Chapter 5

The Temporal Decoupling Problem

This chapter addresses an important family of temporal reasoning problems centered on finding ways of decoupling portions of a temporal network. Variations on the basic theme of temporal decoupling apply to several important problems in collaborative, multi-agent planning. For each variation, the problem is formally defined, theorems are presented characterizing solutions, and sound and complete algorithms are presented. Direct applications of these algorithms to problems related to the combinatorial auction-based ICDP mechanism described in Chapter 3 are discussed throughout this chapter.

5.1 Introduction

The Basic Temporal Decoupling Problem. The basic idea of temporal decoupling is to partition a temporal network into independent subnetworks. The characteristic property of such a decoupling is the Mergeable Solutions Property which says that any solutions to the independent subnetworks may be merged into a solution for the global network. If a given network is not initially decoupled into independent subnetworks, constraints must be added to achieve the decoupling. The temporal decoupling problem (TDP) is to find a set of constraints that achieves such a decoupling without needlessly constraining the network. If agents working on temporally dependent tasks in some group activity temporally decouple those tasks, then they may work independently, without requiring further coordination, thereby drastically reducing the need for negotiation about temporal dependencies, while costing the agents some amount of flexibility in their individual schedules. Thus, the basic TDP algorithm is directly applicable to the Post-Auction Coordination Problem described in Section 3.5.

The Allocative TDP. A slight variant of the basic TDP, called the Allocative TDP, is applicable to the situation faced by an agent wanting to participate in a task-allocation auction of the sort used in the ICDP mechanism described in Chapter 3. To keep its bid-generation computations manageable, the agent needs to restrict the portion of its schedule of pre-existing commitments that it uses as the basis for its bid-generation computations. To do so, the agent must decouple that portion of its schedule. The decoupled portion is called the agent's Auction-Participation Context. For the problem of generating a suitable

APC, there are two subnetworks to consider: $\mathcal{S}_X$, which corresponds to the portion of its schedule to be associated with the APC, and $\mathcal{S}_Y$, which corresponds to the rest of the agent's schedule. Decoupling these subnetworks enables the agent to deal with the rest of its schedule independently of the auction. For example, it need not consider the complicated issues associated with having outstanding bids when considering whether to add a new task to the rest of its schedule.

In the Allocative TDP, the contents of the subnetworks to be decoupled (i.e., the sets of time-points associated with $\mathcal{S}_X$ and $\mathcal{S}_Y$) are not known in advance. An algorithm for the Allocative TDP can be applied directly to the APC-Generation Problem described above. The Allocative TDP algorithm is an iterative algorithm that interleaves the allocation of time-points to the subnetworks with the addition of new constraints aimed at decoupling the subnetworks. The Allocative TDP algorithm is *lazy* in the sense that it does not establish the Mergeable Solutions Property at the end of each iteration; instead, it establishes it only at the end of the final iteration.

Relative Temporal Decoupling. When agents are working on temporally dependent tasks in a group activity, they may not want to accept the degree of extra constrainedness necessary to decouple the global network into completely independent subnetworks. One way to give agents greater flexibility, while still reducing the need for coordination, is to fully decouple the tasks being done by *some* of the agents. In this case, the subnetworks being decoupled do not cover all the time-points in the global network. The time-points that do not belong to any of the subnetworks being decoupled are collectively called the *relativizing set* $\mathcal{T}_W$. The resulting decoupling is called a temporal decoupling *relative* to $\mathcal{T}_W$. The characteristic Mergeable Solutions Property for a relative temporal decoupling says that the merger of any solutions for the decoupled subnetworks may be extended into a solution for the global network.

Another application of the relative temporal decoupling problem is to the problem of bid generation in a task allocation auction. When an agent places a bid on a set of tasks in such an auction, it is allowed to include temporal constraints with that bid. The purpose of the bid's temporal constraints is to allow the agent to protect the feasibility of its private schedule of pre-existing commitments should the bid eventually be awarded. The temporal-constraint-generation problem is, in effect, an instance of the relative temporal decoupling problem in which there is only one subnetwork $\mathcal{S}_1$ being decoupled relative to a leftover set of time-points. This case is not trivial given that the Mergeable Solutions Property must hold. The subnetwork $\mathcal{S}_1$ represents the time-points associated with the group activity (including the tasks on which the agent is bidding). The relativizing set $\mathcal{T}_W$ covers time-points associated with the rest of the agent's auction-participation context.

Adding new constraints to the relativizing set. An agent controlling a decoupled subnetwork may operate independently; however, this implies that an agent controlling the leftover time-points in the relativizing set $\mathcal{T}_W$ cannot be allowed to add any constraints that, upon propagation through the network, would add any extra constrainedness to any of the decoupled subnetworks. For example, suppose an agent generates temporal constraints for a bid in a task-allocation auction as described above. While the bid is outstanding, if that agent wants to add temporal constraints to the rest of the time-points in its auction-participation context (e.g., corresponding to a new commitment it might want to adopt), it must be sure that those new constraints do not jeopardize its ability to honor the outstanding bid should that bid eventually be awarded. Similarly, in the case of multiple, decoupled subnetworks corresponding to tasks being done by different agents, if G_W is an agent controlling the leftover time-points in the relativizing set $\mathcal{T}_W$, then G_W cannot be allowed to add constraints to the time-points in $\mathcal{T}_W$ if doing so would cause any additional constrainedness in the decoupled subnetworks.

The standard approach for determining the bounds on how much constraints in an STN can be tightened (cf. Theorem 4.33) does not accommodate the extra requirement that the tightening of constraints not impose any additional constrainedness on the decoupled subnetwork(s). This chapter presents *lambda bounds* that an agent can use to accommodate this extra requirement. Although the lambda bounds are more complicated than those in the standard case (which derive from the negative-transpose entries in the distance matrix), they can be computed in polynomial time. The lambda bounds are proven to be necessary and sufficient for the purposes described.

Hierarchical partial temporal decoupling. To allow even greater flexibility, a hierarchy of partially decoupled subnetworks may be constructed by recursively applying the relative temporal decoupling algorithm to a given network. This can be used to allow agents to have varying degrees of flexibility and dependence in a group activity. This chapter shows how to construct such hierarchies. In addition, it shows that the lambda bounds described above are applicable to the hierarchical situation as well.

The rest of this chapter is organized as follows. Section 5.2 formally defines the basic Temporal Decoupling Problem in the case of two subnetworks. Section 5.3 treats the Allocative TDP, also restricted to the case of two subnetworks, and shows how an algorithm for solving the Allocative TDP can be applied to the Auction-Participation-Context-Generation Problem. Section 5.4 extends the treatment of the basic Temporal Decoupling Problem to the case of arbitrarily many subnetworks and shows how an algorithm for the General TDP may be applied to the Post-Auction-Coordination Problem. Section 5.5 formally addresses the problem of decoupling subnetworks relative to some leftover set of time-points. That section formally derives the *lambda bounds* needed by agents controlling the leftover time-points in a relative temporal decoupling and discusses applications of the Relative TDP algorithm and the lambda bounds to the Temporal-Constraint-Generation and Post-Auction-Coordination Problems. Section 5.6 discusses related work.

5.2 The Temporal Decoupling Problem (Case $n = 2$)

This section presents the temporal decoupling problem in the case of two subnetworks (which shall be called $\mathcal{S}_X$ and $\mathcal{S}_Y$). (The general case of arbitrarily many subnetworks is treated in Section 5.4.) Section 5.2.1 formally defines the temporal decoupling problem. Section 5.2.2 presents theorems specifying necessary and sufficient characterizations

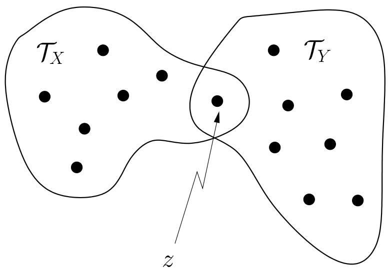


Figure 5.1: A Sample z-Partition

of solutions to instances of the TDP. Sections 5.2.3 and 5.2.4 present sound and complete algorithms for solving instances of the TDP, which are then evaluated experimentally in Section 5.2.5. Section 5.2.6 presents a formulation of the TDP in which solutions are restricted to locally optimal decouplings, called *minimal temporal decouplings*. Section 5.2.7 then recasts the basic TDP as an optimization problem in which a decoupling is sought that maximizes some measure of goodness.

5.2.1 Formal Definition of the TDP

We begin by defining a *z-partition* of a set of time-points. A z-partition is similar to an ordinary partition, except that every subset in the z-partition is required to include the special zero time-point z (since every STN includes z).

Definition 5.1 (z-Partition) *Let $\mathcal{T}$, $\mathcal{T}_X$ and $\mathcal{T}_Y$ be sets of time-point variables. We say that $\mathcal{T}_X$ and $\mathcal{T}_Y$* z-partition *$\mathcal{T}$ if:*

- $\mathcal{T}_X \cap \mathcal{T}_Y = \{z\}$ *and*
- $\mathcal{T}_X \cup \mathcal{T}_Y = \mathcal{T}$.

Figure 5.1 shows a sample z-partition of a set of time-point variables.

Definition 5.2 (Temporal Decoupling) *A* temporal decoupling *of the STN $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ is a pair of STNs, $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$, such that:*

- *$\mathcal{S}_X$ and $\mathcal{S}_Y$ are consistent;*
- *$\mathcal{T}_X$ 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 $\mathcal{S}$.*

To highlight the particular z-partition, we may say that $\mathcal{S}_X$ and $\mathcal{S}_Y$ are a temporal decoupling of $\mathcal{S}$ using *the z-partition $(\mathcal{T}_X, \mathcal{T}_Y)$.*

Lemma 5.3 *For $\mathcal{S}_X$ and $\mathcal{S}_Y$ to be a temporal decoupling of $\mathcal{S}$, it is necessary for $\mathcal{S}$ to be consistent.*

Proof If $\mathcal{S}_X$ and $\mathcal{S}_Y$ are a temporal decoupling of $\mathcal{S}$, then, by Definition 5.2, both $\mathcal{S}_X$ and $\mathcal{S}_Y$ are consistent. Thus, each has at least one solution. By the Mergeable Solutions Property, the merging of any such solutions yields a solution to $\mathcal{S}$. Thus, $\mathcal{S}$ is consistent. ∎

Lemma 5.4 *If $\mathcal{S}_X$ and $\mathcal{S}_Y$ are a temporal decoupling of $(\mathcal{T}, \mathcal{C} \cup \mathcal{C}')$, then $\mathcal{S}_X$ and $\mathcal{S}_Y$ are also a temporal decoupling of $(\mathcal{T}, \mathcal{C})$.*

Proof Any solution for $(\mathcal{T}, \mathcal{C} \cup \mathcal{C}')$ is necessarily a solution for $(\mathcal{T}, \mathcal{C})$. ∎

Lemma 5.5 *If $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$ are a temporal decoupling of $(\mathcal{T}, \mathcal{C})$, then $\mathcal{S}_X$ and $\mathcal{S}_Y$ are also a temporal decoupling of $(\mathcal{T}, \mathcal{C} \cup \mathcal{C}_X \cup \mathcal{C}_Y)$.*

Proof Let $\mathcal{X}$ and $\mathcal{Y}$ be arbitrary solutions for $\mathcal{S}_X$ and $\mathcal{S}_Y$, respectively. Thus, the variable assignments in $\mathcal{X}$ and $\mathcal{Y}$ satisfy the constraints in $\mathcal{C}_X$ and $\mathcal{C}_Y$, respectively. If $\mathcal{S}_X$ and $\mathcal{S}_Y$ are a temporal decoupling of $(\mathcal{T}, \mathcal{C})$, then $\mathcal{X} \cup \mathcal{Y}$ must be a solution for $(\mathcal{T}, \mathcal{C})$. Thus, the variable assignments in $\mathcal{X} \cup \mathcal{Y}$ satisfy the constraints in $\mathcal{C}$. Since the constraints in $\mathcal{C}_X$ do not constrain the variables in $\mathcal{T}_Y$, and the constraints in $\mathcal{C}_Y$ do not constrain the variables in $\mathcal{T}_X$, the variable assignments in $\mathcal{X} \cup \mathcal{Y}$ satisfy the constraints in $\mathcal{C} \cup \mathcal{C}_X \cup \mathcal{C}_Y$. In other words, $\mathcal{X} \cup \mathcal{Y}$ is a solution for $(\mathcal{T}, \mathcal{C} \cup \mathcal{C}_X \cup \mathcal{C}_Y)$. Since the solutions $\mathcal{X}$ and $\mathcal{Y}$ were arbitrary, $\mathcal{S}_X$ and $\mathcal{S}_Y$ are a temporal decoupling of $(\mathcal{T}, \mathcal{C} \cup \mathcal{C}_X \cup \mathcal{C}_Y)$. ∎

Note. In some applications (e.g., in the temporal-constraint-generation problem discussed in Section 5.5.6), it is convenient to replace the Mergeable Solutions Property with an equivalent property called the *Mergeable Constraints Property*. The Mergeable Constraints Property says that no matter how the agents subsequently constrain their decoupled subnetworks, the global network $\mathcal{S}$ will remain consistent.

