \), pick any \( (\mathcal{B} - \mathcal{A} \leq w_{\mathcal{A},\mathcal{B}}) \in \mathcal{C}'_t \).

By hypothesis, it holds:

\[ \hat{\phi}(\mathcal{B}) - \hat{\phi}(\mathcal{A}) \leq w_{\mathcal{A},\mathcal{B}}. \]

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

So, the following holds:

\[ f(\mathcal{B}) - f(\mathcal{A}) = \hat{\phi}(\mathcal{B}) - \hat{\phi}(\mathcal{A}) + \Delta_{\mathcal{A},\mathcal{B}}. \]

Then, either one of the following two cases holds:
If $l_X - \tilde{\varphi}(X) \leq \delta_X^N(B)$, then $f(B) = \varphi(B)$, so $\Delta_{A,B} \leq 0$. Therefore,

$$f(B) - f(A) \leq \varphi(B) - \varphi(A)$$  
(by (2))

$$\leq w_{A,B}.$$  
(by (1))

If $l_X - \tilde{\varphi}(X) > \delta_X^N(B)$, it is easy to check that $\Delta_{A,B} \leq \delta_X^N(A) - \delta_X^N(B)$.

By definition of $\delta_X^N$ and since $(B - A \leq w_{A,B}) \in C_{t_1}$, then $\delta_X^N(A) \leq \delta_X^N(B) + w_{A,B}$. Therefore,

$$f(B) - f(A) \leq \varphi(B) - \varphi(A) + \delta_X^N(A) - \delta_X^N(B)$$

$$\leq \varphi(B) - \varphi(A) + w_{A,B} = w_{A,B}.$$

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

Finally, clearly $f(X) = l_X$, so $(z - X \leq -l_X) \in C_{t_2}$ is also satisfied.

This proves $f$ is a feasible schedule of $N'$. All in, it is the least feasible, i.e., $f = \varphi$. ▶

With this two facts in mind, the description of $t_2\text{DTP}()$ can now proceed more smoothly.

Recall that, firstly, the STN $N'_{t_1}$ is checked. If $N'_{t_2}$ is already inconsistent, so it is $N_{t_2}$; otherwise, $\varphi_X$ is the least feasible schedule of $N_{t_1}$. So, $w_{X,N} \geq 0$ for every constraint in $C_{t_1}$.

Now, for each target node $X \in T$, the Dijkstra algorithm on input $(N_{t_2}, X)$ computes $\delta_X^{t_2}(T)$. The whole distance matrix $[\delta_X^{t_2}(T)]_{T \in T, X \in T}$ 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 $c_{t_2} \in C_{t_2}$, at increasing the candidate schedule $f$ by the least necessary amount satisfying both $c_{t_2}$ and the whole $C_{t_1}$.

So, let us initialize $f \leftarrow \varphi_X$. Then the following iterates.

While 3 some $X \in T$ and $c_X = \bigwedge_{i=1}^k (l_i \leq X \leq u_i) \in C_{t_2}$ s.t. $f(X)$ doesn’t satisfy $c_X$:

if $f(X) > u_k (= \max_i u_i)$, then $N'_{t_2}$ is inconsistent (see Theorem 4); otherwise, let $i^*$ be the smallest $i \in [1,k]$ such that $f(X) < l_i$. By Lemma 3 and given $f$, then $\delta_X^{t_2}$ is given by:

$$\delta_X^{t_2}(T) \leftarrow \delta_X^{t_2}(T) + (f(T) - \varphi_0(T)) - (f(X) - \varphi(X)), \forall T \in T.$$  
(rule-δ)

So, by Lemma 2 the following updating rule:

$$f(T) \leftarrow f(T) + \max\{0, l_{i^*} - f(X) - \delta_X^{t_2}(T)\}, \forall T \in T.$$  
(rule-f)

yields the least feasible schedule for the next auxiliary STN $N''_{t_2}$ obtained by adding the new lower-bound $t_1$-constraint $(z - X \leq -l_{i^*})$. At each iteration of the whole-loop $N''_{t_2}$ is enriched with an additional lower-bound $t_1$-constraint as above. So, $N''_{t_2}$ has $|T'| = |T| + 1$ time-points (z included) and at most $|C'_{t_1}| + |C_{t_2}|$ $t_1$-constraints (one $t_1$-constraint per $c_{t_2} \in C_{t_2}$ is enough, for each $c_{t_2}$ only its greatest lower-bound counts). If the while-loop completes without ever finding $N'_{t_2}$ to be inconsistent (because, eventually, $f(X) > u_k (= \max_i u_i)$ for some $X \in T$ at some point), then the last updating of $f$ yields the least feasible schedule of $N_{t_2}$ (as shown below in Theorem 5). This concludes the description of $t_2\text{DTP}()$.

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

▶ Theorem 5. $t_2\text{DTP}()$ is correct, i.e., on any input $t_2\text{DTN} N'_{t_2} = (T, C_{t_1} \cup C_{t_2})$, it returns a feasible schedule $\varphi : T \to \mathbb{R}$, if $N'_{t_2}$ is consistent; otherwise, it recognizes $N'_{t_2}$ as inconsistent.
**Proof.** Let \( \ell = 0, 1, 2, \ldots, \ell_h \) be all the iterations of the while-loop of \( t_2\text{DTP}() \), where \( \ell_h \) is assumed to be the last iteration where the updating rule-\( f \) is applied.

