In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-2002).*

Algorithms for a Temporal Decoupling Problem in Multi-Agent Planning

Luke Hunsberger
Department of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138
luke@eecs.harvard.edu

Abstract

The Temporal Decoupling Problem (TDP) arises when a group of agents collaborating on a set of temporally-dependent tasks seek to coordinate their execution of those tasks by applying additional temporal constraints sufficient to ensure that agents working on different tasks may operate independently. This paper: (1) formally defines the TDP, (2) presents theorems that give necessary and sufficient conditions for solutions to the TDP, (3) presents a family of sound and complete algorithms for solving the TDP, and (4) compares the performance of several variations of the basic algorithm. Although this work was motivated by a problem in collaborative multi-agent planning, it represents a contribution to the theory of Simple Temporal Networks that is independent of the motivating application.

Introduction

In prior work on collaborative, multi-agent systems, Hunsberger and Grosz (2000) presented a group decision-making mechanism based on a combinatorial auction in which agents bid on sets of tasks in a proposed group activity. That work focused on the *winner-determination* problem in which the auctioneer seeks to determine whether there is a consistent set of bids covering all the tasks in the proposed activity. Later work (Hunsberger 2002) focused on the *bidding* phase in which bidders protect their private schedules of pre-existing commitments by including temporal constraints in their bids. This paper addresses a third problem, one that agents face after the awarding of bids, namely, how agents can ensure that the network of temporal constraints among their tasks will *remain* consistent until all of the tasks have been completed.

The following scenario illustrates some of the issues involved. Bill and Sue have committed to doing tasks β_1 and β_2 subject to the temporal constraints

$$3{:}00 \;\leq\; \beta_1 \;\leq\; \beta_2 \;\leq\; 5{:}00,$$

with Bill doing β_1 and Sue doing β_2. For simplicity, assume that these tasks have zero duration. The above constraints are satisfied by infinitely many pairs of execution times for β_1 and β_2; however, should Bill decide to execute β_1 at 4:15 (which is consistent with the above constraints) while Sue independently decides to execute β_2 at 3:45 (also consistent with the original constraints), they will end up violating the constraint $\beta_1 \leq \beta_2$. To avoid doing so, they could agree to one of the following:

- Add further constraints (e.g., $\beta_1 \leq 4{:}25 \leq \beta_2$) that would effectively decouple their tasks; or
- Have Sue wait until Bill announces a fixed time for β_1 before she decides to fix a time for β_2.

The first approach imposes additional constraints on the constituent tasks, but ensures that Bill and Sue may henceforth operate independently (without any further communication). The second approach gives Bill greater flexibility in that he may operate independently, but makes Sue dependent on Bill and requires him to communicate his choice of execution time for β_1 to her. This paper focuses on generalizing the first approach: providing algorithms to generate additional constraints to temporally decouple the subproblems being worked on by different agents. We leave to future work finding algorithms that follow the second approach (i.e., asymmetrically distributing authority for adding new constraints).

Simple Temporal Networks

The algorithms in this paper manipulate Simple Temporal Networks (Dechter, Meiri, & Pearl 1991). In this section, we briefly review STNs, highlighting their relevant properties.

Definition 1 (Simple Temporal Network) *(Dechter, Meiri, & Pearl 1991) A* Simple Temporal Network $\mathcal{S}$ *is a pair* $(\mathcal{T}, \mathcal{C})$*, where* $\mathcal{T}$ *is a set* $\{t_0, t_1, \ldots, t_N\}$ *of time-point variables and* $\mathcal{C}$ *is a finite set of binary constraints on those variables, each constraint having the form* $t_j - t_i \leq \delta$ *for some real number* δ*. The "variable"* t_0 *represents an arbitrary, fixed reference point on the time-line. (In this paper, we fix* t_0 *to the value* 0 *and refer to it as the* zero time-point variable*, or* z*.)*

A solution *to the STN* $\mathcal{S}$ *is a set of variable assignments*

$$\{z = 0,\; t_1 = v_1,\; \ldots,\; t_N = v_N\}$$

satisfying all the constraints in $\mathcal{C}$*. An STN* $\mathcal{S}$ *that has at least one solution is called* consistent.

Constraints involving z are equivalent to unary constraints. For example:

$$L_i \leq t_i \quad\Leftrightarrow\quad 0 - t_i \leq -L_i \quad\Leftrightarrow\quad z - t_i \leq -L_i.$$

Definition 2 (Distance Graph) *(Dechter, Meiri, & Pearl 1991) The* distance graph *for an STN $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ is a weighted, directed graph $G_{\mathcal{S}} = (\mathcal{V}_{\mathcal{S}}, \mathcal{E}_{\mathcal{S}})$, whose vertices correspond to the time points of $\mathcal{S}$ and whose edges correspond to the temporal constraints of $\mathcal{S}$, as follows:*

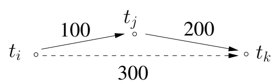
$$\mathcal{V}_{\mathcal{S}} = \mathcal{T} \quad \text{and} \quad \mathcal{E}_{\mathcal{S}} = \{(t_i, \delta, t_j) : (t_j - t_i \leq \delta) \in \mathcal{C}\}.$$

Thus, each constraint $t_j - t_i \leq \delta$ in $\mathcal{S}$ is represented in $G_{\mathcal{S}}$ by a directed edge from t_i to t_j with weight (or length) δ.

In an STN, the *explicit* constraints in $\mathcal{C}$ may give rise to additional *implicit* constraints. For example, the explicit constraints $t_j - t_i \leq 100$ and $t_k - t_j \leq 200$ combine to entail the implicit constraint $t_k - t_i \leq 300$, as follows:

$$t_k - t_i \;=\; (t_k - t_j) + (t_j - t_i) \;\leq\; 200 + 100 \;=\; 300.$$

In graphical terms, the edges from t_i to t_j to t_k form a *path* of length 300 from t_i to t_k, as illustrated below.



Theorem 1 *(Dechter, Meiri, & Pearl 1991) An STN $\mathcal{S}$ is consistent (i.e., has a solution) if and only if its distance graph $G_{\mathcal{S}}$ has no negative cycles (i.e., the path length around any loop is non-negative).*

Definition 3 (Temporal Distance) *(Dechter, Meiri, & Pearl 1991)*[1] *The* temporal distance *from t_i to t_j in an STN $\mathcal{S}$ is the length of the* shortest path *from t_i to t_j in the corresponding distance graph $G_{\mathcal{S}}$.*

Equivalently, the temporal distance from t_i to t_j specifies the *strongest implicit constraint* from t_i to t_j in $\mathcal{S}$ (where "implicit constraints" is taken to subsume "explicit constraints").

If no path exists from t_i to t_j in $G_{\mathcal{S}}$, then the temporal distance is infinite, representing that the temporal *difference*, $t_j - t_i$, is unconstrained. On the other hand, if there is a negative cycle in the distance graph, then some temporal distance is negative infinity, representing a constraint that cannot be satisfied.

Definition 4 (Distance Matrix) *(Dechter, Meiri, & Pearl 1991)*[2] *The* distance matrix *for an STN $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ is a matrix $\mathcal{D}$ such that:*

$$\mathcal{D}(i, j) = \text{ Temporal Distance from } t_i \text{ to } t_j \text{ in } \mathcal{S}.$$

Abusing notation slightly, we may write $\mathcal{D}(t_i, t_j)$ where t_i and t_j are time-point variables rather than indices.