Definition 5.6 (Mergeable Constraints Property) *Let $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$ be consistent STNs such that $\mathcal{T}_X$ and $\mathcal{T}_Y$ z-partition a set of time-points $\mathcal{T}$. We say that $\mathcal{S}_X$ and $\mathcal{S}_Y$ have the* Mergeable Constraints Property *in the STN $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ if for any sets of constraints $\Delta\mathcal{C}_X$ and $\Delta\mathcal{C}_Y$ over $\mathcal{T}_X$ and $\mathcal{T}_Y$, respectively, if $\Delta\mathcal{C}_X$ and $\Delta\mathcal{C}_Y$ are consistent with $\mathcal{S}_X$ and $\mathcal{S}_Y$, respectively, then their union $\Delta\mathcal{C}_X \cup \Delta\mathcal{C}_Y$ is consistent with $\mathcal{S}$.*

Theorem 5.7 (MSP $\equiv$ MCP) *Let $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$ be consistent STNs such that $\mathcal{T}_X$ and $\mathcal{T}_Y$ z-partition a set of time-points $\mathcal{T}$. $\mathcal{S}_X$ and $\mathcal{S}_Y$ have the Mergeable Solutions Property (MSP) in the STN $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ if and only if they have the Mergeable Constraints Property (MCP) in $\mathcal{S}$.*

Proof ($\Rightarrow$) Suppose $\mathcal{S}_X$ and $\mathcal{S}_Y$ have the Mergeable Solutions Property in $\mathcal{S}$. Let $\Delta\mathcal{C}_X$ and $\Delta\mathcal{C}_Y$ be arbitrary sets of constraints over the time-points in $\mathcal{T}_X$ and $\mathcal{T}_Y$, respectively, such that $\Delta\mathcal{C}_X$ and $\Delta\mathcal{C}_Y$ are consistent with $\mathcal{S}_X$ and $\mathcal{S}_Y$, respectively. By Definition 4.9, $\mathcal{S}_X^+ = (\mathcal{T}_X, \mathcal{C}_X \cup \Delta\mathcal{C}_X)$ and $\mathcal{S}_Y^+ = (\mathcal{T}_Y, \mathcal{C}_Y \cup \Delta\mathcal{C}_Y)$ are consistent STNs. Let $\mathcal{X}$ and $\mathcal{Y}$

be arbitrary solutions for $\mathcal{S}_X^+$ and $\mathcal{S}_Y^+$, respectively. Since the constraints in $\Delta\mathcal{C}_X$ do not constrain the variables in $\mathcal{T}_Y$ and, similarly, the constraints in $\Delta\mathcal{C}_Y$ do not constrain the variables in $\mathcal{T}_X$, we have that the variable assignments in $\mathcal{X} \cup \mathcal{Y}$ satisfy all the constraints in $\Delta\mathcal{C}_X \cup \Delta\mathcal{C}_Y$.

Now, since $\mathcal{X}$ and $\mathcal{Y}$ are (also) solutions for $\mathcal{S}_X$ and $\mathcal{S}_Y$, respectively, the Mergeable Solutions Property gives us that $\mathcal{X} \cup \mathcal{Y}$ must be a solution for $\mathcal{S}$, whence the variable assignments in $\mathcal{X} \cup \mathcal{Y}$ also satisfy all the constraints in $\mathcal{C}$. Thus, the variable assignments in $\mathcal{X} \cup \mathcal{Y}$ satisfy all the constraints in $\mathcal{C} \cup \Delta\mathcal{C}_X \cup \Delta\mathcal{C}_Y$, which implies that $(\mathcal{T}, \mathcal{C} \cup \Delta\mathcal{C}_X \cup \Delta\mathcal{C}_Y)$ is a consistent STN. Thus, by Definition 4.9, the constraints in $\Delta\mathcal{C}_X \cup \Delta\mathcal{C}_Y$ are consistent with $\mathcal{S} = (\mathcal{T}, \mathcal{C})$.

($\Leftarrow$) Suppose $\mathcal{S}_X$ and $\mathcal{S}_Y$ have the Mergeable Constraints Property in $\mathcal{S}$. Let $\mathcal{X}$ and $\mathcal{Y}$ be arbitrary solutions for $\mathcal{S}_X$ and $\mathcal{S}_Y$, respectively. These solutions may be represented as sets of constraints $\Delta\mathcal{C}_X$ and $\Delta\mathcal{C}_Y$, respectively, by transforming each variable assignment $t_i = w_i$ into a pair of constraints: $t_i - z \leq w_i$ and $z - t_i \leq -w_i$. Thus, by the Mergeable Constraints Property, the union $\Delta\mathcal{C}_X \cup \Delta\mathcal{C}_Y$ is consistent with $\mathcal{S}$. But then the variable assignments in $\mathcal{X} \cup \mathcal{Y}$ constitute a solution for $\mathcal{S}$. ■

The following definition of the Temporal Decoupling Problem does not include a metric of solution *quality*. Section 5.2.7 presents a formulation of the TDP as an optimization problem in terms of such a metric.

Definition 5.8 (The Temporal Decoupling Problem) *Given an STN $\mathcal{S}$ whose time-points are z-partitioned by $\mathcal{T}_X$ and $\mathcal{T}_Y$, find sets of constraints $\mathcal{C}_X$ and $\mathcal{C}_Y$ such that the STNs $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$ temporally decouple $\mathcal{S}$.*

Note. Upon finding sets $\mathcal{C}_X$ and $\mathcal{C}_Y$ such that $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$ temporally decouple $\mathcal{S} = (\mathcal{T}, \mathcal{C})$, it is typically desirable to add the constraint sets $\mathcal{C}_X$ and $\mathcal{C}_Y$ to $\mathcal{S}$. After all, if the time-points in $\mathcal{T}_X$ and $\mathcal{T}_Y$ are in fact constrained by $\mathcal{C}_X$ and $\mathcal{C}_Y$, there is nothing to gain from failing to represent this information in $\mathcal{S}$. Lemma 5.5 shows that if $\mathcal{S}_X$ and $\mathcal{S}_Y$ temporally decouple $(\mathcal{T}, \mathcal{C})$, then they also temporally decouple $(\mathcal{T}, \mathcal{C} \cup \mathcal{C}_X \cup \mathcal{C}_Y)$.

Lemma 5.9 *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 are z-partitioned by $\mathcal{T}_X$ and $\mathcal{T}_Y$. Let $\mathcal{X}$ be an arbitrary solution for $\mathcal{S}$. Then the following specifies a temporal decoupling of $\mathcal{S}$ using $\mathcal{T}_X$ and $\mathcal{T}_Y$ in which each time-point in $\mathcal{S}$ is fixed to its value in the solution $\mathcal{X}$.

$$\begin{aligned}
\mathcal{C}_X &= \{\, (x - z \leq v) : (x = v) \in \mathcal{X}, x \in \mathcal{T}_X \,\} \cup \{\, (z - x \leq -v) : (x = v) \in \mathcal{X}, x \in \mathcal{T}_X \,\} \\
\mathcal{C}_Y &= \{\, (y - z \leq w) : (y = w) \in \mathcal{X}, y \in \mathcal{T}_Y \,\} \cup \{\, (z - y \leq -w) : (y = w) \in \mathcal{X}, y \in \mathcal{T}_Y \,\}
\end{aligned}$$

■

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). Theorems 5.10 and 5.12 below provide necessary and sufficient characterizations of solutions to the TDP that point the way to TDP algorithms that yield more flexible decoupled subnetworks.

5.2.2 Necessary and Sufficient Characterizations of TDP Solutions

This section presents theorems specifying necessary and sufficient conditions for a temporal decoupling. The conditions are expressed in terms of inequalities involving the distance matrices for the subnetworks $\mathcal{S}_X$ and $\mathcal{S}_Y$, and for the global network $\mathcal{S}$. Each entry in the distance matrix for $\mathcal{S}$ serves as an upper bound for various expressions involving the distance matrices of the subnetworks. Thus, a wide range of TDP solutions is possible. In contrast, Section 5.2.6 defines a *minimal* temporal decoupling which is optimal in the sense that any non-trivial weakening of an intra-subnetwork constraint (i.e., a constraint in $\mathcal{C}_X$ or $\mathcal{C}_Y$) would ruin the decoupling.

Theorem 5.10 (Necessary Conditions) *If 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 $\mathcal{S} = (\mathcal{T}, \mathcal{C})$, then the following properties must hold:*

(Property 1_X) $\mathcal{D}_X(x_i, x_j) \leq \mathcal{D}(x_i, x_j)$, *for all* $x_i, x_j \in \mathcal{T}_X$*;*

(Property 1_Y) $\mathcal{D}_Y(y_i, y_j) \leq \mathcal{D}(y_i, y_j)$, *for all* $y_i, y_j \in \mathcal{T}_Y$*;*

(Property 2_{XY}) $\mathcal{D}_X(x, z) + \mathcal{D}_Y(z, y) \leq \mathcal{D}(x, y)$, *for all* $x \in \mathcal{T}_X$ *and* $y \in \mathcal{T}_Y$*; and*

(Property 2_{YX}) $\mathcal{D}_Y(y, z) + \mathcal{D}_X(z, x) \leq \mathcal{D}(y, x)$, *for all* $x \in \mathcal{T}_X$ *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.

If, in addition, Properties 1_X and 1_Y hold with equality in all instances (i.e., $\mathcal{S}_X \equiv \mathcal{S}_d|_{\mathcal{T}_X}$ and $\mathcal{S}_Y \equiv \mathcal{S}_d|_{\mathcal{T}_Y}$), then Properties 2_{XY} and 2_{YX} also hold with equality in all instances.

Properties 1_X and 1_Y are illustrated in Figure 5.2. Properties 2_{XY} and 2_{YX} are illustrated in Figures 5.3 and 5.4.

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 $\mathcal{S} = (\mathcal{T}, \mathcal{C})$. By definition of a temporal decoupling, $\mathcal{S}_X$ and $\mathcal{S}_Y$ are both consistent.

(Property 1_X) Let $x_i, x_j \in \mathcal{T}_X$ be arbitrary. By Theorem 4.33, 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 $\mathcal{Y}$ yields a solution for $\mathcal{S}$. In that solution, $x_j - x_i = \mathcal{D}_X(x_i, x_j)$. In addition, being a solution for $\mathcal{S}$ implies that $x_j - x_i \leq \mathcal{D}(x_i, x_j)$. Thus,

$$\mathcal{D}_X(x_i, x_j) \;=\; x_j - x_i \;\leq\; \mathcal{D}(x_i, x_j).$$

(Property 2_{XY}) 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 $y - z = \mathcal{D}_Y(z, y)$. Since $\mathcal{S}_X$ and $\mathcal{S}_Y$ are a temporal decoupling of $\mathcal{S}$, merging the solutions $\mathcal{X}$ and $\mathcal{Y}$ yields a solution for $\mathcal{S}$. In this solution,

$$y - x \;=\; (z - x) + (y - z) \;=\; \mathcal{D}_X(x, z) + \mathcal{D}_Y(z, y).$$

In addition, being a solution for $\mathcal{S}$ implies that $y - x \leq \mathcal{D}(x, y)$. Thus,

$$\mathcal{D}_X(x, z) + \mathcal{D}_Y(z, y) \;=\; y - x \;\leq\; \mathcal{D}(x, y).$$