For every iteration \( \ell \in [1, \ell_h] \), the auxiliary STN \( N_{\ell} \) is formally defined as:

\[
N_{\ell} = (T \cup \{ z \}, C_{\ell} \), \]

where \( z \) is the zero time-point, and...

\[
C_{\ell} = C_{\ell} \cup \{ (z - T \leq \ell) \mid T \in T \} \cup \{ (z - X(\ell) \leq -I(\ell)) \mid 1 \leq \gamma \leq \ell \},
\]

where, for all \( \gamma \leq \ell \), \( X(\gamma) \) is the (unique) \( X \in T \) appearing in some \( t_2 \)-constraint that is considered at the while-loop’s \( \gamma \)-th iteration, and \( I(\ell) \) is its corresponding lower-bound.

Also, let \( f(\ell) \) be the candidate schedule as updated by rule-\( f \) during the \( \ell \)-th iteration.

By applying Lemmas 1 and 2 repeatedly, for each iteration \( \ell \), it holds that \( f(\ell) \) is the least feasible schedule of \( N_{\ell} \). This is the key invariant at the heart of \( t_2\text{DTP}() \).

Concerning actual correctness, firstly, assume that \( t_2\text{DTP}() \) recognizes \( N_{\ell} \) as inconsistent.

If \( N_{\ell} \) was already inconsistent (cfr. Theorem 1, so \( N_{\ell} \) is also inconsistent. Otherwise, the inconsistency of \( N_{\ell} \) 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 \( f(X) > u_k = (\max \{ u_l \} \) for some \( t_2 \)-constraint \( c_X = \sum_{i = 1}^n (l_i \leq X \leq u_i) \in C_{\ell} \). Indeed notice that, by (i), no possible feasible schedule \( g < f \) can be neglected (discarded) during the upward monotone (value-iteration rule-) updates of the schedules; and, by (ii), no possible schedule \( g \geq f \) can ever satisfy \( c_X \in C_2 \). So, \( N_{\ell} \) is really inconsistent.

Secondly, assume that \( N_{\ell} \) is recognized as consistent, by returning a schedule \( f^{(\ell_h)} \).

Since \( t_2\text{DTP}() \) can do only after the above while-loop completes, the exit condition of the latter ensures that \( f^{(\ell_h)} \) satisfies every constraint in \( C_{\ell} \). Moreover, the key invariant implies that \( f^{(\ell_h)} \) is the least feasible schedule of \( N_{\ell} \), so that \( f^{(\ell_h)} \) satisfies all of the \( t_1 \)-constraints in \( C_{\ell} \). These two combined, \( f^{(\ell_h)} \) is the least feasible schedule of \( N_{\ell} \). So, \( N_{\ell} \) is indeed consistent.

The next result asserts that \( t_2\text{DTP}() \) always halts in time polynomial in the input size.

**Theorem 6.** Suppose that \( t_2\text{DTP}() \) runs on input \( t_2\text{DTN} = (T, C_{t_1}, C_{t_2}) \).

Then, \( t_2\text{DTP}() \) halts in time \( O(|T| \cdot |C_{t_2}| + |C_{t_1}| \cdot |C_{t_1}| + |T| \cdot \log |T|) + |T| \cdot d_{C_{t_1}} \).

**Proof.** Solving the STP \( N_{\ell} = (T, C_{t_1}, C_{t_2}) \) with BF-VI takes \( O(|T| \cdot |C_{t_2}|) \) time (cfr. Theorem 1). Computing the shortest paths distance matrix \( \{ d_{X}^{(\ell)} (T) \}_{T \in T, X \in T} \) takes \( |C_{t_2}| \) calls to the Dijkstra algorithm (one per \( X \in T \) participating in some \( t_2 \)-constraint), so, \( O(|C_{t_2}| \cdot (|C_{t_1}| + |T| \cdot \log |T|) + |T| \cdot d_{C_{t_1}} \) total time. Checking the while-loop exit condition (i.e., whether there exists some unsatisfied \( c_X \in C_{t_2} \), can be done in \( O(|T| \cdot d_{C_{t_1}}) \) total time (because there are at most \( d_{C_{t_1}} \) iterations and each check can be done in \( O(|T|) \) time). At each iteration of the while-loop, applying rule-\( \delta \) and rule-\( f \) to all \( T \in T \) takes \( O(|T|) \) time per iteration, and we have at most \( d_{C_{t_1}} \) of them; so, notice that it takes only \( O(1) \) time per single application of the rules.

Therefore, the overall time complexity of \( t_2\text{DTP}() \) on any input \( N_{\ell} = (T, C_{t_1}, C_{t_2}) \) is:

\[
\text{Time}_{t_2\text{DTP}}(N_{\ell}) = O(|T| \cdot |C_{t_2}| + |C_{t_1}| \cdot (|C_{t_1}| + |T| \cdot \log |T|) + |T| \cdot d_{C_{t_1}}).
\]

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 \( t_2\text{DTP}() \) algorithm in mind, let us now focus on solving RDTPs fastly. Given an input RDTP \( N = (T, C_{t_1}, C_{t_2}) \), we firstly solve the \( t_2\text{DTP}() \) algorithm in mind, let us now focus on solving RDTPs fastly.
with \( t_2 \text{DTP}() \) (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 \( t_3 \)-constraints by making one single reduction call to the 2-SAT problem (which can be solved in linear-time \( \Pi \)).

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 \text{true} iff \( d' \) is satisfied (and \( d'' \) can be anything), whereas \( x_c \) is \text{false} iff \( d' \) is unsatisfied and \( d'' \) is satisfied.