Fact 2 *(Dechter, Meiri, & Pearl 1991) The distance matrix may be computed in $O(N^3)$ time using, for example, Floyd-Warshall's* all-pairs shortest-path *algorithm (Cormen, Leiserson, & Rivest 1990).*

Fact 3 *From Definitions 3 and 4, we get that the following inequalities hold for any time-points t_i and t_j in an STN:*

$$-\mathcal{D}(t_j, t_i) \;\leq\; t_j - t_i \;\leq\; \mathcal{D}(t_i, t_j).$$

Typically, adding a constraint to an STN causes some entries in the distance matrix to change. The following theorem specifies which constraints can be added without threatening the consistency of the STN.

Theorem 4 *(Dechter, Meiri, & Pearl 1991) For any time-points t_i and t_j in an STN $\mathcal{S}$, the new constraint $t_j - t_i \leq \delta$ will not threaten the consistency of $\mathcal{S}$ if and only if δ satisfies $-\mathcal{D}(t_j, t_i) \leq \delta$. Furthermore, the consistent STN $\mathcal{S}$ has a solution in which $t_j - t_i = \sigma$ if and only if $\sigma \in [-\mathcal{D}(t_j, t_i), \mathcal{D}(t_i, t_j)]$.*

Corollary 5 *The quantity $\mathcal{D}(t_i, t_j) + \mathcal{D}(t_j, t_i)$, which specifies the length of the interval $[-\mathcal{D}(t_j, t_i), \mathcal{D}(t_i, t_j)]$, also specifies the maximum amount by which the strongest implicit constraint from t_i to t_j may be* tightened.

Fact 6 *Given Theorem 1, the following inequality holds for any time-points t_i and t_j in a consistent STN:*

$$\mathcal{D}(t_i, t_j) + \mathcal{D}(t_j, t_i) \;\geq\; 0.$$

Fact 6 says that the length of the shortest path from t_i to t_j and back to t_i is always non-negative. Corollary 5 and Fact 6 together motivate the following new definition.

Definition 5 (Flexibility) *Given time-points t_i and t_j in a consistent STN, the relative* flexibility *of t_i and t_j is the (non-negative) quantity:*

$$\mathit{Flex}(t_i, t_j) \;=\; \mathcal{D}(t_i, t_j) + \mathcal{D}(t_j, t_i).$$

Rigid Components. Adding a constraint $t_j - t_i \leq \delta$ in the extreme case where $\delta = -\mathcal{D}(t_j, t_i)$ (recall Theorem 4), causes the *updated* distance matrix entries to satisfy:

$$-\mathcal{D}(t_j, t_i) \;=\; t_j - t_i \;=\; \mathcal{D}(t_i, t_j).$$

In such a case, the temporal difference $t_j - t_i$ is fixed (equivalently, $\mathit{Flex}(t_i, t_j) = 0$), and t_i and t_j are said to be *rigidly connected* (Tsamardinos, Muscettola, & Morris 1998).

The following measure of rigidity will be used in the experimental evaluation section.[3]

Definition 6 (Rigidity) *The* relative rigidity *of the pair of time-points t_i and t_j in a consistent STN is the quantity:*

$$\mathit{Rig}(t_i, t_j) = \frac{1}{1 + \mathit{Flex}(t_i, t_j)} = \frac{1}{1 + \mathcal{D}(t_i, t_j) + \mathcal{D}(t_j, t_i)}.$$

The RMS rigidity *of a consistent STN $\mathcal{S}$ is the quantity:*

$$\mathit{Rig}(\mathcal{S}) \;=\; \sqrt{\frac{2}{N(N+1)} \sum_{i<j} [\mathit{Rig}(t_i, t_j)]^2}\;.$$

Since $\mathit{Flex}(t_i, t_j) \geq 0$, we have both that $\mathit{Rig}(t_i, t_j) \in [0, 1]$ and that $\mathit{Rig}(\mathcal{S}) \in [0, 1]$. If t_i and t_j are part of a rigid component, then $\mathit{Rig}(t_i, t_j) = 1$. Similarly, if $\mathcal{S}$ is completely rigid, then $\mathit{Rig}(\mathcal{S}) = 1$. At the opposite extreme, if $\mathcal{S}$ has absolutely no constraints, then $\mathit{Rig}(\mathcal{S}) = 0$.

Fact 7 (Triangle Inequality) *(Tsamardinos 2000) From Definitions 3 and 4, we get that the following holds among each triple of time-points t_i, t_j and t_k in an STN:*

$$\mathcal{D}(t_i, t_k) \;\leq\; \mathcal{D}(t_i, t_j) + \mathcal{D}(t_j, t_k).$$

Definition 7 (Tight Edge/Constraint) *(Morris & Muscettola 2000) A* tight *constraint (or edge) is an explicit constraint $(t_j - t_i \leq \delta)$ for which $\delta = \mathcal{D}(t_i, t_j)$.*

[1] The concept of temporal distance, implicit in Dechter et al., is made explicit in Tsamardinos (2000).

[2] The concept of the distance matrix is implicit in Dechter et al. Tsamardinos (2000) uses the term *distance array*.

[3] An earlier paper (Hunsberger 2002) defines a similar measure of rigidity.

The Temporal Decoupling Problem

This section formally defines the Temporal Decoupling Problem and presents theorems characterizing its solutions. To simplify the presentation, we restrict attention to the case of partitioning an STN $\mathcal{S}$ into two independent subnetworks $\mathcal{S}_X$ and $\mathcal{S}_Y$. The case of an arbitrary number of decoupled subnetworks is analogous.

Definition 8 (z-Partition) *If $\mathcal{T}$, $\mathcal{T}_X$ and $\mathcal{T}_Y$ are sets of time-point variables such that:*

$$\mathcal{T}_X \cap \mathcal{T}_Y = \{z\} \quad \textit{and} \quad \mathcal{T}_X \cup \mathcal{T}_Y = \mathcal{T},$$

then we say that $\mathcal{T}_X$ and $\mathcal{T}_Y$ z-partition *$\mathcal{T}$.*

Definition 9 (Temporal Decoupling) *We say that the STNs $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$ are a* temporal decoupling *of the STN $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ if:*

- *$\mathcal{S}_X$ and $\mathcal{S}_Y$ are consistent STNs;*
- *$\mathcal{T}_X$ and $\mathcal{T}_Y$ z-partition $\mathcal{T}$; and*
- *(Mergeable Solutions Property) Merging* any *solutions for $\mathcal{S}_X$ and $\mathcal{S}_Y$ necessarily yields a solution for $\mathcal{S}$.*

(We may also say that $\mathcal{S}_X$ and $\mathcal{S}_Y$ partition *$\mathcal{S}$ into tempo-*rally independent subnetworks.*)*

Result 8 *If $\mathcal{S}_X$ and $\mathcal{S}_Y$ are a temporal decoupling of $\mathcal{S}$, then $\mathcal{S}$ is consistent.*

Proof Since $\mathcal{S}_X$ and $\mathcal{S}_Y$ are required to be consistent, each has at least one solution; the merging of any such solutions yields a solution for $\mathcal{S}$. ■

Definition 10 (The Temporal Decoupling Problem) *Given an STN $\mathcal{S}$ whose time-points $\mathcal{T}$ are z-partitioned by $\mathcal{T}_X$ and $\mathcal{T}_Y$, find sets of constraints $\mathcal{C}_X$ and $\mathcal{C}_Y$ such that $(\mathcal{T}_X, \mathcal{C}_X)$ and $(\mathcal{T}_Y, \mathcal{C}_Y)$ temporally decouple $\mathcal{S}$.*

Result 9 *Any instance of the TDP in which $\mathcal{S}$ is consistent has a solution.*

Proof Let $\mathcal{S}$ be a consistent STN whose time-points $\mathcal{T}$ are z-partitioned by $\mathcal{T}_X = \{z, x_1, \ldots, x_m\}$ and $\mathcal{T}_Y = \{z, y_1, \ldots, y_n\}$. Let

$$\{z = 0, x_1 = v_1, \ldots, x_m = v_m; y_1 = w_1, \ldots, y_n = w_n\}$$

be an arbitrary solution for $\mathcal{S}$. Then the following specifies a temporal decoupling of $\mathcal{S}$:

$$\begin{aligned} \mathcal{C}_X &= \{x_1 = v_1, \ldots, x_m = v_m\} \\ \mathcal{C}_Y &= \{y_1 = w_1, \ldots, y_n = w_n\}. \blacksquare \end{aligned}$$

We call such decouplings *rigid decouplings*. One problem with rigid decouplings is that the subnetworks $\mathcal{S}_X$ and $\mathcal{S}_Y$ are completely rigid (i.e., completely inflexible). Below, we provide necessary and sufficient characterizations of solutions to the TDP that will point the way to TDP algorithms that yield more flexible decoupled subnetworks.