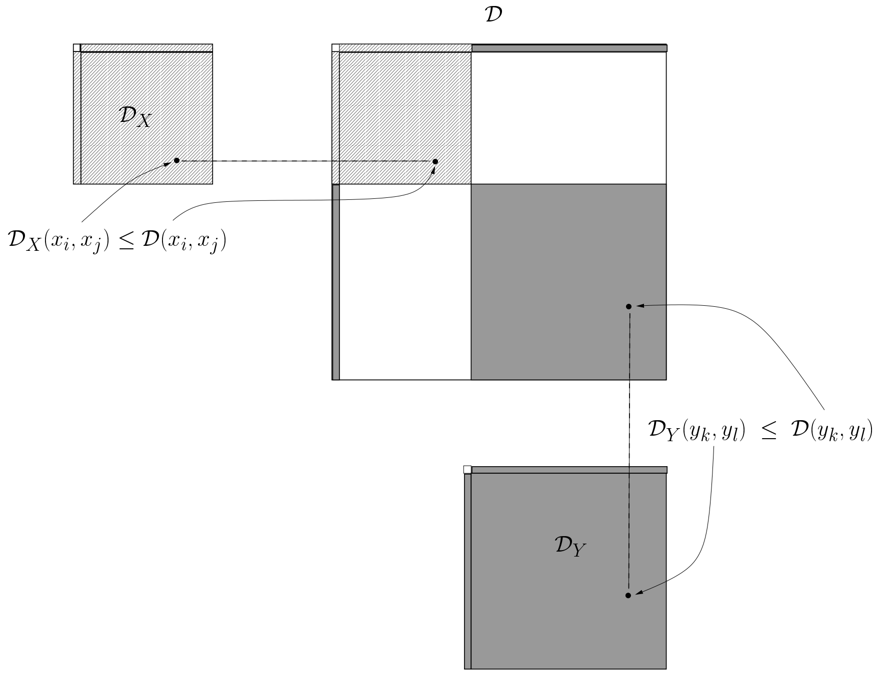


Figure 5.2: Illustration of Properties 1_X and 1_Y from Theorem 5.10

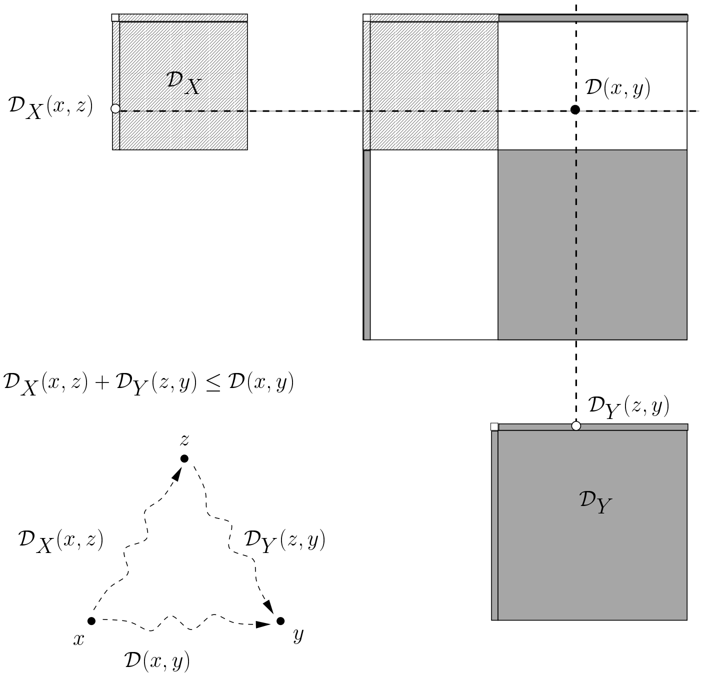


Figure 5.3: Illustration of Property 2_{XY} from Theorem 5.10

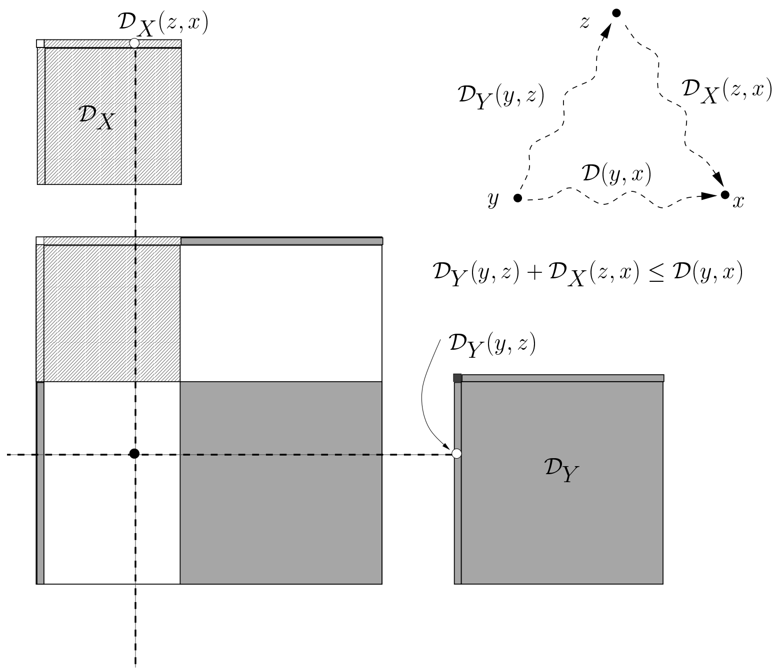


Figure 5.4: Illustration of Property 2_{YX} from Theorem 5.10

Properties 1_Y and 2_{YX} are handled analogously.

If Properties 1_X and 1_Y hold with equality in all instances, then Property 2_{XY} becomes:

$$\mathcal{D}(x,z) + \mathcal{D}(z,y) \;\leq\; \mathcal{D}(x,y), \text{ for all } x \in \mathcal{T}_X \text{ and } y \in \mathcal{T}_Y.$$

However, by the triangle inequality:

$$\mathcal{D}(x,z) + \mathcal{D}(z,y) \geq \mathcal{D}(x,y), \text{ for all } x \in \mathcal{T}_X \text{ and } y \in \mathcal{T}_Y.$$

Thus, equality obtains in Property 2_{XY}. Property 2_{YX} is similarly affected.

Finally, if Property 1_X holds with equality in all instances then $\mathcal{D}_X = \mathcal{D}|_{\mathcal{T}_X}$. Since $\mathcal{D} = \mathcal{D}_d$ (by Proposition 4.29), $\mathcal{D}|_{\mathcal{T}_X} = \mathcal{D}_d|_{\mathcal{T}_X}$. Thus, $\mathcal{D}_X = \mathcal{D}_d|_{\mathcal{T}_X}$. Since $\mathcal{D}_d|_{\mathcal{T}_X}$ is the distance matrix for the d-subnetwork of $\mathcal{S}$ with respect to $\mathcal{T}_X$ (Lemma 4.42), $\mathcal{S}_X \equiv \mathcal{S}_d|_{\mathcal{T}_X}$. Similarly, $\mathcal{S}_Y \equiv \mathcal{S}_d|_{\mathcal{T}_Y}$. ■

Corollary 5.11 *If $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$ temporally decouple $(\mathcal{T}, \mathcal{C})$, then $\mathcal{S}_X$ and $\mathcal{S}_Y$ temporally decouple $(\mathcal{T}, \mathcal{C} \cup \mathcal{C}_X \cup \mathcal{C}_Y)$ such that Properties 1_X and 1_Y (and hence Properties 2_{XY} and 2_{YX}) from Theorem 5.10 hold with equality in all instances.*

Proof Let $\mathcal{D}^+$ be the distance matrix for the STN, $\mathcal{S}^+ = (\mathcal{T}, \mathcal{C} \cup \mathcal{C}_X \cup \mathcal{C}_Y)$. By Lemma 5.5, if $\mathcal{S}_X$ and $\mathcal{S}_Y$ temporally decouple $(\mathcal{T}, \mathcal{C})$, then they also temporally decouple $\mathcal{S}^+$. Thus, it suffices to show that Properties 1_X and 1_Y hold with equality in all instances for this temporal decoupling.

Let $x_i, x_j \in \mathcal{T}_X$ be arbitrary. By Theorem 4.33, there is a solution for $\mathcal{S}^+$ in which $x_j - x_i = \mathcal{D}^+(x_i, x_j)$. However, any solution to $\mathcal{S}^+$ must satisfy the constraints in $\mathcal{C}_X$. Thus, $x_j - x_i \leq \mathcal{D}_X$. Thus,

$$\mathcal{D}^+(x_i, x_j) \;=\; x_j - x_i \;\leq\; \mathcal{D}_X(x_i, x_j).$$

Since the opposite inequality is guaranteed by Property 1_X of a temporal decoupling of $\mathcal{S}^+$, we have that $\mathcal{D}^+(x_i, x_j) = \mathcal{D}_X(x_i, x_j)$. Since x_i and x_j were arbitrary in $\mathcal{T}_X$, Property 1_X holds with equality in all instances for this decoupling. Similarly remarks apply to Property 1_Y. ■

Theorem 5.12 (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_X, 1_Y, 2_{XY} and 2_{YX} of Theorem 5.10 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 5.2) that is non-trivial to verify in this setting is the Mergeable Solutions Property. Let

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

be arbitrary solutions for $\mathcal{S}_X$ and $\mathcal{S}_Y$, respectively. To show that $\mathcal{X} \cup \mathcal{Y}$ is a solution for $\mathcal{S}$, it suffices to show that the variable assignments in $\mathcal{X} \cup \mathcal{Y}$ satisfy all of the constraints in $\mathcal{C}$. Let E: $\tau_j - \tau_i \leq \delta$ be an arbitrary constraint in $\mathcal{C}$, where τ_i and τ_j are meta-variables ranging over the time-point variables in $\mathcal{T} = \mathcal{T}_X \cup \mathcal{T}_Y$.

Case 1: $\tau_i, \tau_j \in \mathcal{T}_X$. Thus, $\tau_i = x_p$ and $\tau_j = x_q$ for some x_p and x_q in $\mathcal{T}_X$. (These equalities do not represent temporal constraints; they simply state which time-point variables the meta-variables τ_i and τ_j represent.) $\mathcal{X}$ being a solution for $\mathcal{S}_X$ implies that $v_q - v_p \leq \mathcal{D}_X(x_p, x_q)$. Since Property 1_X holds, $\mathcal{D}_X(x_p, x_q) \leq \mathcal{D}(x_p, x_q)$. Finally, since E is a constraint in $\mathcal{C}$, we have that $\mathcal{D}(x_p, x_q) \leq \delta$. Thus,

$$v_q - v_p \;\leq\; \mathcal{D}_X(x_p, x_q) \;\leq\; \mathcal{D}(x_p, x_q) \;\leq\; \delta$$

(i.e., the constraint E is satisfied by $x_p = v_p$ and $x_q = v_q$).

Case 2: $\tau_i, \tau_j \in \mathcal{T}_Y$. Handled analogously to Case 1.

Case 3: $\tau_i \in \mathcal{T}_X$ and $\tau_j \in \mathcal{T}_Y$. Thus, $\tau_i = x_p$ and $\tau_j = y_q$ for some $x_p \in \mathcal{T}_X$ and $y_q \in \mathcal{T}_Y$. Since $x_p = v_p$ is part of a solution for $\mathcal{S}_X$, we have that $0 - v_p \leq \mathcal{D}_X(x_p, z)$. Similarly, $w_q - z \leq \mathcal{D}_Y(z, y_q)$. Thus,

$$w_q - v_p \;=\; (z - v_p) + (w_q - z) \;\leq\; \mathcal{D}_X(x_p, z) + \mathcal{D}_Y(z, y_q).$$

Since Property 2_{XY} holds, $\mathcal{D}_X(x_p, z) + \mathcal{D}_Y(z, y_q) \leq \mathcal{D}(x_p, y_q)$. Finally, since E is a constraint in $\mathcal{C}$, we have that $\mathcal{D}(x_p, y_q) \leq \delta$. Thus,

$$w_q - v_p \;\leq\; \mathcal{D}_X(x_p, z) + \mathcal{D}_Y(z, y_q) \;\leq\; \mathcal{D}(x_p, y_q) \;\leq \delta$$

(i.e., the constraint E is satisfied by $x_p = v_p$ and $y_q = w_q$).

Case 4: $\tau_i \in \mathcal{T}_Y$ and $\tau_j \in \mathcal{T}_X$. Handled analogously to Case 3.

Since the constraint E was arbitrary in $\mathcal{C}$, $\mathcal{X} \cup \mathcal{Y}$ is a solution for $\mathcal{S}$. ■

5.2.3 Toward a TDP Algorithm

Properties 2_{XY} and 2_{YX} of Theorem 5.10 specify numerous inequalities that must be satisfied by any temporal decoupling. The analysis in this section, culminating in Lemma 5.18, shows that it suffices to satisfy only the inequalities pertaining to what are called *tight, proper xy-edges*. Based on this analysis, the TDP algorithm in Section 5.2.4 operates exclusively on tight, proper xy-edges.

Definition 5.13 (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:*

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

If, in addition, neither t_i nor t_j is the zero time-point variable, 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.*

Given some proper xy-edge, the *zero-path shortfall* associated with that edge specifies a lower bound on the amount by which the corresponding inequality in Property 2_{XY} or 2_{YX} in Theorem 5.10 *fails* to hold. For a *tight*, proper xy-edge, the zero-path shortfall specifies the *exact* amount by which the corresponding inequality fails to hold. The TDP algorithm in the next section operates on tight, proper xy-edges until the zero-path shortfall associated with each such edge is equal to zero (and hence the corresponding inequality from Property 2_{XY} or 2_{YX} *holds*).

Definition 5.14 (Zero-Path Shortfall) *The* zero-path shortfall *(ZPS) associated with the proper xy-edge, E: $(t_j - t_i \leq \delta)$, is defined by:*

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

Lemma 5.15 *If E is a tight, proper xy-edge, then $\mathrm{ZPS}(E) \geq 0$.*

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

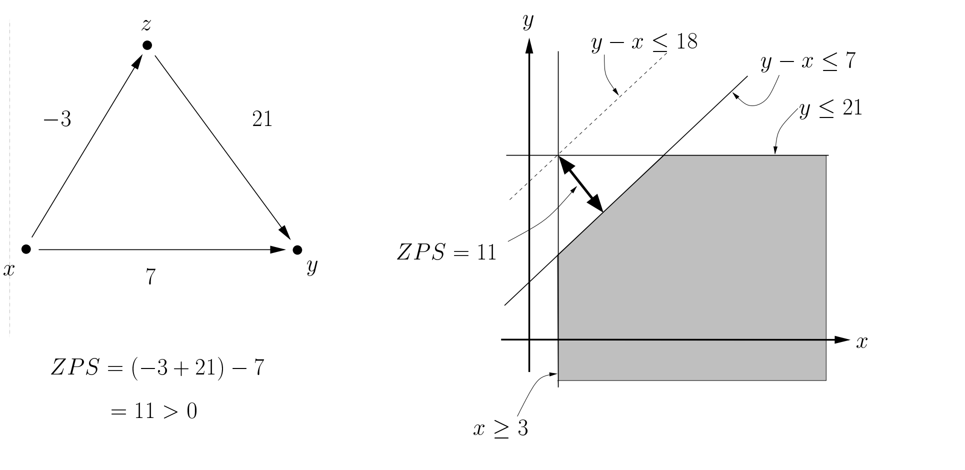


Figure 5.5: An xy-edge that is *not* dominated by a path through zero

$$\delta \;=\; \mathcal{D}(t_i, t_j) \;\leq\; \mathcal{D}(t_i, z) + \mathcal{D}(z, t_j).$$

Thus, ZPS$(E) \geq 0$. ■

Definition 5.16 (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.[1]

Figure 5.5 shows a tight, proper xy-edge, $y - x \leq 7$, that is *not* dominated by a path through zero (i.e., it has a *positive* ZPS value) along with an xy-plane representation of the involved constraints (in which the set of solutions is shaded). The ZPS value for a given edge $y - x \leq \delta$ may be reduced by strengthening one or both of the xz- and zy-edges. Figure 5.6 shows the xy-edge from Figure 5.5, but with the xz- and zy-edges strengthened such that the xy-edge is now dominated by a path through zero. Notice that in the xy-plane representation for this situation, the dominated xy-constraint no longer plays a role in shaping the space of solutions.