The 2-CNF formula \( \mathcal{C}_d \mathcal{N} \) 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 \( d \) of any other \( t_3 \)-constraint \( c \) \( \neq c \) 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 \( \hat{u} \) must become a strict lower-bound for the scheduling value of some \( \hat{X} \in \mathcal{T} \) that appears in some other \( t_3 \)-disjunct \( d' = (l \leq \hat{X} \leq \bar{u}) \). This is formalized in Definition 8 (below). If that is the case, a binary clause asserting the above implication is added to \( \mathcal{C}_d \mathcal{N} \). Let us formally describe the details of this construction.

\begin{definition}
Consider the \( t_2 \text{DTP} \mathcal{N} \{ d'_c \} \cup \{ \mathcal{C}_d \mathcal{N} \} \), in which \( d'_c \) is added to \( \mathcal{C}_{t_1} \), as a pair of \( t_1 \)-constraints, i.e.,

\[ \mathcal{N} \{ d'_c \}_{t_2} = \left( \mathcal{C}_{t_1} \cup \{(z - X_i \leq -l_i), (X_i - z \leq u_i)\} \right) \]

\[ \cup \{ z - T \leq 0 | T \in \mathcal{T} \} \cup \mathcal{C}_{t_2} \}
\end{definition}

If \( \mathcal{N} \{ d'_c \}_{t_2} \) is consistent, let \( \hat{\varphi} \) be its least feasible schedule; otherwise, add the unary clause \( \neg x_c \) to \( \mathcal{C}_d \mathcal{N} \). For each \( \hat{c} \neq c \in \mathcal{C}_{t_3} \), e.g., \( \hat{c} = (l_1 \leq X_1 \leq \bar{u}_1) \lor (l_2 \leq X_j \leq \bar{u}_2) \) \( \in \mathcal{C}_{t_3} \),

\begin{enumerate}
\item if \( \hat{\varphi} \{ d'_c \}_{X_1} > \bar{u}_1 \) then add the implication \( x_c \Rightarrow \neg x_{\hat{c}} \) (i.e., clause \( \neg x_c \lor \neg x_{\hat{c}} \)) to \( \mathcal{C}_d \mathcal{N} \);
\item if \( \hat{\varphi} \{ d'_c \}_{X_2} > \bar{u}_2 \) then add the implication \( x_c \Rightarrow x_{\hat{c}} \) (i.e., clause \( \neg x_{\hat{c}} \lor x_c \)) to \( \mathcal{C}_d \mathcal{N} \).
\end{enumerate}

2. Consider the \( t_2 \text{DTP} \mathcal{N} \{ d''_c \} \cup \{ \mathcal{C}_d \mathcal{N} \} \), in which \( d''_c \) is added to \( \mathcal{C}_{t_1} \) (similarly as above). If \( \mathcal{N} \{ d''_c \}_{t_2} \) is consistent, let \( \hat{\varphi} \) be its least feasible schedule; otherwise, add the unary clause \( x_c \) to \( \mathcal{C}_d \mathcal{N} \). Again, for each \( \hat{c} \neq c \in \mathcal{N} \), e.g., \( \hat{c} = (l_1 \leq X_1 \leq \bar{u}_1) \lor (l_2 \leq X_j \leq \bar{u}_2) \):

\begin{enumerate}
\item if \( \hat{\varphi} \{ d''_c \}_{X_1} > \bar{u}_1 \) then add the implication \( \neg x_c \Rightarrow \neg x_{\hat{c}} \) (i.e., clause \( x_c \lor \neg x_{\hat{c}} \)) to \( \mathcal{C}_d \mathcal{N} \);
\item if \( \hat{\varphi} \{ d''_c \}_{X_2} > \bar{u}_2 \) then add the implication \( \neg x_c \Rightarrow x_{\hat{c}} \) (i.e., the clause \( x_c \lor x_{\hat{c}} \)) instead.
\end{enumerate}

So, if the 2-SAT problem instance \( \mathcal{C}_d \mathcal{N} \) is unsatisfiable, the input RDTP \( \mathcal{N} \) is inconsistent. Otherwise, for each \( c \neq d' \lor d'' \in \mathcal{C}_{t_3} \), we get at least one feasible \( t_2 \text{DTP} \) either \( \mathcal{N} \{ d'_c \}_{t_2} \) or \( \mathcal{N} \{ d''_c \}_{t_2} \), which is related to the first disjunct \( \{ d' \} \cup \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2} \) or \( \{ d'' \} \cup \mathcal{C}_{t_1} \cup \mathcal{C}_{t_2} \) (according to whether \( x_c \) is \text{true} or not in the satisfying assignment of \( \mathcal{C}_d \mathcal{N} \)). Then we compute the pointwise-maximum schedule taken among all of those. Formally,

\begin{definition}
Let \( \phi : \{ x_c \}_{c \in \mathcal{C}_{t_3}} \rightarrow \{ \text{true, false} \} \) be any satisfying assignment of \( \mathcal{C}_d \mathcal{N} \).

For each \( c \neq d' \lor d'' \in \mathcal{C}_{t_3} \), let us define:

\[ d^c_0 \triangleq \left\{ \begin{array}{ll}
\{ d'_c \}, & \text{if } \phi(x_c) = \text{true};
\{ d''_c \}, & \text{otherwise.}
\end{array} \right. \]

then, \( \hat{\varphi}(T) \triangleq \max_{c \in \mathcal{C}_{t_3}} \hat{\varphi}(d^c_0)(T), \forall T \in \mathcal{T} \),

where \( \hat{\varphi}(d^c_0) \) denotes the least feasible schedule of the consistent \( t_2 \text{DTP} \mathcal{N} \{ d^c_0 \}_{t_2} \).

\footnote{Here, recall the rule of material implication \( p \rightarrow q \iff \neg p \lor q \).}
The above pointwise-maximum schedule \( \hat{\phi}_N \) turns out to be feasible for the input RDTP \( N' \), as we show next. It is assumed we are given an RDTP \( N' \) for which \( \text{CL}_{N'} \) is satisfiable.