Theorem 10 (Necessary Conditions) *If the STNs $\mathcal{S}_X$ and $\mathcal{S}_Y$ are a temporal decoupling of the STN $\mathcal{S}$, then the following four properties must hold:*

(1) $\mathcal{D}_X(x_i, x_j) \le \mathcal{D}(x_i, x_j)$ *for each* $x_i, x_j \in \mathcal{T}_X$*;*
(2) $\mathcal{D}_Y(y_i, y_j) \le \mathcal{D}(y_i, y_j)$ *for each* $y_i, y_j \in \mathcal{T}_Y$*;*
(3) $\mathcal{D}_X(x, z) + \mathcal{D}_Y(z, y) \le \mathcal{D}(x, y)$ *for each* $x \in \mathcal{T}_X$ *and* $y \in \mathcal{T}_Y$*; and*
(4) $\mathcal{D}_Y(y, z) + \mathcal{D}_X(z, x) \le \mathcal{D}(y, x)$ *for each* $x \in \mathcal{T}_X$ *and* $y \in \mathcal{T}_Y$*,*

where $\mathcal{D}_X$, $\mathcal{D}_Y$, and $\mathcal{D}$ are the distance matrices for $\mathcal{S}_X$, $\mathcal{S}_Y$ and $\mathcal{S}$, respectively.

Proof Let $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$ be an arbitrary temporal decoupling of the STN $\mathcal{S} = (\mathcal{T}, \mathcal{C})$. From Definition 9, both $\mathcal{S}_X$ and $\mathcal{S}_Y$ must be consistent.

Property 1: Let $x_i, x_j \in \mathcal{T}_X$ be arbitrary. By Theorem 4, there is a solution $\mathcal{X}$ for $\mathcal{S}_X$ in which $x_j - x_i = \mathcal{D}_X(x_i, x_j)$. Let $\mathcal{Y}$ be an arbitrary solution for $\mathcal{S}_Y$. Since $\mathcal{S}_X$ and $\mathcal{S}_Y$ are a temporal decoupling of $\mathcal{S}$, merging the solutions $\mathcal{X}$ and $\mathcal{Y}$ must yield a solution for $\mathcal{S}$. In that solution (for $\mathcal{S}$) we have that $x_j - x_i = \mathcal{D}_X(x_i, x_j)$. However, being a solution for $\mathcal{S}$ also implies that: $x_j - x_i \le \mathcal{D}(x_i, x_j)$.

Property 3: Let $x \in \mathcal{T}_X$ and $y \in \mathcal{T}_Y$ be arbitrary. Let $\mathcal{X}$ be a solution for $\mathcal{S}_X$ in which $z - x = \mathcal{D}_X(x, z)$. Similarly, let $\mathcal{Y}$ be a solution for $\mathcal{S}_Y$ in which $y - z = \mathcal{D}_Y(z, y)$. Merging the solutions $\mathcal{X}$ and $\mathcal{Y}$ must yield a solution for $\mathcal{S}$. In that solution, we have that: $y - x = (y - z) + (z - x) = \mathcal{D}_Y(z, y) + \mathcal{D}_X(x, z)$. However, being a solution for $\mathcal{S}$ also implies that: $y - x \le \mathcal{D}(x, y)$.

Properties 2 and 4 are handled analogously. ■

Theorem 11 (Sufficient Conditions) *Let $\mathcal{S} = (\mathcal{T}, \mathcal{C})$, $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$ be consistent STNs such that $\mathcal{T}_X$ and $\mathcal{T}_Y$ z-partition $\mathcal{T}$. If Properties 1–4 of Theorem 10 hold, then $\mathcal{S}_X$ and $\mathcal{S}_Y$ are a temporal decoupling of $\mathcal{S}$.*

Proof Suppose $\mathcal{S}$, $\mathcal{S}_X$ and $\mathcal{S}_Y$ satisfy the above conditions. The only part of the definition of a temporal decoupling (Definition 9) that is non-trivial to verify in this setting is the Mergeable Solutions Property. Let

$$\begin{aligned} \mathcal{X} &= \{z = 0,\ x_1 = v_1,\ \ldots,\ x_m = v_m\} \text{ and} \\ \mathcal{Y} &= \{z = 0,\ y_1 = w_1,\ \ldots,\ y_n = v_n\} \end{aligned}$$

be arbitrary solutions for $\mathcal{S}_X$ and $\mathcal{S}_Y$, respectively. We need to show that $\mathcal{X} \cup \mathcal{Y}$ is a solution for $\mathcal{S}$. Let E: $t_j - t_i \le \delta$ be an arbitrary constraint in $\mathcal{C}$. We need to show that the constraint E is satisfied by the values in $\mathcal{X} \cup \mathcal{Y}$.

Case 1: $t_i = x_p$ and $t_j = x_q$ are both elements of $\mathcal{T}_X$.
Since $\mathcal{X}$ is a solution for $\mathcal{S}_X$, we have that: $v_q - v_p \le \mathcal{D}_X(x_p, x_q)$. Since Property 1 holds, we have that: $\mathcal{D}_X(x_p, x_q) \le \mathcal{D}(x_p, x_q)$. Finally, since E is a constraint in $\mathcal{C}$, we have that: $\mathcal{D}(x_p, x_q) \le \delta$. Thus, $v_q - v_p \le \delta$ (i.e., E is satisfied by $x_p = v_p$ and $x_q = v_q$).

Case 2: $t_i = y_p$ and $t_j = y_q$ are both elements of $\mathcal{T}_Y$.
Handled analogously to Case 1.

Case 3: $t_i = x_p \in \mathcal{T}_X$ and $t_j = y_q \in \mathcal{T}_Y$.
Since $x_p = v_p$ is part of a solution for $\mathcal{S}_X$, we have that: $0 - v_p \le \mathcal{D}_X(x_p, z)$. Similarly, $w_q - 0 \le \mathcal{D}_Y(z, y_q)$. Thus $w_q - v_p \le \mathcal{D}_X(x_p, z) + \mathcal{D}_Y(z, y_q)$. Since Property 3 holds, we get that: $\mathcal{D}_X(x_p, z) + \mathcal{D}_Y(z, y_q) \le \mathcal{D}(x_p, y_q)$.

Finally, since E is a constraint in $\mathcal{C}$, we have that: $\mathcal{D}(x_p, y_q) \leq \delta$. Thus, $w_q - v_p \leq \delta$ (i.e., the constraint E is satisfied by $x_p = v_p$ and $y_q = w_q$).