Lemma 5.17 *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.*

[1] As will be seen below, if each tight, proper xy-edge is dominated by a path through zero, then $\mathcal{S}_X$ and $\mathcal{S}_Y$ will necessarily form a temporal decoupling of $\mathcal{S}$. If we ignore these dominated (and hence redundant) constraints in $\mathcal{S}$, we have that z is a *separation vertex* of $\mathcal{S}$, according to the definition given by Dechter et al. (1991). Whereas Dechter et al. focus on finding non-separable components of a given STN, our focus is on finding a set of constraints to impose on an STN to achieve a temporal decoupling of that STN into two (or more) subnetworks. See Section 5.6.1.

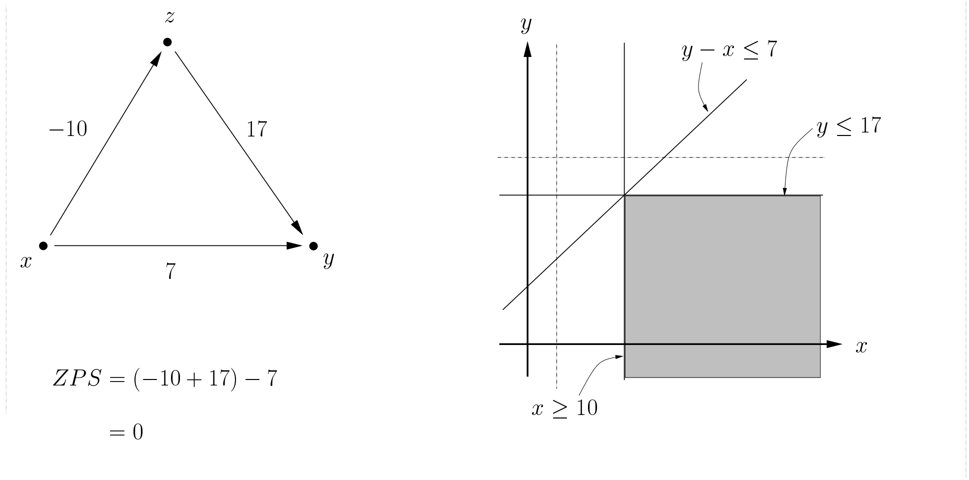


Figure 5.6: An xy-edge that *is* dominated by a path through zero

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. ■

Lemma 5.18 shows that it suffices to consider only tight, proper xy-edges when seeking a solution to an instance of the TDP.

Lemma 5.18 *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* $\mathcal{S}$*, 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* $\mathcal{S}$.

Proof Suppose that the inequality $\mathcal{D}(t_p, z) + \mathcal{D}(z, t_q) \leq \delta$ holds for every tight, proper xy-*edge* $(t_q - t_p \leq \delta)$ in $\mathcal{S}$. Let (t_i, t_j) be an arbitrary xy-*pair* in $\mathcal{S}$. It suffices to show that the inequality $\mathcal{D}(t_i, z) + \mathcal{D}(z, t_j) \leq \mathcal{D}(t_i, t_j)$ 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 *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, by Proposition 4.20).

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 (cf. Proposition 4.20); hence $\delta_{xy} = \mathcal{D}(x, y)$. Thus, the Lemma's premise, applied to the tight, proper xy-edge E_{xy}, implies that: $\mathcal{D}(x, z) + \mathcal{D}(z, y) \leq \delta_{xy}$. However, the Triangle Inequality provides the opposite inequality: $\delta_{xy} = \mathcal{D}(x, y) \leq \mathcal{D}(x, z) + \mathcal{D}(z, y)$. 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

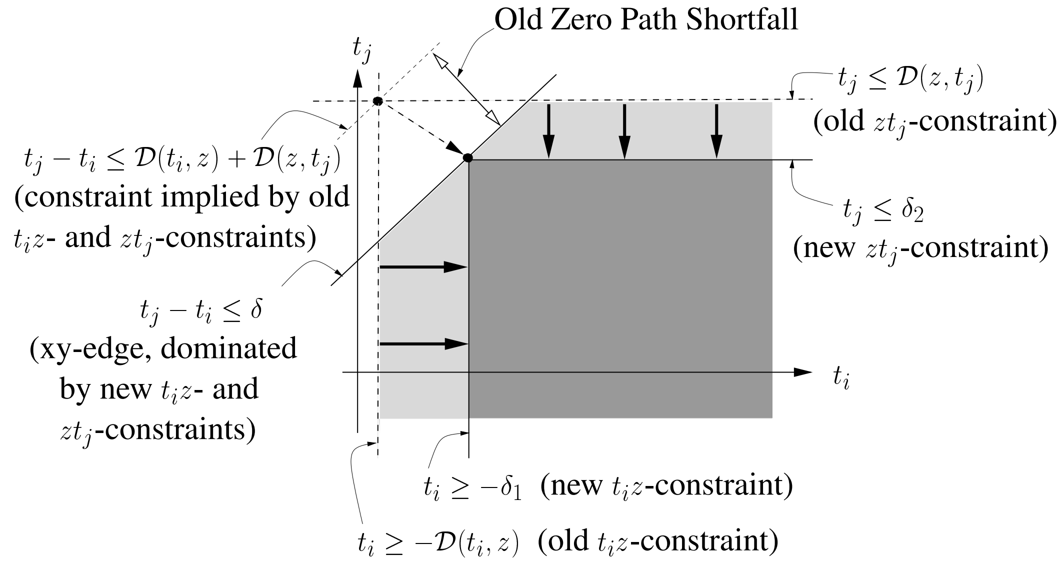


Figure 5.7: Reducing the zero-path shortfall for an xy-edge

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. ■

5.2.4 Algorithms for Solving the TDP

This section presents a family of sound and complete algorithms for solving the Temporal Decoupling Problem. The basic TDP algorithm, which is directly motivated by Theorem 5.12 and Lemma 5.18, is sound, but not complete: because it is not guaranteed to terminate. However, Theorems 5.26 and 5.28 (below) specify a variety of simple strategies for strengthening the basic algorithm to ensure that it terminates and, hence, that it is complete.

The Basic TDP Algorithm

The main tool of the basic 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 5.7 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 basic TDP algorithm is given in pseudo-code in Figure 5.8. It takes as input an STN $\mathcal{S}$ whose time-points are z-partitioned by the sets $\mathcal{T}_X$ and $\mathcal{T}_Y$. During the course of the algorithm, constraints are added to $\mathcal{S}$ and the distance matrix $\mathcal{D}$ is updated accordingly.

At Step 1, the algorithm checks whether $\mathcal{S}$ is consistent. If $\mathcal{S}$ is inconsistent, the algorithm returns `NIL` and halts because, by Lemma 5.3, only consistent STNs can be decoupled. Otherwise, the algorithm initializes $\mathcal{E}$ to the set of tight, proper xy-edges in $\mathcal{S}$,

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 $E = (t_j - t_i \leq \delta)$ from $\mathcal{E}$ 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 $\mathcal{D}$ accordingly.

(5) Go to Step 2.

(6) Return:

$$\begin{array}{rcccl} \mathcal{C}_X & = & \mathcal{C}_d|_{\mathcal{T}_X} & = & \{(t_j - t_i \leq \mathcal{D}(t_i,t_j)) : t_i,t_j \in \mathcal{T}_X\}; \\ \mathcal{C}_Y & = & \mathcal{C}_d|_{\mathcal{T}_Y} & = & \{(t_j - t_i \leq \mathcal{D}(t_i,t_j)) : t_i,t_j \in \mathcal{T}_Y\}, \end{array}$$

where $\mathcal{C}$ includes all constraints added in passes of Step 4, and $\mathcal{D}$ has been updated accordingly.

Figure 5.8: Pseudo-code for the basic TDP algorithm

which merely 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 5.16).

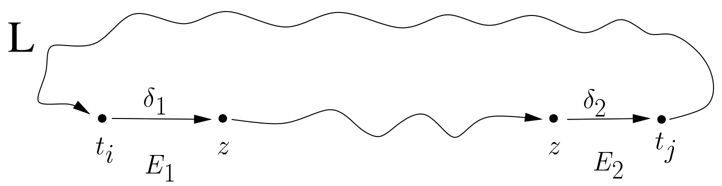
For each iteration, the algorithm does the following. In Step 2, a tight, proper xy-edge E: $(t_j - t_i \leq \delta)$ with a positive zero-path shortfall, ζ, is selected from the set $\mathcal{E}$. In Steps 3 and 4, new constraints E_1: $z - t_i \leq \delta_1$ and E_2: $t_j - z \leq \delta_2$ are added to $\mathcal{S}$. (Theorem 5.19 (below) ensures that adding E_1 and E_2 to $\mathcal{S}$ will not cause $\mathcal{S}$ to become inconsistent.) After updating the distance matrix $\mathcal{D}$ 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 (as shown in Corollary 5.20 below). Thus, the *updated* ZPS value for E, which we denote ζ^*, will be: $\zeta^* = \delta_1 + \delta_2 - \delta$. The third Step 3 constraint guarantees that ζ^* will be strictly smaller than ζ (i.e., that the ZPS value of E has been reduced).

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). Theorem 5.23 guarantees that the algorithm is sound: if the algorithm terminates at Step 6, then the constraint sets $\mathcal{C}_X$ and $\mathcal{C}_Y$ will be such that $(\mathcal{T}_X, \mathcal{C}_X)$ and $(\mathcal{T}_Y, \mathcal{C}_Y)$ are a temporal decoupling of $\mathcal{S}$.

Theorem 5.19 *Let $\mathcal{S}$ be a consistent STN; let E: $t_j - t_i \leq \delta$ be a tight, proper xy-edge whose ZPS value is positive; and let δ_1 and δ_2 be arbitrary values satisfying the requirements of Step 3 of the basic TDP algorithm. Then adding the* pair *of corresponding Step 4 constraints (i.e., E_1 and E_2 in Figure 5.8) will not threaten the consistency of $\mathcal{S}$.*

Proof Let $\zeta > 0$ be the ZPS value for the edge E. Suppose that adding the corresponding Step 4 constraints E_1 and E_2 caused $\mathcal{S}$ to become inconsistent. Then, by Theorem 4.30, there must be a loop with negative path-length in the distance graph $\mathcal{G}$. By Theorem 4.33, the first two Step 3 requirements (from Figure 5.8) imply that E_1 and E_2 are individually consistent with $\mathcal{S}$. Thus, any loop with negative path length in $\mathcal{G}$ must contain *both* E_1 and E_2. Of all such loops, let L be one that has the minimum number of edges. L is illustrated below.



Let L' be the subpath of L from t_i (at the beginning of E_1) to t_j (at the end of E_2). Then,

$$|L'| \geq \delta_1 + \mathcal{D}(z, z) + \delta_2 \geq \delta_1 + 0 + \delta_2 = \delta_1 + \delta_2.$$

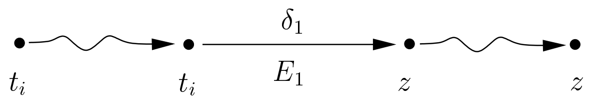
However, part of the third Step 3 requirement (from Figure 5.8) says that $\delta_1 + \delta_2 \geq \delta$. Thus, $|L'| \geq \delta$. As a result, the subpath L', which contains at least two edges, could be replaced in L by the single edge E (which has length δ), resulting in a loop L'' such that $|L''| \leq |L| < 0$. But L'' has fewer edges than L, which contradicts the choice of L. Thus, no such L exists. Thus, adding both E_1 and E_2 to $\mathcal{S}$ leaves $\mathcal{S}$ consistent. ■

Corollary 5.20 *Let $\mathcal{S}$ be a consistent STN. Let E_1: $z - t_i \leq \delta_1$ and E_2: $t_j - z \leq \delta_2$ be constraints satisfying the Step 3 requirements of the basic TDP algorithm. Let $\mathcal{S}^+$ be the STN resulting from adding E_1 and E_2 to $\mathcal{S}$. Let $\mathcal{D}^+$ be the corresponding distance matrix. Then,*

$$\mathcal{D}^+(t_i, z) = \delta_1 \quad \text{and} \quad \mathcal{D}^+(z, t_j) = \delta_2.$$

Proof Let P be a shortest path from t_i to z in $\mathcal{S}^+$. Thus, $|P| = \mathcal{D}^+(t_i, z)$. Since E_1 by itself constitutes a path from t_i to z in $\mathcal{S}^+$ with length δ_1, we have that $|P| = \mathcal{D}^+(t_i, z) \leq \delta_1$.