**Proposition 2.** Given \( N' \) as above, the schedule \( \hat{\phi}_N \) satisfies every \( c \in C_{t_1} \).

**Proof.** Let \( c_{t_1} = (Y - X \leq w) \in C_{t_1} \) be any \( t_1 \)-constraint, some \( X, Y \in T \) and \( w \in \mathbb{R} \). Pick any \( c'Y \in \arg \max_{c \in C_{t_1}} \hat{\phi}_N[\hat{d}^0_c](Y) \). Clearly, \( \max_{c \in C_{t_1}} \hat{\phi}_N[\hat{d}^0_c](X) \geq \hat{\phi}_N[\hat{d}^0_{c'}](X) \). Therefore:

\[
\hat{\phi}_N(Y) - \hat{\phi}_N(X) = \max_{c \in C_{t_1}} \hat{\phi}_N[\hat{d}^0_c](Y) - \max_{c \in C_{t_1}} \hat{\phi}_N[\hat{d}^0_c](X) \\
\leq \hat{\phi}_N[\hat{d}^0_{c'}](Y) - \hat{\phi}_N[\hat{d}^0_{c'}](X) \leq w,
\]

where the very last inequality holds because \( \hat{\phi}_N[\hat{d}^0_{c'}] \) is feasible for \( (T, C_{t_1}) \). So, \( \hat{\phi}_N \) satisfies \( c_{t_1} \).

**Proposition 3.** Given \( N' \) as above, the schedule \( \hat{\phi}_N \) satisfies every \( c \in C_{t_2} \).

**Proof.** Let \( c_{t_2} = \bigvee_{i=1}^k (l_i \leq X \leq u_i) \in C_{t_2} \) be any \( t_2 \)-constraint, some \( X \in T \), \( l_i, u_i \in \mathbb{R} \). Pick any \( c'X \in \arg \max_{c \in C_{t_2}} \hat{\phi}_N[\hat{d}^0_c](X) \). By definition \( \hat{\phi}_N[\hat{d}^0_{c'}] \) is a feasible schedule of \( N'[\hat{d}^0_{c'}] \) for \( (T, C_{t_2}) \) too. Therefore,

\[
l_q \leq \hat{\phi}_N[\hat{d}^0_{c'}](X) \leq u_q, \text{ for some } q \in \{1, \ldots, k\}.
\]

Since, \( \hat{\phi}_N(X) = \hat{\phi}_N[\hat{d}^0_{c'}](X) \), then \( \hat{\phi}_N(X) \in [l_q, u_q] \) for the same \( q \). So, \( \hat{\phi}_N \) satisfies \( c_{t_2} \).

**Proposition 4.** Given \( N' \) as above, the schedule \( \hat{\phi}_N \) satisfies every \( c \in C_{t_3} \).

**Proof.** Let \( c_{t_3} = (l_1 \leq X \leq u_1) \lor (l_2 \leq Y \leq u_2) \in C_{t_3} \) be any \( t_3 \)-constraint, some \( X, Y \in T \), \( X < Y \) and \( l_1, l_2, u_1, u_2 \in \mathbb{R} \). Assume w.l.o.g. \( \phi(x_{c_3}) = \text{true} \). Then, \( l_1 \leq \hat{\phi}_N[\hat{d}^0_{c_3}](X) \leq u_1 \).

If \( c_{t_3} \in \arg \max_{c \in C_{t_3}} \hat{\phi}_N[\hat{d}^0_c](X) \), then \( \hat{\phi}_N(X) = \hat{\phi}_N[\hat{d}^0_{c_3}](X) \in [l_1, u_1] \); so, \( \hat{\phi}_N \) would satisfy \( c_{t_3} \). Otherwise, \( c_{t_3} \not\in \arg \max_{c \in C_{t_3}} \hat{\phi}_N[\hat{d}^0_c](X) \), and assume \( \hat{\phi}_N(X) \not\in [l_1, u_1] \) towards a contradiction. Pick any \( c'X \in \arg \max_{c \in C_{t_3}} \hat{\phi}_N[\hat{d}^0_c](X) \). All these hypotheses combined:

\[
\hat{\phi}_N(X) = \hat{\phi}_N[\hat{d}^0_{c'}](X) > u_1.
\]

Therefore, \( \phi \) must satisfy either \( p \Rightarrow \neg x_{c_3} \) or \( \neg p \Rightarrow \neg x_{c_3} \), for some boolean variable \( p \) (where the actual case depends on the actual value of \( d^0_{c_3} \)). Since \( \phi \) satisfies either \( p \) or \( \neg p \), then \( \phi \) must satisfy \( \neg x_{c_3} \); i.e., \( \phi(x_{c_3}) = \text{false} \). This is absurd, as we assumed \( \phi(x_{c_3}) = \text{true} \).

The proof of the other case, in which \( \phi(x_{c_3}) = \text{false} \) is initially assumed, is symmetric. So, \( \hat{\phi}_N(X) \) satisfies \( c_{t_3} \).

Let us mention that our algorithm is called \( \text{RDTP}(\cdot) \), basically, it aims at computing \( \hat{\phi}_N \) as above; if it fails in that (either because \( N'_{t_2} \) is already inconsistent or \( \text{CL}_{N'} \) is unsatisfiable), it recognizes the input RDTP \( N' \) as inconsistent. Now, we can prove this is correct and fast.

**Theorem 7.** \( \text{RDTP}(\cdot) \) is correct, i.e., on any \( \text{RDTP} \) \( N = (T, C_{t_1} \cup C_{t_2} \cup C_{t_3}) \), it returns a feasible schedule \( \hat{\phi}_N : T \rightarrow \mathbb{R} \), if \( N' \) is consistent; otherwise, \( N' \) is recognized as inconsistent.