Case 4: $t_i = y_q \in \mathcal{T}_Y$ and $t_j = x_p \in \mathcal{T}_X$.
Handled analogously to Case 3.

Since the constraint E was chosen arbitrarily from $\mathcal{C}$, we have that $\mathcal{X} \cup \mathcal{Y}$ is a solution for $\mathcal{S}$. ■

Toward a TDP Algorithm

Definition 11 (xy-Pairs, xy-Edges) *Let $\mathcal{T}_X$, $\mathcal{T}_Y$ and $\mathcal{T}$ be sets of time-points such that $\mathcal{T}_X$ and $\mathcal{T}_Y$ z-partition $\mathcal{T}$. Let t_i and t_j be arbitrary time-points in $\mathcal{T}$. The pair (t_i, t_j) is called an* xy-pair *if:*

$$(t_i \in \mathcal{T}_X \text{ and } t_j \in \mathcal{T}_Y) \quad \text{or} \quad (t_i \in \mathcal{T}_Y \text{ and } t_j \in \mathcal{T}_X).$$

If, in addition, neither t_i nor t_j is the zero time-point variable z, then (t_i, t_j) is called a proper *xy-pair. A constraint (or edge), $(t_j - t_i \leq \delta)$, is called an* xy-edge *if (t_i, t_j) is an xy-pair. An xy-edge is called a* proper *xy-edge if the corresponding pair is a proper xy-pair.*

Definition 12 (Zero-Path-Shortfall) *Let E be a tight, proper xy-edge $(t_j - t_i \leq \delta)$. The* zero-path shortfall *(ZPS) associated with E is the quantity:*

$$\text{ZPS}(E) = [\mathcal{D}(t_i, z) + \mathcal{D}(z, t_j)] - \delta.$$

Result 12 *For any tight, proper xy-edge E, $ZPS(E) \geq 0$.*

Proof Let E be a tight, proper xy-edge $(t_j - t_i \leq \delta)$. Since E is tight, we have: $\mathcal{D}(t_i, t_j) = \delta$; hence, from the Triangle Inequality: $\delta = \mathcal{D}(t_i, t_j) \leq \mathcal{D}(t_i, z) + \mathcal{D}(z, t_j)$. ■

Definition 13 (Dominated by a Path through Zero) *If the ZPS value for a tight, proper xy-edge is zero, we say that that edge is* dominated by a path through zero.

Result 13 *If adding a set of constraints to a consistent STN $\mathcal{S}$ does not make $\mathcal{S}$ inconsistent, then the ZPS values for any pre-existing tight, proper xy-edges in $\mathcal{S}$ cannot increase.*

Proof Adding constraints to an STN causes its shortest paths to become shorter or stay the same, and hence its distance matrix entries to decrease or stay the same. Given that δ is a constant, this implies that $[\mathcal{D}(t_i, z) + \mathcal{D}(z, t_j)] - \delta$ can only decrease or stay the same. ■

The following lemma shows that we can restrict attention to tight, proper xy-edges when seeking a solution to an instance of the TDP.

Lemma 14 *If $\mathcal{D}(t_p, z) + \mathcal{D}(z, t_q) \leq \delta$ holds for every* tight, proper *xy*-edge *$(t_q - t_p \leq \delta)$ in a consistent STN, then $\mathcal{D}(t_i, z) + \mathcal{D}(z, t_j) \leq \mathcal{D}(t_i, t_j)$ holds for every xy*-pair *(t_i, t_j) in that STN.*

Proof Suppose that $\mathcal{D}(t_p, z) + \mathcal{D}(z, t_q) \leq \delta$ holds for every tight, proper xy-*edge* $(t_q - t_p \leq \delta)$. Let (t_i, t_j) be an arbitrary xy-*pair* in $\mathcal{S}$. We must show that $\mathcal{D}(t_i, z) + \mathcal{D}(z, t_j) \leq \mathcal{D}(t_i, t_j)$ holds.

If t_i or t_j is the zero time-point z, then the inequality holds trivially (since $\mathcal{D}(z, z) = 0$ in a consistent STN). Thus, suppose that (t_i, t_j) is a *proper* xy-pair. Without loss of generality, suppose $t_i \in \mathcal{T}_X$ and $t_j \in \mathcal{T}_Y$. Let P be an arbitrary

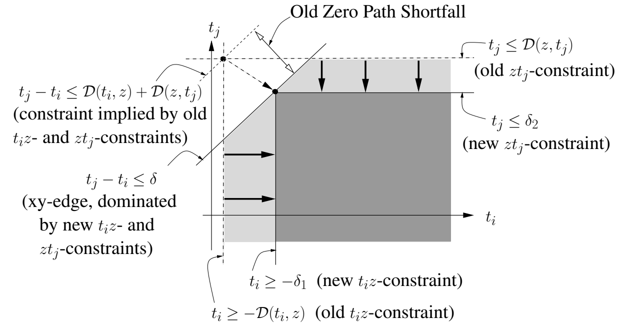


Figure 1: Reducing the zero-path shortfall for an xy-edge

shortest path from t_i to t_j in $\mathcal{S}$. If z is on the path P, then the inequality holds (since the subpaths from t_i to z and from z to t_j must also be shortest paths).

Now suppose z is not on P. Then P must contain at least one proper xy-edge E_{xy} of the form $(y - x \leq \delta_{xy})$, where $x \in \mathcal{T}_X$ and $y \in \mathcal{T}_Y$. Since E_{xy} is on a shortest path, it must be tight; hence $\delta_{xy} = \mathcal{D}(x, y)$. Thus, using the Triangle Inequality: $\delta_{xy} = \mathcal{D}(x, y) \leq \mathcal{D}(x, z) + \mathcal{D}(z, y)$ holds. But the Lemma's premise, applied to the tight, proper xy-edge E_{xy}, gives us that: $\mathcal{D}(x, z) + \mathcal{D}(z, y) \leq \delta_{xy}$. Thus, $\delta_{xy} = \mathcal{D}(x, z) + \mathcal{D}(z, y)$. Thus, we may replace E_{xy} in P by a pair of shortest paths, one from x to z, one from z to y, without changing the length of P. But then this version of P is a shortest path from t_i to t_j that contains z. As argued earlier, this implies that the desired inequality holds. ■

Algorithms for Solving the TDP

This section presents a family of sound and complete algorithms for solving the Temporal Decoupling Problem. We begin by presenting a preliminary TDP algorithm, directly motivated by Theorem 11, above. The preliminary algorithm is sound, but not complete: because it is not guaranteed to terminate. In Theorems 21 and 22, below, we specify ways of strengthening the preliminary algorithm to ensure that it terminates and, hence, that it is complete.

The Preliminary TDP Algorithm

The main tool of the TDP algorithm is to reduce the zero-path shortfall for each tight, proper xy-edge, $t_j - t_i \leq \delta$, by strengthening the corresponding t_iz- and/or zt_j-edges. Figure 1 illustrates the case of an xy-edge's ZPS value being reduced to zero through the addition of edges $z - t_i \leq \delta_1$ (i.e., $t_i \geq -\delta_1$) and $t_j - z \leq \delta_2$ (i.e., $t_j \leq \delta_2$). Adding weaker constraints may reduce the zero-path shortfall but not eliminate it entirely.

The preliminary algorithm is given in pseudo-code in Figure 2. It takes as input an STN $\mathcal{S}$ whose time-points are z-partitioned by the sets $\mathcal{T}_X$ and $\mathcal{T}_Y$.