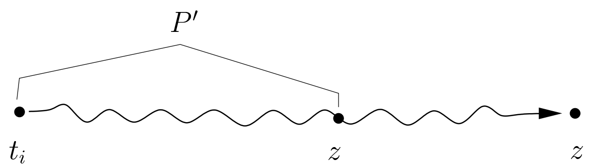
Case 1: E_1 is contained in P, as shown below.



Since any subpaths of a shortest path must themselves be shortest paths (and since $\mathcal{S}^+$ is guaranteed to be consistent by Theorem 5.19),

$$|P| \;=\; \mathcal{D}(t_i, t_i) + \delta_1 + \mathcal{D}(z, z) \;=\; 0 + \delta_1 + 0 \;=\; \delta_1.$$

Case 2: E_1 is not contained in P. Let P' be the subpath of P from the beginning of P to the first occurrence of z in P, as shown below.



(Since z appears at the end of P, the subpath P' is well-defined. Furthermore, it may be that $P' = P$.) Since P' is a subpath of a shortest path, P' must itself be a shortest path. However, P' is also a path from t_i to z. Thus, $|P'| = |P|$. Furthermore, P' does not contain either E_1 (by assumption) or E_2 (since the only occurrence of z in P' is at the end of P'). Thus, $|P'| = \mathcal{D}(t_i, z)$ (where $\mathcal{D}$ is the distance matrix for $\mathcal{S}$, not $\mathcal{S}^+$). But the first Step 3 requirement says that $\mathcal{D}(t_i, z) \geq \delta_1$. Thus, $|P| = |P'| = \mathcal{D}(t_i, z) \geq \delta_1$. Since it has already been established that $|P| \leq \delta_1$, it must be that $|P| = \delta_1$.

In either case, it has been shown that a shortest path from t_i to z in $\mathcal{S}^+$ has length δ_1; thus $\mathcal{D}^+(t_i, z) = \delta_1$. That $\mathcal{D}^+(z, t_j) = \delta_2$ follows in a similar fashion. ■

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 R by which the ZPS value for the edge under consideration is thereby reduced, as well as the fractions of this ZPS reduction corresponding to the amounts by which the t_iz- and zt_j-edges were tightened (specified by α and $1-\alpha$). Lemma 5.21 (below) specifies the relationship between the pairs of values (δ_1, δ_2) and (R, α). Lemma 5.21 is subsequently used to demonstrate various properties of the TDP algorithm (e.g., its soundness).

Lemma 5.21 *Let E be some tight, proper xy-edge $t_j - t_i \leq \delta$ whose ZPS value $\zeta = \mathrm{ZPS}(E)$ is positive. Let Ω be the set of ordered pairs (δ_1, δ_2) satisfying the Step 3 requirements of the basic TDP algorithm (recall Figure 5.8). 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:*

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

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

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 5.8) correspond to the boundaries of the region Ω in Figure 5.9. 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 Lemma 4.21,

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

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 5.9).

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

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

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

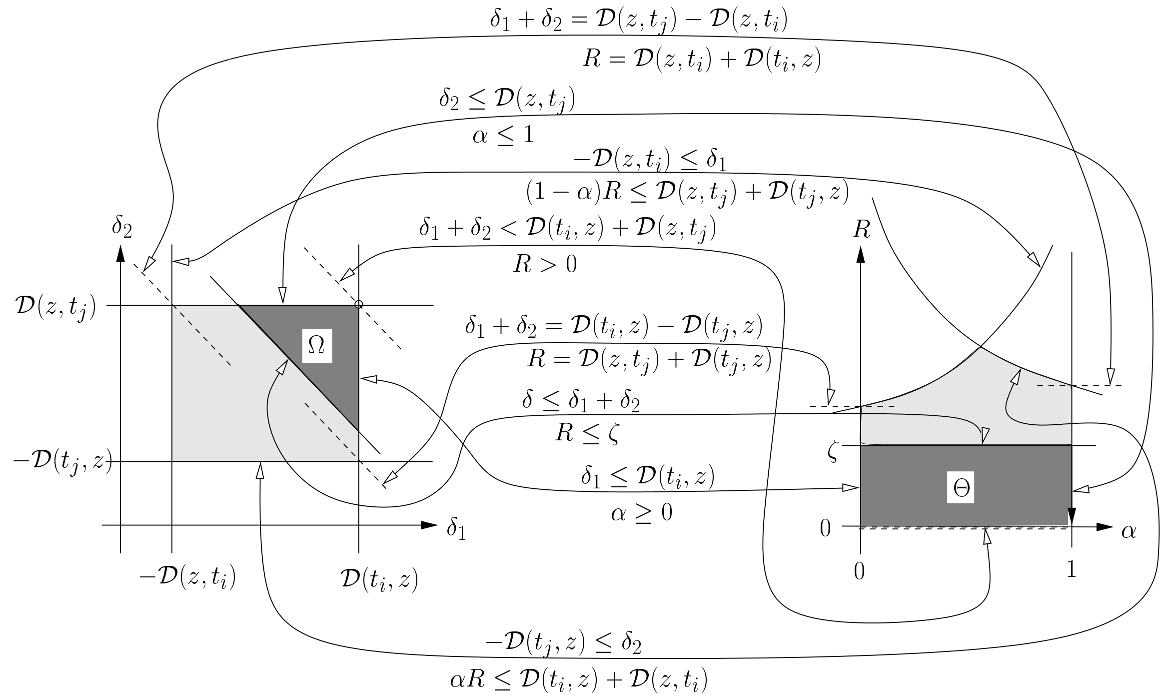


Figure 5.9: The regions Ω and Θ from Lemma 5.21

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

Hence, the fraction of the total ZPS reduction produced by strengthening the t_iz-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 zt_j-edge is precisely $(1-\alpha)$, and $\delta_2 = g(R,\alpha) = \mathcal{D}(z,t_j) - (1-\alpha)R$.

That the transformations T_1 and T_2 are inverses of one another is easily verified by showing that the following equations hold for any $(\delta_1,\delta_2)\in\Omega$ and any $(R,\alpha)\in\Theta$:

$$\left\{\begin{array}{l} f(u(\delta_1,\delta_2),v(\delta_1,\delta_2)) = \delta_1 \\ g(u(\delta_1,\delta_2),v(\delta_1,\delta_2)) = \delta_2 \end{array}\right\} \text{ and } \left\{\begin{array}{l} u(f(R,\alpha),g(R,\alpha)) = R \\ v(f(R,\alpha),g(R,\alpha)) = \alpha \end{array}\right\}.$$

Also, given any $(\delta_1,\delta_2)\in\Omega$, we may verify that $(R,\alpha) = (u(\delta_1,\delta_2),v(\delta_1,\delta_2))\in\Theta$ by showing that $u(\delta_1,\delta_2)\in(0,\zeta]$ and $v(\delta_1,\delta_2)\in[0,1]$, as follows:

- $\mathcal{D}(t_i,z)+\mathcal{D}(z,t_j) \;>\; \delta_1+\delta_2$ (Step 3 Requirement)
  $\mathcal{D}(t_i,z)+\mathcal{D}(z,t_j)-(\delta_1+\delta_2) \;>\; 0$ (Rearrange terms)
  $u(\delta_1,\delta_2) \;>\; 0$ (Defn. of $u(\delta_1,\delta_2)$)

- $\delta \;\le\; \delta_1+\delta_2$ (Step 3 Requirement)
  $-(\delta_1+\delta_2) \;\le\; -\delta$ (Rearrange terms)
  $\mathcal{D}(t_i,z)+\mathcal{D}(z,t_j)-(\delta_1+\delta_2)$
  $\quad\le\; \mathcal{D}(t_i,z)+\mathcal{D}(z,t_j)-\delta$ (Add $\mathcal{D}(t_i,z)+\mathcal{D}(z,t_j)$ to both sides)
  $u(\delta_1,\delta_2) \;\le\; \zeta$ (Defns. of $u(\delta_1,\delta_2)$ and ζ)

- $\mathcal{D}(t_i, z) \ \geq \ \delta_1$ (Step 3 Requirement)
  $\mathcal{D}(t_i, z) - \delta_1 \ \geq \ 0$ (Rearrange terms)
  $\frac{\mathcal{D}(t_i,z)-\delta_1}{\mathcal{D}(t_i,z)+\mathcal{D}(z,t_j)-(\delta_1+\delta_2)} \ \geq \ 0$ (Divide by positive number)
  $v(\delta_1, \delta_2) \ \geq \ 0$ (Defn. of $v(\delta_1, \delta_2)$)

- $\delta_2 \ \leq \ \mathcal{D}(z, t_j)$ (Step 3 Requirement)
  $\mathcal{D}(t_i, z) - \delta_1$
  $\leq \ \mathcal{D}(t_i, z) + \mathcal{D}(z, t_j) - (\delta_1 + \delta_2)$ (Add $\mathcal{D}(t_i, z) - \delta_1 - \delta_2$ to both sides)
  $\frac{\mathcal{D}(t_i,z)-\delta_1}{\mathcal{D}(t_i,z)+\mathcal{D}(z,t_j)-(\delta_1+\delta_2)} \ \leq \ 1$ (Divide by RHS, which is positive)
  $v(\delta_1, \delta_2) \ \leq \ 1$ (Defn. of $v(\delta_1, \delta_2)$)

Furthermore, since all these steps are reversible, we get that

$$(\delta_1, \delta_2) = (f(R, \alpha), g(R, \alpha)) \in \Omega, \text{ for any } (R, \alpha) \in \Theta.$$

Finally, from Theorem 4.33, the quantity $\mathcal{D}(z, t_j) + \mathcal{D}(t_j, z)$ specifies the maximum amount that the zt_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:

$\mathcal{D}(t_i, z) - \mathcal{D}(t_j, z) \ \leq \ \delta$ (Lemma 4.21)
$\mathcal{D}(t_i, z) + \mathcal{D}(z, t_j) - \delta \ \leq \ \mathcal{D}(z, t_j) + \mathcal{D}(t_j, z)$ (Add like quantities and rearrange)
$\zeta \ \leq \ \mathcal{D}(z, t_j) + \mathcal{D}(t_j, z)$ (Defn. of ζ).

Thus, the entire zero-path-shortfall for E may be eliminated by tightening only the zt_j-constraint. Similarly, the entire zero-path-shortfall may be eliminated by instead tightening only the t_iz-constraint. In Figure 5.9, 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 5.22 *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 adding the Step 4 constraints (E_1 and E_2) results in the ZPS value for E being reduced by R and the t_iz- and zt_j-edges being* tightened *by the amounts αR and $(1 - \alpha)R$, respectively.*

Theorem 5.23 (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 5.12) to show that $\mathcal{S}_X$ and $\mathcal{S}_Y$ are each consistent and that Properties 1_X, 1_Y, 2_{XY} and 2_{YX} from Theorem 5.10 hold. (It is given that the sets $\mathcal{T}_X$ and $\mathcal{T}_Y$ z-partition $\mathcal{T}$.)

Theorem 5.19 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 $\mathcal{C}_Y$, $\mathcal{S}_X$ 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 $\mathcal{T}_X$. Thus, since $\mathcal{D}^+(x_p, x_q) \leq \mathcal{D}(x_p, x_q)$, Property 1_X of Theorem 5.10 holds. Similarly, Property 1_Y also holds.

To prove that Property 2_{XY} holds, first notice that the premise of Lemma 5.18 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 5.18 holds—with respect to $\mathcal{S}^+$. Hence, from the conclusion of Lemma 5.18, 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 $\mathcal{D}^+(t_i, t_j) \leq \mathcal{D}(t_i, t_j)$, we get that $\mathcal{D}_X(t_i, z) + \mathcal{D}_Y(z, t_j) \leq \mathcal{D}(t_i, t_j)$, which is Property 2_{XY}. Similarly, Property 2_{YX} 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 basic TDP algorithm, as shown in Figure 5.8, is not guaranteed to terminate. Theorems 5.26 and 5.28, 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 5.21). 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.

Definition 5.24 (Greedy Strategy) *At each pass of Step 3, the entire zero-path shortfall for the edge under consideration is eliminated (i.e., choose: $R = \zeta$).*

Proposition 5.25 will be used in the proofs of the theorems.

Proposition 5.25 *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 Lemma 5.17, the ZPS values cannot ever increase. Thus, any progress made by the algorithm is never lost.*

Theorem 5.26 *Using the Greedy Strategy (cf. Definition 5.24), the TDP algorithm is guaranteed to terminate after at most $2|\mathcal{T}_X||\mathcal{T}_Y|$ iterations.*

Proof By Corollary 5.22, it is always possible to choose δ_1 and δ_2 in Step 3 such that the entire zero-path shortfall will be eliminated. Thus, the Greedy Strategy is feasible. By Proposition 5.25, 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 TDP algorithm is dominated by the propagation of the temporal constraints added in Step 4. This is no worse than $O(|\mathcal{T}|^3)$, by Proposition 4.22.

Although the Greedy Strategy leads to quick results, a less greedy approach can lead to decoupled networks having greater flexibility, as will be seen in the experimental evaluation section (Section 5.2.5). Below, a Less-Greedy Strategy is defined in which the ZPS value of the edge E under consideration in Step 3 is reduced by a fractional amount (unless it is already below some threshold). By doing so, the algorithm enables the processing of other edges to contribute to reducing the ZPS value for the edge E in between its visits to E, thereby reducing the possibility of overconstraining the network.

Unlike the Greedy Strategy, the Less-Greedy Strategy requires all of the initial ZPS values to be finite—which is always the case in practice.