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

On the other side, by Propositions 2, 3 and 4 schedule \( \hat{\phi}_N \) is really feasible for \( N' \).
The next result asserts that the halting time is strongly polynomial in the input size.

\begin{theorem}
Let \( \mathcal{RDTP}(\cdot) \) run on any input \( \mathcal{RDTP} \mathcal{N} = (\mathcal{T}, \mathcal{C}_t \cup \mathcal{C}_s \cup \mathcal{C}_3) \).
Its always halts within time \( O(|\mathcal{T}| \cdot |\mathcal{C}_t| + |\mathcal{C}_s| \cdot (|\mathcal{C}_t| + |\mathcal{T}| \cdot \log |\mathcal{T}| + |\mathcal{T}| \cdot d_{\mathcal{C}_s}^2 \cdot |\mathcal{C}_3|^2) \).
\end{theorem}

\begin{proof}
By Theorem 6, \((\mathcal{T}, \mathcal{C}_t \cup \mathcal{C}_s)\) takes \( O(|\mathcal{T}| \cdot |\mathcal{C}_t| + |\mathcal{C}_s| \cdot (|\mathcal{C}_t| + |\mathcal{T}| \cdot \log |\mathcal{T}| + |\mathcal{T}| \cdot d_{\mathcal{C}_s}^2) \) time to be checked. Using that solution as an initial candidate, solving the two \( \mathcal{t}_2 \mathcal{DTPs} \) \( \mathcal{N}[d_{\mathcal{C}_s}^2]_{\mathcal{C}_3} \) and \( \mathcal{N}[d_{\mathcal{C}_s}^2]_{\mathcal{C}_3} \), for each \( c \in \mathcal{C}_3 \) where \( c = d_c^3 \lor d_c^3 \), it takes \( O(|\mathcal{T}| \cdot |\mathcal{C}_t| + |\mathcal{C}_s| \cdot (|\mathcal{C}_t| + |\mathcal{T}| \cdot \log |\mathcal{T}| + |\mathcal{T}| \cdot d_{\mathcal{C}_s}^2 \cdot |\mathcal{C}_3|) \) total time. Next, for each \( c, \tilde{c} \in \mathcal{C}_3 \) such that \( \tilde{c} \neq c \), eventually adding the corresponding clauses to \( \mathcal{C}_{\mathcal{T}_N} \) takes \( O(1) \) time per clause; so, \( \mathcal{C}_{\mathcal{T}_N} \) is built in total time \( O(|\mathcal{C}_3|^2) \). Since \( |\mathcal{C}_{\mathcal{T}_N}| = O(|\mathcal{C}_3|^2) \), solving the 2-SAT problem on input \( \mathcal{C}_{\mathcal{T}_N} \) takes time \( O(|\mathcal{C}_3|^2) \) (e.g., with the algorithm of \([\text{I}]\)). Finally, computing \( d_c^3 \) and \( \tilde{\phi}_N \) takes \( O(|\mathcal{T}| \cdot |\mathcal{C}_3|) \) time. All in, the above mentioned time complexity of \( \mathcal{RDTP}(\cdot) \) follows.
\end{proof}

\section{NP-completeness of Multi-Tail & Multi-Head \( \mathcal{t}_3 \mathcal{HyTPs} \)}

This section enquires the tractability frontier of RDTPs by considering HyTNs \([3]\), where the constraints can be modeled only by multi-tail hyperarcs instead. Let us now focus on \( \mathcal{t}_3 \mathcal{HyTP} \).

\begin{theorem}
TAIL-\( \mathcal{t}_3 \mathcal{HyTP} \) is NP-complete in a strong sense, i.e., even if the input \((\mathcal{T}, \mathcal{A} \cup \mathcal{C}_3)\) are restricted to satisfy \( w_A(\cdot) \in \{-1, 0, 1\}, |\mathcal{T}_A| \leq 2 \) for every \( A \in \mathcal{A} \), and every \( \mathcal{t}_3 \)-constraint \((l_i \leq X \leq u_i) \lor (l_j \leq Y \leq u_j) \in \mathcal{C}_3 \) has all zero-valued lower/upper-bounds.
\end{theorem}

\begin{proof}
We claim that if \( \mathcal{H} = (\mathcal{T}, \mathcal{A} \cup \mathcal{C}_3) \) is an integer-weighted and consistent multi-tail \( \mathcal{t}_3 \mathcal{HyTN} \), it admits an integer-valued feasible schedule \( s : \mathcal{T} \rightarrow \{-T, \ldots, T\} \) where \( T = \sum_{A \in \mathcal{A}, v \in \mathcal{V}} -\lceil w_A(v) \rceil + \sum_{c \in \mathcal{C}_3} c = (l_i \leq X \leq u_i) \lor (l_j \leq Y \leq u_j) \). Indeed let \( s \) be a feasible schedule (integer-valued or not) of \( \mathcal{H} \), and consider the projection \( \mathcal{H}^* = (\mathcal{T}, \mathcal{A}^*) \), for \( \mathcal{A}^* \triangleq \mathcal{A} \cup \bigcup_{c \in \mathcal{C}_3} \mathcal{A}^*_c \), where for every \( c = ((l_i \leq X \leq u_i) \lor (l_j \leq Y \leq u_j) \in \mathcal{C}_3 \) we pick the following pair of \( \mathcal{t}_1 \)-constraints:
\[
\begin{align*}
\mathcal{A}^*_c \triangleq \{ & (Z - X \leq -l_i), (X - Z \leq u_i) \}, & \text{if } l_i \leq s(X) \leq u_i; \\
& (Z - Y \leq -l_j), (Y - Z \leq u_j) \}, & \text{otherwise}.
\end{align*}
\]

By construction of \( \mathcal{H}^* \), \( s \) is a feasible for \( \mathcal{HyTN} \mathcal{H}^* \). So, by Proposition 6, \( \mathcal{H}^* \) admits an integer-valued feasible schedule \( s^* \) bounded by \(-T\) and \(+T\) as above. By construction of \( \mathcal{H}^* \), \( s^* \) 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-\( \mathcal{t}_3 \mathcal{HyTP} \) 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:
\[
\varphi(x_1, \ldots, x_n) = \bigwedge_{i=1}^{m} (\alpha_i \lor \beta_i \lor \gamma_i), \]
where \( C_i = (\alpha_i \lor \beta_i \lor \gamma_i) \) is the \( i \)-th clause of \( \varphi \) and each \( \alpha_i, \beta_i, \gamma_i \in \{x_j, \overline{x}_j, 1 \leq j \leq n\} \) is either a positive or a negative literal.