At Step 1, the algorithm checks whether $\mathcal{S}$ is consistent. If $\mathcal{S}$ is inconsistent, the algorithm returns `NIL` and halts because, by Result 8, only consistent STNs can be decoupled. Otherwise, the algorithm initializes the set $\mathcal{E}$ to the set of

Given: An STN $\mathcal{S}$ whose time-points $\mathcal{T}$ are z-partitioned by the sets $\mathcal{T}_X$ and $\mathcal{T}_Y$.

(1) Compute the distance matrix $\mathcal{D}$ for $\mathcal{S}$. If $\mathcal{S}$ is inconsistent, return `NIL` and halt; otherwise, initialize $\mathcal{E}$ to the set of tight, proper xy-edges in $\mathcal{S}$, and continue.

(2) Select a tight, proper xy-edge $E = (t_j - t_i \leq \delta)$ in $\mathcal{E}$ whose ZPS value is positive. (If, in the process, any edges in $\mathcal{E}$ are discovered that are no longer tight or that have a ZPS value of zero, remove those edges from $\mathcal{E}$.) If no such edges exist (i.e., if $\mathcal{E}$ has become empty), go to Step 6; otherwise, continue.

(3) Pick values δ_1 and δ_2 such that:

$$\begin{array}{rcccl} -\mathcal{D}(z,t_i) & \leq & \delta_1 & \leq & \mathcal{D}(t_i,z), \\ -\mathcal{D}(t_j,z) & \leq & \delta_2 & \leq & \mathcal{D}(z,t_j), \quad \text{and} \\ \delta & \leq & \delta_1+\delta_2 & < & \mathcal{D}(t_i,z)+\mathcal{D}(z,t_j). \end{array}$$

(4) Add the constraints, E_1: $z - t_i \leq \delta_1$ and E_2: $t_j - z \leq \delta_2$, to $\mathcal{S}$, updating $\mathcal{D}$ to reflect the new constraints.

(5) Go to Step 2.

(6) Return: $\mathcal{C}_X = \{(t_j - t_i \leq \mathcal{D}(t_i,t_j)) : t_i, t_j \in \mathcal{T}_X\}$;
$\mathcal{C}_Y = \{(t_j - t_i \leq \mathcal{D}(t_i,t_j)) : t_i, t_j \in \mathcal{T}_Y\}$.

Figure 2: Pseudo-code for the Preliminary TDP Algorithm

tight, proper xy-edges in $\mathcal{S}$, an $O(N^2)$ computation that requires checking each edge against the corresponding entries in the distance matrix. The algorithm then iteratively operates on edges drawn from $\mathcal{E}$ until each such edge is dominated by a path through zero (recall Definition 13).

For each iteration, the algorithm does the following. In Step 2, a tight, proper xy-edge E: $(t_j - t_i \leq \delta)$ with a positive zero-path shortfall is selected from the set $\mathcal{E}$. In Steps 3 and 4, new constraints involving the zero time-point variable z are added to $\mathcal{S}$. After propagating these constraints (i.e., after updating the distance matrix to reflect the new constraints), the new values of $\mathcal{D}(t_i,z)$ and $\mathcal{D}(z,t_j)$ will be δ_1 and δ_2, respectively, where δ_1 and δ_2 are the values from Step 3. Thus, the *updated* ZPS value for E, which we denote ζ^*, will be: $\zeta^* = \delta_1 + \delta_2 - \delta$.

Upon adding the Step 4 edges to $\mathcal{S}$, it may be that some of the edges in $\mathcal{E}$ are no longer tight or no longer have positive ZPS values. However, the algorithm need not check for that in Step 4. Instead, if any such edges are ever encountered during the selection process in Step 2, they are simply removed from $\mathcal{E}$ at that time.

If it ever happens that every tight, proper xy-edge is dominated by a path through zero, as evidenced by the set $\mathcal{E}$ becoming empty, then the algorithm terminates (see Steps 2 and 6). The sets $\mathcal{C}_X$ and $\mathcal{C}_Y$ returned by the algorithm are derived from the distance matrix $\mathcal{D}$ which has been updated to include all of the constraints added during passes of Step 4.

Soundness of the Preliminary TDP Algorithm

The values δ_1 and δ_2 chosen in Step 3 of the algorithm specify the strengths of the constraints E_1 and E_2 added in Step 4. It is also useful to think in terms of the amount by which the ZPS value for the edge under consideration is thereby reduced (call it R), as well as the fractions of

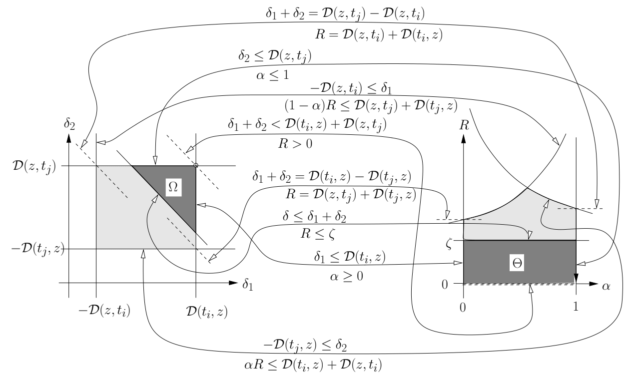


Figure 3: The regions Ω and Θ from Lemma 16

this ZPS reduction due to the tightening of the t_iz- and zt_j-edges, respectively (specified by α and $1-\alpha$). Lemma 16 below gives the precise relationship between the pairs of values (δ_1,δ_2) and (R,α). We subsequently use the results of Lemma 16 to prove that the preliminary TDP algorithm is sound. Result 15 is used in Lemma 16.

Result 15 *For an edge E of the form $t_j - t_i \leq \delta$, if that edge is* tight*, then the following inequalities necessarily hold:*

$$\mathcal{D}(t_i,z) - \mathcal{D}(t_j,z) \leq \delta \;\; \textit{and} \;\; \mathcal{D}(z,t_j) - \mathcal{D}(z,t_i) \leq \delta.$$

Proof The first inequality may be proven as follows:

$\mathcal{D}(t_i,z) \leq \mathcal{D}(t_i,t_j) + \mathcal{D}(t_j,z)$	(Triangle Inequality)
$\mathcal{D}(t_i,z) \leq \delta + \mathcal{D}(t_j,z)$	(Since E is a tight edge)
$\mathcal{D}(t_i,z) - \mathcal{D}(t_j,z) \leq \delta$	(Rearrange terms)

The second inequality follows similarly. ■

Lemma 16 *Let E be some tight, proper xy-edge $t_j - t_i \leq \delta$ whose ZPS value $\zeta = ZPS(E)$ is positive. Let Ω be the set of ordered pairs (δ_1,δ_2) satisfying the Step 3 requirements of the preliminary algorithm (recall Figure 2). Let Θ be the set of ordered pairs (R,α) such that $R \in (0,\zeta]$ and $\alpha \in [0,1]$. Let T_1 and T_2 be the following 2-by-2 transformations:*

$$T_1\!: \begin{cases} \delta_1 = f(R,\alpha) = \mathcal{D}(t_i,z) - \alpha R \\ \delta_2 = g(R,\alpha) = \mathcal{D}(z,t_j) - (1-\alpha)R \end{cases}$$

$$T_2\!: \begin{cases} R = u(\delta_1,\delta_2) = \mathcal{D}(t_i,z) + \mathcal{D}(z,t_j) - (\delta_1+\delta_2) \\ \alpha = v(\delta_1,\delta_2) = \frac{\mathcal{D}(t_i,z)-\delta_1}{\mathcal{D}(t_i,z)+\mathcal{D}(z,t_j)-(\delta_1+\delta_2)} \end{cases}$$