Definition 5.27 (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. At each pass of Step 3 in the TDP algorithm, choose R (the amount by which the ZPS value ζ for the edge currently under consideration is reduced) as follows:*

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

Theorem 5.28 *If the initial ZPS values among all of the tight, proper xy-edges in a consistent STN $\mathcal{S}$ are finite, then, using the Less-Greedy Strategy (cf. Definition 5.27), the TDP algorithm is guaranteed to terminate after at most $2|\mathcal{T}_X||\mathcal{T}_Y|\left[\lceil\frac{\log(Z/\epsilon)}{\log(1/(1-r))}\rceil + 1\right]$ iterations.*

Proof Corollary 5.22 ensures that the value of R may always be chosen in accordance with the Less-Greedy Strategy.

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_0 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 $\left[\lceil\frac{\log(Z/\epsilon)}{\log(1/(1-r))}\rceil + 1\right]$ appearances of E in Step 3, its ZPS value ζ will be zero. Since E was an arbitrary edge from $\mathcal{E}_0$ and $|\mathcal{E}_0| \leq 2|\mathcal{T}_X||\mathcal{T}_Y|$, the result is proven. ■

Step 2 Choice	N	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 the edge from that subset whose processing in Step 3 will result in the minimal change to the 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 zt_j-edges.

Figure 5.10: Variations of the TDP Algorithm Tested

The factor $\left[\lceil \frac{\log(Z/\epsilon)}{\log(1/(1-r))} \rceil + 1\right]$ 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 11.

Corollary 5.29 (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 Lemma 5.3, $\mathcal{S}$ must be consistent. Thus, the TDP algorithm will not halt at Step 1. Using either of the strategies in Theorems 5.26 or 5.28, the algorithm will eventually terminate at Step 6. By Theorem 5.23, this only happens when the algorithm has found a solution to the TDP. ∎

Finally, the subnetworks $\mathcal{S}_X$ and $\mathcal{S}_Y$ generated by the TDP algorithm may be most concisely represented by their canonical forms, as described in Section 4.2. (By Theorem 4.53, the canonical form of an STN $\hat{\mathcal{S}}$ is guaranteed to have the minimum number of edges among all STNs equivalent to $\hat{\mathcal{S}}$.)

5.2.5 Experimental Evaluation

This section compares 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 zt_j-edges. Figure 5.10 shows the algorithm variations tested in the experiments. Each variation is identified by its parameter settings using the abbreviations in Figure 5.10. The goal of the experiments was to measure the tradeoff between solution

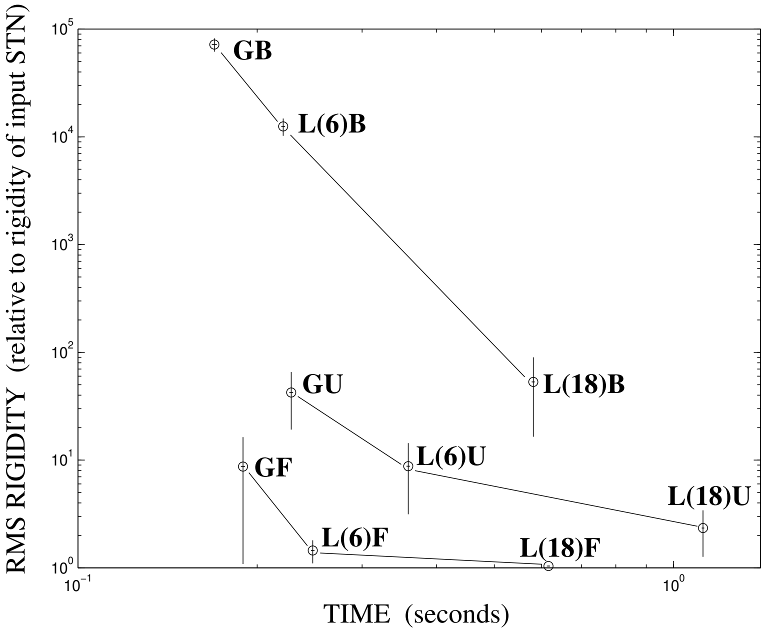


Figure 5.11: Comparing different strategies for choosing R and α in Step 3 of the TDP algorithm

quality (as measured by the rigidity of the decoupled STN) and execution time across the different algorithm variations.

The first experiment fixed the Step 2 choice function to option N (cf. Figure 5.10) while varying the Step 3 strategies for choosing R and α. 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 Figure 5.11. The horizontal axis measures time in seconds. The vertical axis measures the RMS Rigidity (cf. Definition 4.38) of the decoupled STN (as a multiple of the RMS Rigidity of the input STN). 95% confidence intervals are shown for both time and rigidity, but the intervals for time are barely visible. Both scales are logarithmic.

Lines in the plot are used to connect variations using the same α-selection function. In each grouping, the Greedy Strategy (G) is faster, but the Less-Greedy Strategy (L) results in decouplings that are substantially more flexible (i.e., less rigid). In addition, using a larger computation-multiplier in the Less-Greedy approach (18 vs. 6), which corresponds to a smaller value of the threshold ϵ (cf. Definition 5.27), results in decouplings that are more flexible.

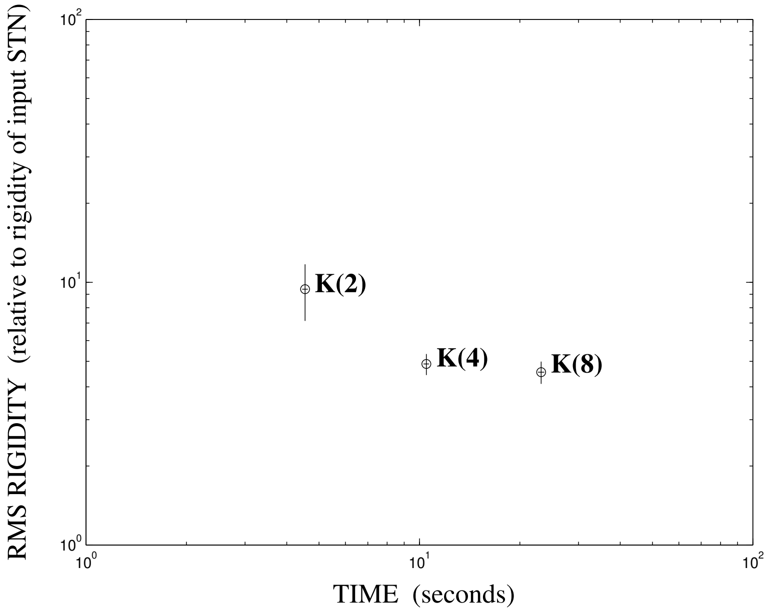


Figure 5.12: Comparing different strategies for the Step 2 choice function for the TDP algorithm

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 $t_i z$- and zt_j-edges (as is done in the Fair method, F) gives consistently more flexible decouplings, while taking less time to do so. The L(18)F variation produced decouplings that were scarcely more rigid than the input STN.

The second experiment tested the performance of the K-item-subset Step 2 choice function (i.e., option K in Figure 5.10). This Step 2 choice function is computationally intensive since for each edge in the K-item subset, constraints must be propagated (and reset) and the rigidity of the STN must be computed. Subset sizes of 2, 4 and 8 were tested in the experiment. The Step 3 choice functions for choosing R and α were fixed to options that performed well in the first experiment: the Less-Greedy Strategy (L) with a computation-multiplier of 6, and the Fair method (F) for choosing 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, shown in Figure 5.12, demonstrate that the K-item-subset Step 2 choice function can generate decoupled networks with greater flexibility. However, it is not immune to the Law of Diminishing Returns. In this case, using an 8-item subset did not appear to be worth the extra computational effort.

5.2.6 Minimal Temporal Decouplings

In the previous section, it was shown that the strategies employed in the TDP algorithm can have a significant impact on the quality of the decoupling generated by the algorithm (as measured by the rigidity of the decoupled network). This section defines a *minimal temporal decoupling* of an STN $\mathcal{S}$ to be a temporal decoupling of $\mathcal{S}$ with the property that any non-trivial weakening of the constraints in the decoupled subnetworks would necessarily cause them to no longer decouple $\mathcal{S}$. It then presents a theorem characterizing minimal temporal decouplings and a sound, complete and polynomial-time Iterative Weakening algorithm for constructing minimal temporal decouplings.

Definition 5.30 (Minimal Temporal Decoupling) *Let* $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$, $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$, *and* $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ *be STNs such that* $\mathcal{S}_X$ *and* $\mathcal{S}_Y$ *temporally decouple* $\mathcal{S}$. *The decoupling of* $\mathcal{S}$ *by* $\mathcal{S}_X$ *and* $\mathcal{S}_Y$ *is called a* minimal temporal decoupling *if the following condition holds:*

> *For any STNs* $\mathcal{S}'_X = (\mathcal{T}_X, \mathcal{C}'_X)$ *and* $\mathcal{S}'_Y = (\mathcal{T}_Y, \mathcal{C}'_Y)$ *that temporally decouple* $\mathcal{S}$*: if the constraints in* $\mathcal{C}_X$ *and* $\mathcal{C}_Y$ *entail the constraints in* $\mathcal{C}'_X$ *and* $\mathcal{C}'_Y$*, respectively, then* $\mathcal{C}_X \equiv \mathcal{C}'_X$ *and* $\mathcal{C}_Y \equiv \mathcal{C}'_Y$.

An Example of Finding a Minimal Decoupling

Consider the STN $\mathcal{S}$ in Figure 5.13. Using the z-partition defined by $\mathcal{T}_X = \{z, x_1, x_2\}$ and $\mathcal{T}_Y = \{z, y\}$, $\mathcal{S}$ has two tight, proper xy-edges: $y - x_1 \leq 4$ and $y - x_2 \leq 4$, each of which has a zero-path shortfall of $3 + 3 - 4 = 2$. If the TDP algorithm from Figure 5.8 in Section 5.2.4 were run on this STN, for example, using the Greedy Strategy (cf. Definition 5.24), it might first process the edge $y - x_1 \leq 4$, eliminating the zero-path shortfall, say, by tightening the edge from x_1 to z, as shown in Figure 5.14. Next, it might process the edge $y - x_2 \leq 4$, eliminating the zero-path shortfall, say, by tightening the edge from z to y, as shown in Figure 5.15, at which point all tight, proper xy-edges are dominated by paths through zero and, hence, the network is decoupled. In particular, $\mathcal{S}$ is decoupled by the subnetworks $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$, where

$$\mathcal{C}_X = \{ (z - x_1 \leq 1), (z - x_2 \leq 3) \} \quad \text{and} \quad \mathcal{C}_Y = \{ (y - z \leq 1) \}.$$

However, $\mathcal{S}_X$ and $\mathcal{S}_Y$ do not form a *minimal* decoupling of $\mathcal{S}$ since the edge $z - x_1 \leq 1$ may be replaced by the weaker, original edge $z - x_1 \leq 3$ without disturbing the decoupling, as shown in Figure 5.16.

The decoupling in Figure 5.16 *is* minimal, since weakening any of the intra-subnetwork constraints would result in subnetworks that do *not* decouple $\mathcal{S}$. Alternative minimal decouplings of the original STN are shown in Figure 5.17. However, the minimal decoupling in Figure 5.16 is the only minimal decoupling for the original STN that can be generated by weakening intra-subnetwork constraints in the non-minimal decoupling generated by the TDP algorithm. In other words, if $\mathcal{C}^m_X$ and $\mathcal{C}^m_Y$ are the constraint sets corresponding to the minimal decoupling shown in Figure 5.16, then the constraints in $\mathcal{C}_X$ and $\mathcal{C}_Y$ entail the constraints in $\mathcal{C}^m_X$ and $\mathcal{C}^m_Y$, respectively. The analogous statements do not hold for the minimal decouplings shown in Figure 5.17.