We associate to \( \varphi \) a multi-tail \( \mathcal{t}_3 \mathcal{HyTP} \mathcal{H}_\varphi = (\mathcal{T}, \mathcal{A} \cup \mathcal{C}_3) \), where each boolean variable \( x_i \) occurring in \( \varphi \) 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 \(t_3\)-constraint \((0 \leq x_i \leq 0) \lor (0 \leq \overline{x}_i \leq 0)\) \(\in C_{ls}\). If \(H_\varphi\) is consistent, the multi-tail hyperarc and the \(t_3\)-constraint associated to \(x, \overline{x}\) assure that \(H_\varphi\) 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 \(H_\varphi\) contains also a time-point \(C_j\) for each clause \(C_j\) of \(\varphi\); 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 \(H_\varphi\) admits a feasible schedule \(s\), then \(s\) assigns scheduling time \(1\) at least to one of the time-points representing the literals connected with the multi-tail hyperarc.

Fig. 4 depicts the gadgets. A more formal definition of \(H_\varphi\) is given in Appendix A.

The reader can check that \(|T| = 1 + 2n + m = O(m + n)\), \(m_A = O(m + n)\), \(|C_{ls}| = O(n)\); therefore, the transformation is linearly bounded.

We next show that \(\varphi\) is satisfiable if and only if \(H_\varphi\) is consistent.

Any truth assignment \(\nu : \{x_1, \ldots, x_n\} \rightarrow \{\text{true, false}\}\) satisfying \(\varphi\) can be translated into a feasible schedule \(s : T \rightarrow \mathbb{Z}\) of \(H_\varphi\) 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 \text{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, \(H_\varphi\) is consistent.

Vice versa, assume that \(H_\varphi\) is consistent. Then, it admits an integer-valued feasible schedule \(s\) (as we mentioned above). After the translation \(s(\nu) \triangleq s(\nu) - s(z)\), 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 \(\nu : \{x_1, \ldots, x_n\} \rightarrow \{\text{true, false}\}\) defined by \(\nu(x_i) = \text{true if } s(x_i) = 1\) (and \(s(\overline{x}_i) = 0\)); \(\nu(x_i) = \text{false if } s(x_i) = 0\) (and \(s(\overline{x}_i) = 1\)) for every \(i = 1, \ldots, n\). So, \(\varphi\) is satisfiable.

To conclude, we observe that any hyperarc \(A \in \mathcal{A}\) of \(H_\varphi\) has weights \(w_A(\cdot) \in \{-1, 0, 1\}\), size \(|A| \leq 3\), and any \(t_3\)-constraint \(c = (l_i \leq X \leq u_i) \lor (l_j \leq Y \leq u_j) \in \mathcal{C}_{ls}\) 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 \(A \in \mathcal{A}\).

In order to prove that head-\(t_3\)-HyTP is also NP-complete, we could proceed with an argument similar to that of Theorem 9. However, we also observe that the same result follows.
as an immediate corollary of the following inter-reducibility between the two models.

**Definition 10.** 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 \{\(t_2, t_3\)\} disjunctive temporal constraints.

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

**Proposition 5.** 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 \(t_3\)HyTPs are inter-reducible by such reductions. (The proof is in Appendix A)

Therefore, by Proposition 5 it follows that \textsc{head-}\(t_3\)HyTP is also strongly NP-complete.

### 6 Pseudo-Polynomial Time Algorithm for \(t_2\)HyTPs

We end by studying multi-tail and multi-head \(t_2\)HyTNs (i.e., temporal networks in which the temporal constraints can be only \(t_2\) disjunctive temporal constraints and either only multi-tail or multi-head hyperarcs). It turns out that checking the corresponding temporal problems, \textsc{tail-}\(t_2\)HyTP and \textsc{head-}\(t_2\)HyTP, lies in NP \& co-NP and admits pseudo-polynomial time algorithms. By Proposition 5 it is sufficient to focus on multi-head \(t_2\)HyTPs only. The corresponding pseudo-polynomial time algorithm is named \(t_2\)HyTP(), and described below – notice that it generalizes \(t_2\)DTP(). Given any integer-weighted multi-head \(t_2\)HyTPs \(\mathcal{H}_{t_2} = (\mathcal{T}, \mathcal{A} \cup \mathcal{C}_{t_2})\) in input, we firstly solve the HyTP \(H = (\mathcal{T}, \mathcal{A})\) with the VI algorithm of Theorem 3. If \(H\) is recognized as inconsistent, the algorithm halts. Otherwise, let \(\varphi\) be the least feasible schedule of \(H\). Then proceed as follows:

While \(\exists\) some \(X \in \mathcal{T}\) and \(c_X = \bigvee_{i=1}^k (l_i \leq X \leq u_i) \in \mathcal{C}_{t_2}\) s.t. \(\varphi(X)\) doesn’t satisfy \(c_X\):

If \(\varphi(X) > u_k (= \max_k u_i)\), then \(\mathcal{H}_{t_2}\) 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 3 is invoked on input \((\mathcal{H}, \varphi)\), so, then, \(\varphi\) 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 \(\mathcal{H}_{t_2}\) as inconsistent, \(\varphi\) is returned. The correctness and the time complexity are asserted below. (The proof is in Appendix A)