Then T_1 and T_2 are invertible transformations between Ω and Θ (with $T_1^{-1} = T_2$) such that for any pair $(\delta_1,\delta_2) \in \Omega$, the corresponding pair $(R,\alpha) \in \Theta$ satisfies that:

- *R is the amount by which the pair of corresponding Step 4 constraints, E_1: $z - t_i \leq \delta_1$ and E_2: $t_j - z \leq \delta_2$, reduce the ZPS value for the edge E, and*
- *α and $(1-\alpha)$ represent the fractions of this ZPS reduction due to the tightening of the t_iz- and zt_j-edges, respectively.*

Proof The Step 3 requirements (from Figure 2) correspond to the boundaries of the region Ω in Figure 3. Note that the point $(\mathcal{D}(t_i,z), \mathcal{D}(z,t_j))$ is *not* part of Ω due to the strict inequality: $\delta_1 + \delta_2 < \mathcal{D}(t_i,z) + \mathcal{D}(z,t_j)$. Also, by Result 15,

$\mathcal{D}(t_i, z) - \mathcal{D}(t_j, z) \leq \delta$ and $\mathcal{D}(z, t_j) - \mathcal{D}(z, t_i) \leq \delta$. These inequalities ensure that the diagonal boundary of Ω, which corresponds to the constraint $\delta \leq \delta_1 + \delta_2$, lies above and to the right of the lines $\mathcal{D}(t_i, z) - \mathcal{D}(t_j, z) = \delta_1 + \delta_2$ and $\mathcal{D}(z, t_j) - \mathcal{D}(z, t_i) = \delta_1 + \delta_2$ (shown as dashed lines in the $\delta_1\delta_2$-plane in Figure 3).

The amount of reduction in the ZPS value ζ that results from adding the corresponding Step 4 constraints, E_1: $z - t_i \leq \delta_1$ and E_2: $t_j - z \leq \delta_2$, is given by:

$$\begin{aligned}\text{(old ZPS value) - (new ZPS value)} &= \zeta - \zeta^* \\ &= [\mathcal{D}(t_i, z) + \mathcal{D}(z, t_j) - \delta] - [\delta_1 + \delta_2 - \delta] \\ &= \mathcal{D}(t_i, z) + \mathcal{D}(z, t_j) - (\delta_1 + \delta_2)\end{aligned}$$

which is precisely the value $R = u(\delta_1, \delta_2)$. The amount by which the $t_i z$-edge is strengthened is given by:

$$\text{(old value) - (new value)} = \mathcal{D}(t_i, z) - \delta_1.$$

Hence, the fraction of the total ZPS reduction produced by strengthening the $t_i z$-edge is precisely $\alpha = v(\delta_1, \delta_2)$, from which it also follows that $\delta_1 = f(R, \alpha) = \mathcal{D}(t_i, z) - \alpha R$. Similarly, the fraction of the total ZPS reduction produced by strengthening the $z t_j$-edge is precisely $(1 - \alpha)$, and $\delta_2 = g(R, \alpha) = \mathcal{D}(z, t_j) - (1 - \alpha)R$.

It is easy to verify that the transformations T_1 and T_2 are inverses of one another. The arrows in Figure 3 show how the boundaries of the regions Ω and Θ correspond.

Finally, from Theorem 4, the quantity $\mathcal{D}(z, t_j) + \mathcal{D}(t_j, z)$ specifies the maximum amount that the $z t_j$-edge can be tightened without threatening the consistency of the STN. The following establishes that ζ (i.e., the ZPS value for the edge E) is no more than this amount:

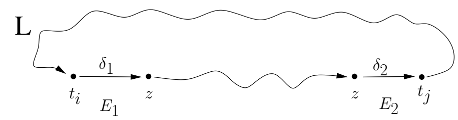
$$\begin{aligned}&\mathcal{D}(t_i, z) - \mathcal{D}(t_j, z) \leq \delta && \text{(Established earlier)} \\ \Rightarrow\ &\mathcal{D}(t_i, z) + \mathcal{D}(z, t_j) - \delta \leq \mathcal{D}(z, t_j) + \mathcal{D}(t_j, z) && \\ \Rightarrow\ &\zeta \leq \mathcal{D}(z, t_j) + \mathcal{D}(t_j, z) && \text{(Defn. of } \zeta)\end{aligned}$$

Thus, the entire zero-path-shortfall for E may be eliminated by tightening only the $z t_j$-constraint. Similarly, the entire zero-path-shortfall may be eliminated by instead tightening only the $t_i z$-constraint. In Figure 3, these constraints on ζ ensure that the top horizontal boundary of the region Θ lies below the curves $\alpha R = \mathcal{D}(t_i, z) + \mathcal{D}(z, t_i)$ and $(1 - \alpha)R = \mathcal{D}(z, t_j) + \mathcal{D}(t_j, z)$. Thus, the values for R and α may be independently chosen. ■

Corollary 17 *Let E: $t_j - t_i \leq \delta$ be a tight, proper xy-edge with ZPS value $\zeta > 0$. Let $R \in (0, \zeta]$ and $\alpha \in [0, 1]$ be arbitrary. It is always possible to choose δ_1 and δ_2 satisfying the Step 3 requirements such that the ZPS value for E will be reduced by R and such that the $t_i z$- and $z t_j$-edges will be tightened by the amounts αR and $(1 - \alpha)R$, respectively.*

Theorem 18 *Suppose $\mathcal{S}$ is a consistent STN and that E: $t_j - t_i \leq \delta$ is a tight, proper xy-edge whose ZPS value is positive. Let δ_1 and δ_2 be arbitrary values chosen according to the requirements of Step 3. Then adding the* pair *of corresponding Step 4 constraints (i.e., E_1 and E_2 in Figure 2) will not threaten the consistency of $\mathcal{S}$.*

Proof Let $\zeta > 0$ be the ZPS value for edge E. Suppose that adding the corresponding Step 4 constraints E_1 and E_2 caused $\mathcal{S}$ to become inconsistent. Then, by Theorem 1, there must be a loop in $G_{\mathcal{S}}$ with negative path-length. By Theorem 4, the first two Step 3 requirements (from Figure 2) imply that E_1 and E_2 are individually consistent with $\mathcal{S}$. Thus, any loop with negative path length in $G_{\mathcal{S}}$ must contain *both* E_1 and E_2. Of all such loops, let L be one that has the minimum number of edges.



Now consider the subpath from z (at the end of E_1) to z (at the beginning of E_2). This is itself a loop. Since that loop has fewer edges than L, it must, by the choice of L, have non-negative path-length. But then extracting this subpath from L would result in a loop L' still having negative path-length. Since the choice of L precludes L' having fewer edges than L, it must be that the subpath from z to z is empty. However, part of the third Step 3 requirement (from Figure 2) says that $\delta \leq \delta_1 + \delta_2$, which implies that the edges E_1 and E_2 in L could be replaced by the edge, $t_j - t_i \leq \delta$, resulting in a loop having negative path-length but with fewer edges than L, contradicting the choice of L. Thus, no such L exists. Thus, adding both E_1 and E_2 to $\mathcal{S}$ leaves $\mathcal{S}$ consistent. ■

Theorem 19 (Soundness) *If the temporal decoupling algorithm terminates at Step 6, then the constraint sets $\mathcal{C}_X$ and $\mathcal{C}_Y$ returned by the TDP algorithm are such that $(\mathcal{T}_X, \mathcal{C}_X)$ and $(\mathcal{T}_Y, \mathcal{C}_Y)$ are a temporal decoupling of the input STN $\mathcal{S}$.*

Proof During each pass of Step 4, the TDP algorithm modifies the input STN by adding new constraints. To distinguish the input STN $\mathcal{S}$ from the modified STN existing at the end of the algorithm's execution (i.e., at Step 6), we shall refer to the latter as $\mathcal{S}'$. If $\mathcal{C}'$ is the set of all Step 4 constraints added during the execution of the algorithm, then $\mathcal{S}' = (\mathcal{T}, \mathcal{C} \cup \mathcal{C}')$. Let $\mathcal{D}'$ be the distance matrix for $\mathcal{S}'$. (Thus, using this notation, it is $\mathcal{D}'$ that is used to construct the constraint sets $\mathcal{C}_X$ and $\mathcal{C}_Y$ in Step 6.) Since every constraint in $\mathcal{S}$ is present in $\mathcal{S}'$, $\mathcal{D}'(t_i, t_j) \leq \mathcal{D}(t_i, t_j)$ for all $t_i, t_j \in \mathcal{T}$.