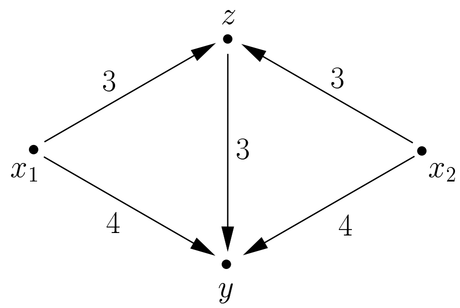


Figure 5.13: An STN for which the TDP algorithm might generate a non-minimal decoupling

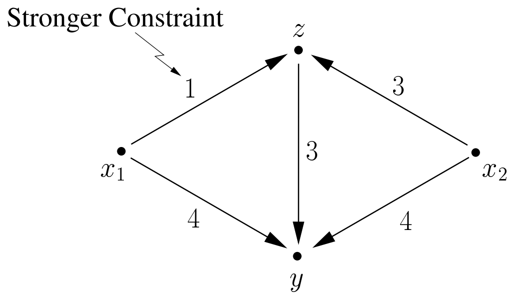


Figure 5.14: The STN from Figure 5.13 after eliminating the zero-path shortfall for one of the xy-edges

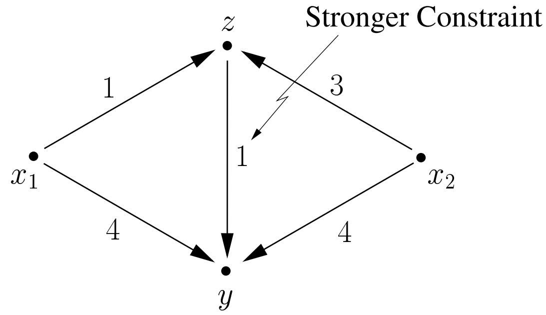


Figure 5.15: The STN from Figure 5.13 after eliminating the zero-path shortfall for both of the xy-edges

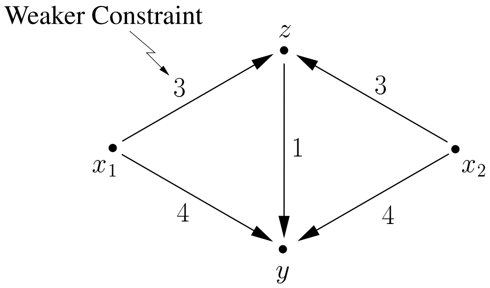


Figure 5.16: A minimal decoupling of the STN from Figure 5.13

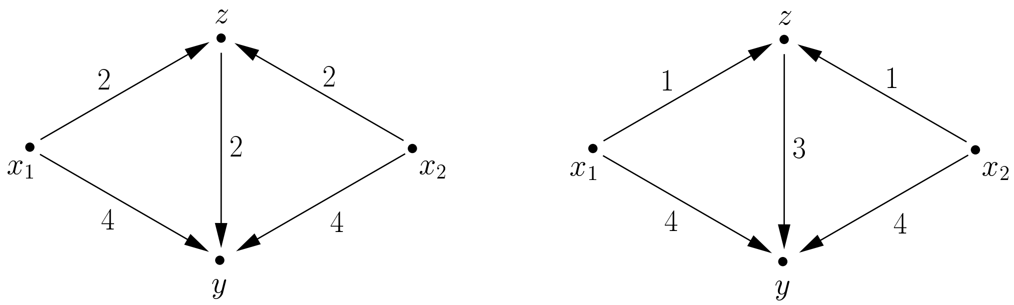


Figure 5.17: Alternative minimal decouplings of the STN from Figure 5.13

Necessary Conditions for a Minimal Decoupling

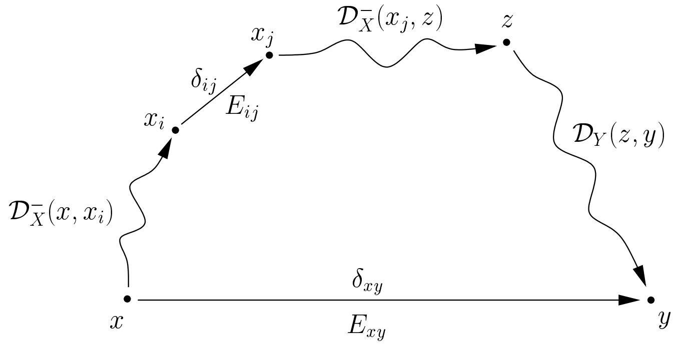
This section presents Theorem 5.33 that specifies necessary conditions for a minimal temporal decoupling. To simplify the analysis, Definition 5.31 specifies a Standard Problem Setup. Since any decoupling of $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ by $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$ requires the constraints in $\mathcal{C}_X$ and $\mathcal{C}_Y$ to entail the constraints in $\mathcal{C}_d|_{\mathcal{T}_X}$ and $\mathcal{C}_d|_{\mathcal{T}_Y}$, respectively, the constraint sets $\mathcal{C}_d|_{\mathcal{T}_X}$ and $\mathcal{C}_d|_{\mathcal{T}_Y}$ are included explicitly in the Standard Setup. However, notice that the Standard Setup does not, in and of itself, require that $\mathcal{S}_X$ and $\mathcal{S}_Y$ decouple $\mathcal{S}$.

Definition 5.31 (Standard Setup) *Let $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ be an STN whose time-points are z-partitioned by the sets $\mathcal{T}_X$ and $\mathcal{T}_Y$. Let $\mathcal{D}$ be the distance matrix for $\mathcal{S}$. Let $\mathcal{C}_X^0$ be an arbitrary set of constraints over time-points in $\mathcal{T}_X$ such that each E_{ij}: $x_j - x_i \leq \delta_{ij}$ in $\mathcal{C}_X^0$ satisfies that $\delta_{ij} < \mathcal{D}(x_i, x_j)$. Similarly, let $\mathcal{C}_Y^0$ be an arbitrary set of constraints over time-points in $\mathcal{T}_Y$ such that each E_{pq}: $y_q - y_p \leq \delta_{pq}$ in $\mathcal{C}_Y^0$ satisfies that $\delta_{pq} < \mathcal{D}(y_p, y_q)$. Let $\mathcal{C}_X = \mathcal{C}_d|_{\mathcal{T}_X} \cup \mathcal{C}_X^0$ and $\mathcal{C}_Y = \mathcal{C}_d|_{\mathcal{T}_Y} \cup \mathcal{C}_Y^0$. Finally, let $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$; and let $\mathcal{D}_X$ and $\mathcal{D}_Y$ be the corresponding distance matrices.*

Definition 5.32 (Necessary Role in a Dominating Path Through Zero) *Given the Standard Setup (cf. Definition 5.31), let E_{ij}: $x_j - x_i \leq \delta_{ij}$ be an arbitrary constraint in $\mathcal{C}_X^0$.*

Let $\mathcal{C}_X^- = \mathcal{C}_d|_{\mathcal{T}_X} \cup (\mathcal{C}_X \setminus \{E_{ij}\})$ *be the constraint set that results from removing the edge* E_{ij} *from* $\mathcal{C}_X$*. Let* $\mathcal{S}_X^- = (\mathcal{T}_X, \mathcal{C}_X^-)$*, and let* $\mathcal{D}_X^-$ *be the corresponding distance matrix. Let* E_{xy}*:* $y - x \leq \delta_{xy}$ *be some proper xy-edge in* $\mathcal{C}$*. The edge* E_{ij} *is said to play a* necessary role in a dominating path through zero *for the edge* E_{xy} *with respect to* $\mathcal{S}_X$ *and* $\mathcal{S}_Y$ *if:*

- $\mathcal{D}_X^-(x, z) + \mathcal{D}_Y(z, y) > \delta_{xy}$ *(i.e., in the absence of* E_{ij}*, the edge* E_{xy} *is not dominated by a path through zero); and*
- $\mathcal{D}_X^-(x, x_i) + \delta_{ij} + \mathcal{D}_X^-(x_j, z) + \mathcal{D}_Y(z, y) \leq \delta_{xy}$ *(i.e., in the presence of* E_{ij}*, the edge* E_{xy} is *dominated by a path through zero, as shown below).*



The definitions for E_{ij} *playing a necessary role for an edge* E_{yx} *of the form* $x - y \leq \delta_{yx}$*, and for edges in* $\mathcal{C}_Y^0$ *playing necessary roles for xy-edges in* $\mathcal{C}$ *are analogous.*

Theorem 5.33 (Necessary Conditions for a Minimum Decoupling) *Given the Standard Setup (cf. Definition 5.31), suppose that* $\mathcal{S}_X$ *and* $\mathcal{S}_Y$ *form a* minimal *temporal decoupling of* $\mathcal{S} = (\mathcal{T}, \mathcal{C})$*. Let* E_{ij}*:* $x_j - x_i \leq \delta_{ij}$ *be an arbitrary edge in* $\mathcal{C}_X^0$ *such that the removal of* E_{ij} *from* $\mathcal{C}_X^0$ *would cause a change in the distance matrix for* $\mathcal{S}_X$*. Let* $\mathcal{C}_X^-, \mathcal{S}_X^-$ *and* $\mathcal{D}_X^-$ *be as in Definition 5.32. Then* E_{ij} *plays a necessary role in a dominating path through zero for some tight, proper xy-edge in* $\mathcal{C}$*. The analogous statement for edges in* $\mathcal{C}_Y^0$ *also holds.*

Proof Let E_{ij}: $x_j - x_i \leq \delta_{ij}$ be an arbitrary edge in $\mathcal{C}_X^0$ whose removal from $\mathcal{C}_X^0$ would cause a change in the distance matrix for $\mathcal{S}_X$. Let $\mathcal{C}_X^-, \mathcal{S}_X^-$ and $\mathcal{D}_X^-$ be as in Definition 5.32.

Since the constraints in $\mathcal{S}_X$ and $\mathcal{S}_Y$ entail the constraints in $\mathcal{S}_X^-$ and $\mathcal{S}_Y$, respectively, but $\mathcal{C}_X^- \not\equiv \mathcal{C}_X$ (by construction), it must be that $\mathcal{S}_X^-$ and $\mathcal{S}_Y$ do *not* temporally decouple $\mathcal{S}$. (Otherwise, it would violate that $\mathcal{S}_X$ and $\mathcal{S}_Y$ are a *minimal* decoupling of $\mathcal{S}$.) Thus, by Theorem 5.12, at least one of the Properties 1_X, 1_Y, 2_{XY} or 2_{YX} from Theorem 5.10 fails to hold (with respect to $\mathcal{S}_X^-$ and $\mathcal{S}_Y$).

Since the constraints in $\mathcal{S}_X^-$ and $\mathcal{S}_Y$ entail the constraints in $\mathcal{C}_d|_{\mathcal{T}_X}$ and $\mathcal{C}_d|_{\mathcal{T}_Y}$, respectively, Properties 1_X and 1_Y necessarily hold. Thus, some instance of Properties 2_{XY} or 2_{YX} from Theorem 5.10 fails to hold. Thus, by Lemma 5.18, there must be at least one tight, proper xy-edge E_{xy} in $\mathcal{C}$ that is *not* dominated by a path through zero using edges in $\mathcal{S}_X^-$ and $\mathcal{S}_Y$. Without loss of generality, suppose E_{xy} has the form $y - x \leq \delta_{xy}$, where $x \in \mathcal{T}_X$ and $y \in \mathcal{T}_Y$. Then,

$$\mathcal{D}_X^-(x, z) + \mathcal{D}_Y(z, y) \ > \ \delta_{xy}.$$

Suppose that E_{ij} does *not* play a necessary role in the dominating path through zero (with respect to $\mathcal{S}_X$ and $\mathcal{S}_Y$) for E_{xy}. Given the above inequality, which is the first condition in Definition 5.32, it must be that

$$\mathcal{D}_X^-(x, x_i) + \delta_{ij} + \mathcal{D}_X^-(x_j, z) + \mathcal{D}_Y(z, y) \ > \ \delta_{xy},$$

which is the negation of the second condition in Definition 5.32. However, this pair of inequalities together imply that the edge E_{xy} is not dominated by a path through zero using edges in $\mathcal{C}_X$ and $\mathcal{C}_Y$, whether using E_{ij} or not, which contradicts that $\mathcal{S}_X$ and $\mathcal{S}_Y$ decouple $\mathcal{S}$. Thus, the result is proven for the edge E_{ij} in $\mathcal{C}_X^0$.

Edges in $\mathcal{C}_Y^0$ whose removal from $\mathcal{C}_Y$ would cause a change in the distance matrix for $\mathcal{S}_Y$ are handled analogously. ∎

The Iterative Weakening Algorithm for Constructing a Minimal Decoupling

This section presents a sound and complete algorithm for constructing a minimal temporal decoupling that runs in polynomial time. The algorithm works by weakening intra-subnetwork edges in a non-minimal decoupling (e.g., one produced by the TDP algorithm from Section 5.2.4) until a minimal decoupling is found. The amount by which each intra-subnetwork edge E may be weakened is determined by the *mu bound* for E (cf. Definition 5.34, below). The mu bound for E ensures that the tight, proper xy-edges in $\mathcal{C}$ continue to be dominated by paths through zero despite any weakening of E.

Definition 5.34 (Mu Bound) *Given the Standard Setup (cf. Definition 5.31), let* E_{ij}: $x_j - x_i \leq \delta_{ij}$ *be an arbitrary edge in* $\mathcal{C}_X^0$. *Let* $\mathcal{C}_X^-, \mathcal{S}_X^-$ *and* $\mathcal{D}_X^-$ *be as in Definition 5.32.*

Let $\mathcal{C}^Z$ *be the set of all tight, proper xy-edges in* $\mathcal{C}$ *that are not dominated by a path through zero in the absence of the edge* E_{ij}. *In other words,* $\mathcal{C}^Z = \mathcal{C}_{xy}^Z \cup \mathcal{C}_{yx}^Z$, *where:*

$$\begin{aligned}
\mathcal{C}_{xy}^Z \ &= \ \{\ (y - x \leq \delta_{xy}) \in \mathcal{C} : x \in \mathcal{T}_X,\ y \in \mathcal{T}_Y,\ \delta_{xy} = \mathcal{D}(x, y) \\
&\qquad \text{and } \mathcal{D}_X^-(x, z) + \mathcal{D}_Y(z, y) \ > \ \delta_{xy}\ \}; \quad \text{and} \\
\mathcal{C}_{yx}^Z \ &= \ \{\ (x - y \leq \delta_{yx}) \in \mathcal{C} : x \in \mathcal{T}_X,\ y \in \mathcal{T}_Y,\ \delta_{yx} = \mathcal{D}(y, x) \\
&\qquad \text{and } \mathcal{D}_Y(y, z) + \mathcal{D}_X^-(z, x) \ > \ \delta_{yx}\ \}.
\end{aligned}$$

If $\mathcal{C}^Z$ *is empty, then the* mu bound *for the edge* E_{ij} *is not defined; otherwise, it is given by the following:*

$$\mu(E_{ij}) \ = \ \min \left\{ \begin{array}{l} \{\ \delta_{xy} - \mathcal{D}_X^-(x, x_i) - \mathcal{D}_X^-(x_j, z) - \mathcal{D}_Y(z, y) : (y - x \leq \delta_{xy}) \in \mathcal{C}_{xy}^Z\ \} \\ \cup\ \ \{\ \delta_{yx} - \mathcal{D}_Y^-(y, z) - \mathcal{D}_X^-(z, x_i) - \mathcal{D}_X^-(x_j, x) : (x - y \leq \delta_{yx}) \in \mathcal{C}_{yx}^Z\ \} \end{array} \right.$$

The mu bound for an edge of the form E_{pq}: $y_q - y_p \leq \delta_{pq}$ *in* $\mathcal{C}_Y^0$ *is defined analogously.*

Lemma 5.35 (Properties of the Mu Bound) *Given the Standard Setup (cf. Definition 5.31) but such that* $\mathcal{S}_X$ *and* $\mathcal{S}_Y$ *temporally decouple* $\mathcal{S}$, *let* E_{ij}: $x_j - x_i \leq \delta_{ij}$ *be an arbitrary* tight *constraint in* $\mathcal{C}_X^0$. *Let* $\mathcal{C}_X^-, \mathcal{S}_X^-$ *and* $\mathcal{D}_X^-$ *be as in Definition 5.32. Let the set* $\mathcal{C}^Z$ *and the mu bound* $\mu(E_{ij})$ *be as in Definition 5.34. If the set* $\mathcal{C}^Z$ *is non-empty, then* $\delta_{ij} \leq \mu(E_{ij}) < \infty$. *The analogous statement applies to tight edges* E_{pq} *in* $\mathcal{C}_Y^0$.