**Theorem 10.** \(t_2\)HyTP() is correct, i.e., running on any integer-weighted multi-head \(t_2\)HyTP \(\mathcal{H}_{t_2} = (\mathcal{T}, \mathcal{A} \cup \mathcal{C}_{t_2})\), an integer-valued feasible schedule \(\varphi : \mathcal{T} \rightarrow \mathbb{Z}\) 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.,

\[
\text{Time}_{t_2\text{-HyTP}}(\mathcal{H}_{t_2}) = O\left(|\mathcal{T}| \cdot |\mathcal{A}| \cdot m_{\mathcal{A}} \cdot |W\mathcal{A},\mathcal{C}_{t_2}|\right),
\]

where \(W\mathcal{A},\mathcal{C}_{t_2} \triangleq \max_{A \in \mathcal{A}} \max_{h \in \mathcal{A}} |w_A(h)| \cdot \max_{\forall 1 \leq i \leq |\mathcal{T}| \leq u_i \in \mathcal{C}_{t_2}} l_j\).

Finally, since \(t_2\)HyTP() is correct, it is possible to establish the following complexity result.

**Theorem 11.** \(\textsc{tail-}t_2\)HyTP, \(\textsc{head-}t_3\)HyTP \(\in\) NP\&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/extension 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:https://doi.org/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*, ECSQARU’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. Schwab and L. Vila. Temporal constraints: A survey. *Constraints*, 3(2):129–149, 1998.
16. Costas 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 5. 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 $t_2$ and $t_3$-constraints are also reversed.

Given a multi-tail RHyTN $H = (\mathcal{T}, \mathcal{A} \cup C_{t_2} \cup C_{t_3})$, we associate to $H$ a multi-head RHyTN $H' = (\mathcal{T}, \mathcal{A}' \cup C'_{t_2} \cup C'_{t_3})$ by reversing all multi-tail hyperarcs, all $t_2$ and $t_3$-constraints. Formally,

$$A' \triangleq \{(v, S, w) \mid (S, v, w) \in A\}, \quad C'_{t_2} \triangleq \left\{ \bigvee_{i=1}^{k} (-u_i \leq X \leq -l_i) \mid \bigvee_{i=1}^{k} (l_i \leq X \leq u_i) \in C_{t_2}\right\},$$

$$C'_{t_3} \triangleq \left\{ ((-u_1 \leq X \leq -l_1) \lor (-u_2 \leq Y \leq -l_2)) \mid ((l_1 \leq X \leq u_1) \lor (l_2 \leq Y \leq u_2)) \in C_{t_3}\right\}.$$

We claim that $H$ is consistent if and only if $H'$ is consistent. To prove it, we note that each schedule $s$ for $H$ can be associated, with a flip of the time direction, to the schedule $s' \triangleq -s$. Then, it holds that $s$ is feasible for $H$ if and only if $s'$ is feasible for $H'$. Indeed, $s$ satisfies the constraint represented by an hyperarc $A = (T_A, h_A, w_A) \in A$, 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 $t_2$-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 $t_3$-constraint $((l_1 \leq X \leq u_1) \lor (l_2 \leq Y \leq u_2))$ 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 $H_\varphi$ in the proof of Theorem 9. More formally, $H_\varphi = (\mathcal{T}, \mathcal{A} \cup C_{t_3})$ is:

$\mathcal{T} = \{z\} \cup \{x_i \mid 1 \leq i \leq n\} \cup \{\overline{x}_i \mid 1 \leq i \leq n\} \cup \{C_j \mid 1 \leq j \leq m\};$

$\mathcal{A} = \bigcup_{i=1}^{n} \text{Var}_i \cup \bigcup_{j=1}^{m} \text{Cl}_{C_j}$, where:

$\text{Var}_i = \left\{ (z, x_i, 1), (x_i, z, 0), (z, \overline{x}_i, 1), (\overline{x}_i, z, 0), \right\}$

$\text{Cl}_{C_j} = \left\{ (z, C_j, 1), (C_j, z, -1), \right\}$

$\text{Var}'_{t_3} = \left\{ ((0 \leq x_i \leq 0) \lor (0 \leq \overline{x}_i \leq 0)) \right\}$

This completes the variable gadget of $x_i$ as depicted in Fig. 4(left);
Proof of head-$t_2$HyTP, tail-$t_2$HyTP $\in \text{NP}$. We claim that $\mathcal{H} = (\mathcal{T}, \mathcal{A} \cup \mathcal{C}_{t_2})$ is an integer-weighted and consistent multi-tail $t_2$HyTN, it admits an integer-valued feasible schedule $s : \mathcal{T} \to \{-T, \ldots, T\}$ where $T = \sum_{A \in \mathcal{A}, v \in V} |w_A(v)| + \sum_{c \in \mathcal{C}_{t_2}, c = \bigvee_{i=1}^k (l_i, X, u_i)} (|l_i| + |u_i|)$. Indeed, let $s$ be a feasible schedule (integer-valued or not) of $\mathcal{H}$, and consider the projection $\text{HyTN}$:

$$\mathcal{H}^s \triangleq (\mathcal{T} \cup \{z\}, \mathcal{A}'),$$

$$\mathcal{A}' \triangleq \mathcal{A} \cup \{(z - T \leq 0) \mid T \in \mathcal{T}\} \cup \bigcup_{c \in \mathcal{C}_{t_2}} \mathcal{A}^s_c,$$

where for every $c = \bigvee_{i=1}^k (l_i, X, u_i) \in \mathcal{C}_{t_2}$ this pair of $t_1$-constraints is taken:

$$\mathcal{A}^s_c \triangleq \{(z - X \leq -l_i), (X - z \leq u_i) \mid \text{for the smallest } i \in \{1, \ldots, k\} \text{ s.t. } l_i \leq s(X) \leq u_i\}.$$