To show that $\mathcal{S}_X$ and $\mathcal{S}_Y$ are a temporal decoupling of $\mathcal{S}$, it suffices (by Theorem 11) to show that $\mathcal{S}_X$ and $\mathcal{S}_Y$ are each consistent and that Properties 1–4 from Theorem 10 hold. (It is given that the sets $\mathcal{T}_X$ and $\mathcal{T}_Y$ z-partition $\mathcal{T}$.)

Theorem 18 guarantees that $\mathcal{S}'$ is consistent. Furthermore, since any solution for $\mathcal{S}'$ must satisfy the constraints represented in $\mathcal{D}'$, which cover all the constraints in $\mathcal{C}_X$ and $\mathcal{C}_Y$, both $\mathcal{S}_X$ and $\mathcal{S}_Y$ must also be consistent.

By construction, $\mathcal{D}_X(x_p, x_q) = \mathcal{D}'(x_p, x_q)$ for every x_p and x_q in $\mathcal{T}_X$. Since $\mathcal{D}'(x_p, x_q) \leq \mathcal{D}(x_p, x_q)$, Property 1 of Theorem 10 holds. Similarly, Property 2 also holds.

To prove that Property 3 holds, first notice that the premise of Lemma 14 is equivalent to saying that ZPS ≤ 0 for each tight, proper xy-*edge*, which is precisely what the exit clause of Step 2 requires. Thus, the algorithm will not terminate at Step 6 unless the premise of Lemma 14 holds—with respect to $\mathcal{S}'$. Hence, from the conclusion of Lemma 14, we have that for any xy-*pair* in $\mathcal{S}'$: $\mathcal{D}'(t_i, z) + \mathcal{D}'(z, t_j) \leq \mathcal{D}'(t_i, t_j)$. In the case where $t_i \in \mathcal{T}_X$ and $t_j \in \mathcal{T}_Y$, we have that $\mathcal{D}'(t_i, z) = \mathcal{D}_X(t_i, z)$

and $\mathcal{D}'(z, t_j) = \mathcal{D}_Y(z, t_j)$. Since $\mathcal{D}'(t_i, t_j) \le \mathcal{D}(t_i, t_j)$, we get that $\mathcal{D}_X(t_i, z) + \mathcal{D}_Y(z, t_j) \le \mathcal{D}(t_i, t_j)$, which is Property 3. Similarly, Property 4 holds. ■

Ensuring Completeness for the TDP Algorithm

The Step 3 requirement that $\delta_1 + \delta_2 < \mathcal{D}(t_i, z) + \mathcal{D}(z, t_j)$ ensures that the ZPS value for the edge under consideration will decrease. However, it does not ensure that *substantial* progress will be made. As a result, the preliminary algorithm, as shown in Figure 2, is not guaranteed to terminate. Theorems 21 and 22, below, specify two ways of strengthening the Step 3 requirements, each sufficient to ensure that the TDP algorithm will terminate and, hence, that it is complete. Each strategy involves a method for choosing R, the amount by which the ZPS value for the edge currently under consideration is to be reduced (recall Lemma 16). Each strategy leaves the distribution of additional constrainedness among the $t_i z$ and $z t_j$ edges (i.e., the choice of α) unrestricted.

Fact 20 will be used in the proofs of the theorems.

Fact 20 *If an xy-edge ever loses its tightness, it cannot ever regain it. Thus, since the algorithm never adds any proper xy-edges, the pool of tight, proper xy-edges relevant to Step 2 can never grow. Furthermore, by Result 13, the ZPS values cannot ever increase. Thus, any progress made by the algorithm is never lost.*

Theorem 21 (Greedy Strategy) *If at each pass of Step 3, the entire zero-path-shortfall for the edge under consideration is eliminated—which is always possible by Corollary 17—then the TDP algorithm will terminate after at most $2|\mathcal{T}_X||\mathcal{T}_Y|$ iterations.*

Proof By Fact 20, the algorithm needs to do Step 3 processing of each tight, proper xy-edge at most once. There are at most $2|\mathcal{T}_X||\mathcal{T}_Y|$ such edges. ■

Barring some extravagant selection process in Step 2, the computation in each iteration of the algorithm is dominated by the propagation of the temporal constraints added in Step 4. This is no worse than $O(N^3)$, where $N + 1$ is the number of time-points in $\mathcal{S}$ (recall Fact 2).

The following theorem specifies a less-greedy approach which, although more computationally expensive than the greedy approach, is shown in the experimental evaluation section to result in decoupled networks that are more flexible. In this strategy, the ZPS value of the edge under consideration in Step 3 is reduced by a fractional amount (unless it is already below some threshold). Unlike the Greedy strategy, this strategy requires all of the initial ZPS values to be finite—which is always the case in practice.

Theorem 22 (Less-Greedy Strategy) *Let Z be the maximum of the initial ZPS values among all the tight, proper xy-edges in $\mathcal{S}$. Let $\epsilon > 0$ and $r \in (0, 1)$ be arbitrary constants. Suppose that at each pass of Step 3 in the TDP algorithm, R (the amount by which the ZPS value ζ for the edge currently under consideration is reduced) is given by:*

$$R = \begin{cases} r\zeta, & \text{if } \zeta > \epsilon \\ \zeta, & \text{otherwise} \end{cases}$$

(Corollary 17 ensures that this is always possible.) If Z is finite, then this strategy ensures that the algorithm will terminate after at most $2|\mathcal{T}_X||\mathcal{T}_Y| \left\lceil \frac{\log(Z/\epsilon)}{\log(1/(1-r))} + 1 \right\rceil$ iterations.

Proof Let $\mathcal{E}_0$ be the initial set of tight, proper xy-edges having positive ZPS values. Let $E \in \mathcal{E}_0$ be arbitrary. Let ζ be E's initial ZPS value. Suppose E has been processed by the algorithm (in Step 3) n times so far. Given the above strategy for choosing R, E's current ZPS value ζ is necessarily bounded above by $\zeta_0(1 - r)^n$ and hence also by $Z(1 - r)^n$. If $n > \frac{\log(Z/\epsilon)}{\log(1/(1-r))}$, we get that $Z(1 - r)^n < \epsilon$. Thus, after at most $\left\lceil \frac{\log(Z/\epsilon)}{\log(1/(1-r))} + 1 \right\rceil$ appearances of E in Step 3, its ZPS value ζ will be zero. Since E was arbitrary and $|\mathcal{E}_0| \le 2|\mathcal{T}_X||\mathcal{T}_Y|$, the result is proven. ■

The factor $\left\lceil \frac{\log(Z/\epsilon)}{\log(1/(1-r))} + 1 \right\rceil$ specifies an upper bound on the run-time using the Less Greedy strategy as compared to the Greedy strategy. In practice, this factor may be kept small by choosing ϵ appropriately. For example, if $r = 0.5$ and $Z/\epsilon = 1000$, this factor is less than 11.

Corollary 23 (Completeness) *Using either the Greedy or Less-Greedy strategy, the TDP algorithm is complete.*

Proof Suppose a solution exists for an instance of the TDP for an STN $\mathcal{S}$. By Result 8, $\mathcal{S}$ must be consistent. Thus, the TDP algorithm will not halt at Step 1. Using either of the strategies in Theorems 21 or 22, the algorithm will eventually terminate at Step 6. By Theorem 19, this only happens when the algorithm has found a solution to the TDP. ■

Step 2 Choice	R	Randomly choose an edge from $\mathcal{E}$.
	K	Randomly select a K-item subset of $\mathcal{E}$, where K is one of $\{2, 4, 8\}$; choose edge from that subset whose processing in Step 3 will result in minimal change to STN's rigidity.
Choice of R	G	Greedy strategy
	L	Less-Greedy strategy where $r = 0.5$ and the computation-multiplier is either 6 or 18.
Choice of α	B	Randomly choose α from $\{0, 1\}$.
	U	Randomly choose α from $[0, 1]$ (uniform distribution).
	F	Randomly choose α from $[0, 1]$ with distribution weighted by flexibility of $t_i z$- and $z t_j$-edges.

Figure 4: Variations of the TDP Algorithm Tested

Experimental Evaluation

In this section, we compare the performance of the TDP algorithm across the following dimensions: (1) the function used in Step 2 to select the next edge to work on; (2) the function used in Step 3 to determine R (i.e., the amount of ZPS reduction); and (3) the function used in Step 3 to determine α, which governs the distribution of additional constrainedness among the $t_i z$- and $z t_j$-edges. The chart in Figure 4 shows the algorithm variations tested in the experiments. Each variation is identified by its parameter settings using the abbreviations in the chart.

Option K for the Step 2 choice function is expected to be computationally expensive since for each edge in the K-item subset, constraints must be propagated (and reset) and

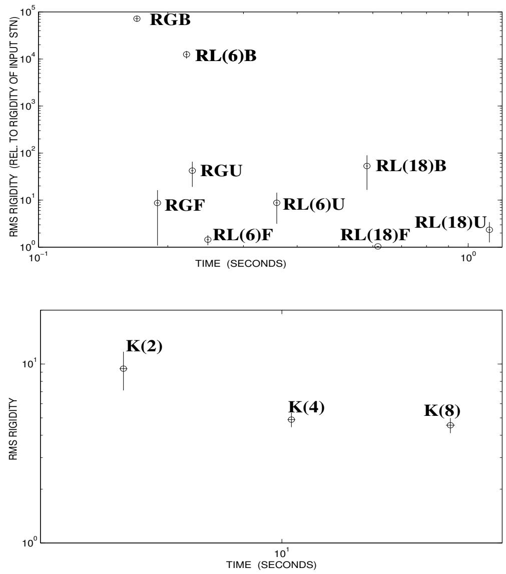


Figure 5: Experimental Results

the rigidity of the STN must be computed. However, it is hypothesized that this method will result in more flexible decouplings. Similarly, the Less-Greedy strategy, which is computationally more expensive than the Greedy strategy, is hypothesized to result in decouplings that are more flexible. Regarding the choice of α, it is hypothesized that one of the pseudo-continuous strategies (i.e., U or F) will result decouplings that are more flexible than when using the discrete strategy (i.e., B).

The first experiment tested the random Step 2 function (R). It consisted of 500 trials, each restricted to the time-interval $[0, 100]$. For each trial, the STN contained start and finish time-points for 30 actions (i.e., 60 time-points). Half of the actions/time-points were allocated to $\mathcal{T}_X$, half to $\mathcal{T}_Y$. Constraints were generated randomly, as follows. For each action, a lower bound d was drawn uniformly from the interval $[0, 1]$; an upper bound was drawn from $[d, d+1]$. Also, 400 constraints among time-points in $\mathcal{T}_X$, 400 among time-points in $\mathcal{T}_Y$, and 800 xy-edges were generated, the strength of each determined by selecting a random value from $[0, F]$, where F was 30% of the maximum amount the constraint could be tightened.

The results of the first experiment are shown in the top half of Figure 5. The horizontal axis measures time in seconds. The vertical axis measures the rigidity of the STN after the decoupling (as a multiple of the rigidity of the STN before the decoupling). 95% confidence intervals are shown for both time and rigidity, but the intervals for time are barely visible. Both scales are logarithmic.

As hypothesized, the Greedy approach (G) is faster, but the Less-Greedy approach (L) results in decouplings that are substantially more flexible (i.e., less rigid). Similarly, using a larger computation-multiplier in the Less-Greedy approach (18 vs. 6), which corresponds to a smaller value of ϵ, results in decouplings that are more flexible. The most surprising result is the dramatic benefit from using either of the two pseudo-continuous methods, U or F, for choosing α. Biasing the distribution according to the flexibility in the t_iz- and zt_j-edges (as is done in the F method) gives consistently more flexible decouplings, while taking less time to do so. The RL(18)F variation produced decouplings that were scarcely more rigid than the input STN.

The second experiment used only the K-item-set function (K) for the Step 2 selection function, varying the size of the subset (2, 4 or 8). The other dimensions were fixed: Less-Greedy approach with a computation-multiplier of 6, together with the F method of selecting α. The experiment consisted of 200 trials. For each trial the STN contained 40 actions (80 time-points), as well as 1600 constraints among time-points in $\mathcal{T}_X$, 1600 among time-points in $\mathcal{T}_Y$ and 3200 xy-edges. The results are shown in the bottom of Figure 5. As hypothesized, using the K-item-subset Step 2 function can generate decoupled networks that are substantially more flexible. However, using this method is not immune from the Law of Diminishing Returns. In this case, using an 8-item subset was not worth the extra computational effort.

Conclusions

In this paper, we formally defined the Temporal Decoupling Problem, presented theorems giving necessary and sufficient characterizations of solutions to the TDP, and gave a parameterized family of sound and complete algorithms for solving it. Although the algorithms were presented only in the case of decoupling an STN into two subnetworks, they are easily extended to the case of multiple subnetworks.

Acknowledgments

This research was supported by NSF grants IIS-9978343 and IRI-9618848. The author thanks Barbara J. Grosz and David C. Parkes for their helpful suggestions.

References

Cormen, T. H.; Leiserson, C. E.; and Rivest, R. L. 1990. *Introduction to Algorithms*. Cambridge, MA: The MIT Press.

Dechter, R.; Meiri, I.; and Pearl, J. 1991. Temporal constraint networks. *Artificial Intelligence* 49:61–95.

Hunsberger, L., and Grosz, B. J. 2000. A combinatorial auction for collaborative planning. In *Fourth International Conference on MultiAgent Systems (ICMAS-2000)*, 151–158. IEEE Computer Society.

Hunsberger, L. 2002. Generating bids for group-related actions in the context of prior commitments. In *Intelligent Agents VIII*, volume 2333 of *LNAI*. Springer-Verlag.

Morris, P., and Muscettola, N. 2000. Execution of temporal plans with uncertainty. In *Proc. of the 17th National Conference on Artificial Intelligence (AAAI-2000)*, 491–496.

Tsamardinos, I.; Muscettola, N.; and Morris, P. 1998. Fast transformation of temporal plans for efficient execution. In *Proc. of the 15th Nat'l. Conf. on AI (AAAI-98)*. 254–261.

Tsamardinos, I. 2000. Reformulating temporal plans for efficient execution. Master's thesis, Univ. of Pittsburgh.