By construction of $\mathcal{H}^s$, $s$ is a feasible for HyTN $\mathcal{H}^s$. So, by Proposition 4 $\mathcal{H}^s$ 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 $\mathcal{A} \cup \mathcal{C}_{t_2}$; hence, head-$t_2$HyTP is in NP. Thus, by Proposition 5 tail-$t_2$HyTP $\in \text{NP}$. ▶

Definition 11 (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 $S \neq \emptyset$;
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 $v_1, v_{k+1}, \ldots, v_p$, where $v_{p+1} = v_1$ is the only repeated node in the sequence. A cycle $(S, \mathcal{C}_0)$ is negative if for any finite cyclic sequence $v_1, v_2, \ldots, v_p$, it holds that $\sum_{t=1}^{p-1} a(v_t)(v_{t+1}) < 0$, where $a(v)$ denotes the unique arc $A \in \mathcal{C}_0$ with $t_A = v$ as required in previous item 2.

Definition 12 (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 $\varphi_{\text{cert}} \triangleq \{\varphi, \mathcal{F}\}$, where $\varphi : \mathcal{T} \to \mathbb{Z}$ is a feasible schedule of $\mathcal{H}$, and $\mathcal{F} \triangleq \{\mathcal{C}_X | X \tau\}$ is a family of hypercycles of $\mathcal{H}$ (which works as a certificate of minimality for $\varphi$, 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 $t_1$-constraint requiring $X$ to be scheduled strictly before time $\varphi(X)$,

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

$\varphi_{\text{cert}}$ can be verified in strongly polynomial time, because negative hypercycles can be checked so (as shown e.g., in Lemma 3 of [3]) and feasibility of $\varphi$ can be checked by inspection. The soundness of CLFSs follows from the proof argument of Proposition 1 (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\textsuperscript{11} and\textsuperscript{12} in mind, we can proceed with the following proof.

**Proof of head-\texttt{t}_2\texttt{HyTP}, tail-\texttt{t}_2\texttt{HyTP} ∈ co-NP.** To show head-\texttt{t}_2\texttt{HyTP} ∈ 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 \texttt{t}_2\texttt{HyTP}(\cdot), assuming the input instance \(\mathcal{H} = (\mathcal{T}, \mathcal{A} \cup \mathcal{C}_x)\) is inconsistent. If the HyTN \(\mathcal{H}_0 \equiv (\mathcal{T}, \mathcal{A})\) is already inconsistent, then it admits a negative hyercycle \(\mathcal{C}_0\) (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 \(\varphi_0\) be a CLFS of \(\mathcal{H}_0\), then there must exist some \(X_0 \in \mathcal{T}\) and \(c_{X_0} = \bigvee_{i=1}^k (l_i \leq X_0 \leq u_i) \in \mathcal{C}_x\) s.t. \(\varphi_0(X_0)\) doesn’t satisfy \(c_{X_0}\). If \(\varphi_0(X_0) > u_k(= \max_i u_i)\), then \((\varphi_0, c_{X_0})\) 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 \(t_1\)-constraint requiring \(X_0\) to be scheduled at or after time \(l_{i_0^*}\):

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

Then, let \(\varphi_1\) be a CLFS of \(\mathcal{H}_1\) (notice \(\varphi_1\) exists because \(\mathcal{H}_0\) was assumed to be feasible). Again, there must exist some \(X_1 \in \mathcal{T}\) and \(c_{X_1} = \bigvee_{i=1}^k (l_i \leq X_1 \leq u_i) \in \mathcal{C}_x\) 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 \(t_1\)-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):

\[
\left((\varphi_0, c_{X_0}), (\varphi_1, c_{X_1}), \ldots, (\varphi_N, c_{X_N})\right),
\]

where, notice, it’s length is at most \(N ≤ d_{\mathcal{C}_x}\) (because, for each iteration of the construction, one disjunct of some \(t_2\)-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\textsuperscript{10} implies the correctness of these certificates. This proves that head-\texttt{t}_2\texttt{HyTP} ∈ co-NP. Thus, by Proposition\textsuperscript{5} tail-\texttt{t}_2\texttt{HyTP} ∈ co-NP too. ▲

**Proof of Theorem\textsuperscript{10}** The correctness argument is similar to that for proving correctness of \texttt{t}_2\texttt{DTP}(\cdot), the details are simpler in this case because the only algorithm that is used to update the schedule \(\varphi\) monotonically is the VI algorithm of Theorem\textsuperscript{3} (instead of Bellman-Ford and multiple calls to Dijkstra as it was for \texttt{t}_2\texttt{DTP}(\cdot)); indeed, the VI algorithm of Theorem\textsuperscript{3} 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 \texttt{t}_2\texttt{HyTP}(\cdot) is a direct consequence of the complexity of the VI algorithm of Theorem\textsuperscript{3} where the maximal weight measure \(W\) is increased to \(W_{A, \mathcal{C}_x}\) (as defined above) in order to take into account the lower-bound constraints (i.e., those of type \((z - X ≤ -l_i)\) 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 \(l_\star\) to satisfy the last \((z - X ≤ -l_\star)\)). Finally, checking the while-loop’s condition takes time \(O(d_{\mathcal{C}_x} \cdot |\mathcal{T}|)\) total time, and since \(d_{\mathcal{C}_x} ≤ |\mathcal{T}| \cdot W_{A, \mathcal{C}_x}\), then \(O(d_{\mathcal{C}_x} \cdot |\mathcal{T}|) = O(|\mathcal{T}|^2 \cdot W_{A, \mathcal{C}_x})\) (which, notice, it is not a bottleneck asymptotically). So, the Time\texttt{t}_2\texttt{HyTP}(\cdot) bound holds. ▲