Proof Suppose that the set $\mathcal{C}^Z$ is non-empty. Thus, $\mu(E_{ij})$ is well-defined and, furthermore, for some edge E_{xy} in $\mathcal{C}^Z$

$$\mu(E_{ij}) \;=\; \delta_{xy} - \mathcal{D}^-_X(x, x_i) - \mathcal{D}^-_X(x_j, z) - \mathcal{D}_Y(z, y),$$

where, without loss of generality, it has been assumed that E_{xy} has the form, $y - x \leq\; \delta_{xy}$. Since $\mathcal{S}$ is temporally decoupled by $\mathcal{S}_X$ and $\mathcal{S}_Y$, these STNs are all consistent (cf. Definition 5.2 and Lemma 5.3); and since removing edges from a consistent STN cannot make that STN inconsistent, the STN $\mathcal{S}^-_X$ is necessarily consistent, as well. Thus, none of the distance matrix entries in the above equation are negative infinity. Thus, $\mu(E_{ij}) < \infty$.

Now suppose that $\mu(E_{ij}) < \delta_{ij}$. Given the above equation, this implies that

$$\delta_{xy} - \mathcal{D}^-_X(x, x_i) - \mathcal{D}^-_X(x_j, z) - \mathcal{D}_Y(z, y) \;=\; \mu(E_{ij}) \;<\; \delta_{ij},$$

which, in turn, implies that

$$\delta_{xy} \;<\; \mathcal{D}^-_X(x, x_i) + \delta_{ij} + \mathcal{D}^-_X(x_j, z) + \mathcal{D}_Y(z, y).$$

However, this implies that the edge E_{xy} is not dominated by a path through zero containing the edge E_{ij}. However, E_{xy} being in $\mathcal{C}^Z$ implies that E_{xy} is not dominated by a path through zero that does *not* contain the edge E_{ij}. These statements, taken together, contradict that E_{xy} is dominated by some path through zero (whether containing E_{ij} or not) in the temporal decoupling of $\mathcal{S}$ by $\mathcal{S}_X$ and $\mathcal{S}_Y$. Thus, the assumption that $\mu(E_{ij}) < \delta$ was wrong. ■

The Iterative Weakening algorithm for constructing a minimal temporal decoupling is given in pseudo-code in Figure 5.18. The algorithm takes as input a possibly non-minimal temporal decoupling of $\mathcal{S}$ by $\tilde{\mathcal{S}}_X$ and $\tilde{\mathcal{S}}_Y$. After replacing $\tilde{\mathcal{S}}_X$ and $\tilde{\mathcal{S}}_Y$ by equivalent STNs in the form specified in the Standard Setup (cf. Definition 5.31), the algorithm processes the edges from $\mathcal{C}^0_X \cup \mathcal{C}^0_Y$ in some unspecified order. (Each edge is processed exactly once.) For each edge, the algorithm computes a helper distance matrix ($\mathcal{D}^-_X$ or $\mathcal{D}^-_Y$) and the set $\mathcal{C}^Z$ (cf. Definition 5.34). If $\mathcal{C}^Z$ is empty, the edge is safely removed from $\mathcal{C}^0_X \cup \mathcal{C}^0_Y$; otherwise, the corresponding mu bound is computed (cf. Definition 5.34) and the edge is replaced in $\mathcal{C}_X \cup \mathcal{C}_Y$ by a possibly weaker edge whose strength is specified by the mu bound. The algorithm returns a minimal temporal decoupling whose *modified* constraint sets $\mathcal{C}_X$ and $\mathcal{C}_Y$ are entailed by the input constraint sets in $\tilde{\mathcal{S}}_X$ and $\tilde{\mathcal{S}}_Y$, respectively.

Theorem 5.36 *The Iterative Weakening algorithm for constructing a minimal temporal decoupling is sound and complete.*

Proof Since the Iterative Weakening algorithm has at most $|\mathcal{T}_X|^2 + |\mathcal{T}_Y|^2$ iterations, it always terminates. Thus, it suffices to check that subnetworks returned by the algorithm form a minimal temporal decoupling of $\mathcal{S}$.

To avoid confusion between the constraint sets $\mathcal{C}_X$ and $\mathcal{C}_Y$ constructed in Step 0, and those returned by the algorithm, the latter shall be denoted by $\hat{\mathcal{C}}_X$ and $\hat{\mathcal{C}}_Y$, respectively. Similarly, $\hat{\mathcal{S}}_X = (\mathcal{T}_X, \hat{\mathcal{C}}_X)$, $\hat{\mathcal{S}}_Y = (\mathcal{T}_X, \hat{\mathcal{C}}_Y)$, $\hat{\mathcal{D}}_X$ and $\hat{\mathcal{D}}_Y$ denote the corresponding subnetworks and distance matrices existing at the end of the algorithm.

Given: STNs $\tilde{\mathcal{S}}_X = (\mathcal{T}_X, \tilde{\mathcal{C}}_X)$, $\tilde{\mathcal{S}}_Y = (\mathcal{T}_Y, \tilde{\mathcal{C}}_Y)$ and $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ such that $\tilde{\mathcal{S}}_X$ and $\tilde{\mathcal{S}}_Y$ temporally decouple $\mathcal{S}$.

(0) Let $\mathcal{C}_X = \mathcal{C}_d|_{\mathcal{T}_X} \cup \mathcal{C}_X^0$ and $\mathcal{C}_Y = \mathcal{C}_d|_{\mathcal{T}_Y} \cup \mathcal{C}_Y^0$ as specified in the Standard Setup (cf. Definition 5.31), but such that $\mathcal{C}_X \equiv \tilde{\mathcal{C}}_X$ and $\mathcal{C}_Y \equiv \tilde{\mathcal{C}}_Y$. Let $\mathcal{S}_X, \mathcal{S}_Y, \mathcal{D}_X$ and $\mathcal{D}_Y$ be the corresponding STNs and distance matrices (again, as in the Standard Setup).

(1) Initialize $\mathcal{E}$ to the set $\mathcal{C}_X^0 \cup \mathcal{C}_Y^0$.

(2) If $\mathcal{E}$ is empty, go to Step 5. Otherwise, *remove* some edge E from $\mathcal{E}$. If $E \in \mathcal{C}_X^0$, go to Step 3a; otherwise, go to Step 4a.

(3a) Let $\mathcal{C}_X^- = \mathcal{C}_d|_{\mathcal{T}_X} \cup (\mathcal{C}_X \setminus \{E\})$ and $\mathcal{S}_X^- = (\mathcal{T}_X, \mathcal{C}_X^-)$ be as in Definition 5.32. Compute the corresponding distance matrix $\mathcal{D}_X^-$.

(3b) Compute the set $\mathcal{C}^Z = \mathcal{C}_{xy}^Z \cup \mathcal{C}_{yx}^Z$ of tight, proper xy-edges in $\mathcal{C}$ that are not dominated by a path through zero in the absence of the edge E, as specified in Definition 5.34.

(3c) If $\mathcal{C}^Z$ is empty, remove E from the set $\mathcal{C}_X^0$ (it does not make a necessary contribution to the decoupling of $\mathcal{S}$) and go back to Step 2.

(3d) Compute the mu bound μ for the edge E, as specified in Definition 5.34. Since $E \in \mathcal{C}_X^0$, it has the form $x_j - x_i \leq \delta_{ij}$. If $\mu \geq \mathcal{D}(x_i, x_j)$, then simply *remove* the edge E from $\mathcal{C}_X^0$. If $\mu > \delta_{ij}$, then *replace* the edge E in $\mathcal{C}_X^0$ with the weaker edge E_μ: $x_j - x_i \leq \mu$. (If $\mu = \delta_{ij}$, the edge E cannot be weakened.) Go back to Step 2.

(4a) Let $\mathcal{C}_Y^- = \mathcal{C}_d|_{\mathcal{T}_Y} \cup (\mathcal{C}_Y \setminus \{E\})$ and $\mathcal{S}_Y^- = (\mathcal{T}_Y, \mathcal{C}_Y^-)$ be as in Definition 5.32. Compute the corresponding distance matrix $\mathcal{D}_Y^-$.

(4b) Compute the set $\mathcal{C}^Z = \mathcal{C}_{xy}^Z \cup \mathcal{C}_{yx}^Z$ of tight, proper xy-edges in $\mathcal{C}$ that are not dominated by a path through zero in the absence of the edge E, as specified in Definition 5.34.

(4c) If $\mathcal{C}^Z$ is empty, remove E from the set $\mathcal{C}_Y^0$ (it does not make a necessary contribution to the decoupling of $\mathcal{S}$) and go back to Step 2.

(4d) Compute the mu bound μ for the edge E, as specified in Definition 5.34. Since $E \in \mathcal{C}_Y^0$, it has the form $y_q - y_p \leq \delta_{pq}$. If $\mu \geq \mathcal{D}(y_p, y_q)$, then simply *remove* the edge E from $\mathcal{C}_Y^0$. If $\mu > \delta_{pq}$, then *replace* the edge E in $\mathcal{C}_Y^0$ with the weaker edge E_μ: $y_q - y_p \leq \mu$. (If $\mu = \delta_{pq}$, the edge E cannot be weakened.) Go back to Step 2.

(5) **Return:** $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_d|_{\mathcal{T}_X} \cup \mathcal{C}_X^0)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_d|_{\mathcal{T}_Y} \cup \mathcal{C}_Y^0)$, where $\mathcal{C}_X^0$ and $\mathcal{C}_Y^0$ may have been *modified* during the course of the algorithm.

Figure 5.18: The Iterative Weakening algorithm for constructing a minimal temporal decoupling

(Proof that $\hat{\mathcal{S}}_X$ and $\hat{\mathcal{S}}_Y$ temporally decouple $\mathcal{S}$) Suppose not. Then there must have been some point in the algorithm at which the decoupling property first failed to hold. Let E be the edge whose removal or replacement caused the decoupling property to fail for the first time. In other words, prior to removing or replacing E, the decoupling property held, but immediately afterward it did not. Without loss of generality, suppose the edge $E = E_{ij}$ has the form, $x_j - x_i \leq \delta_{ij}$, and let $\mathcal{C}^Z$ and, if applicable, $\mu_{ij} = \mu(E_{ij})$ be as specified in Definition 5.34.

First, since $\mathcal{C}_d|_{\mathcal{T}_X} \subseteq \mathcal{C}_X$ and $\mathcal{C}_d|_{\mathcal{T}_Y} \subseteq \mathcal{C}_Y$ throughout the algorithm, Properties 1_X and 1_Y of Theorem 5.10 hold with respect to $\mathcal{S}_X$ and $\mathcal{S}_Y$ throughout the algorithm. Thus, by Theorem 5.12, the only way the decoupling property can fail to hold is by some instance of Properties 2_{XY} or 2_{YX} failing to hold. Thus, by Lemma 5.18, the only way the decoupling property can fail to hold is by some tight, proper xy-edge in $\mathcal{S}$ failing to be dominated by some path through zero using edges in $\mathcal{C}_X$ and $\mathcal{C}_Y$.

If $\mathcal{C}^Z$ is empty, then there are no tight, proper xy-edges in $\mathcal{C}$ that require the presence of E_{ij}. Thus, removing E_{ij} could not disturb the property that each tight, proper xy-edge in $\mathcal{C}$ is dominated by a path through zero.

On the other hand, suppose $\mathcal{C}^Z$ is non-empty. Let E_{xy} be an arbitrary edge in $\mathcal{C}^Z$. Without loss of generality, E_{xy} has the form, $y - x \leq \delta_{xy}$. By definition of the mu bound,

$$\mu_{ij} \;\leq\; \delta_{xy} - \mathcal{D}^{-}_{X,alg}(x, x_i) - \mathcal{D}^{-}_{X,alg}(x_j, z) - \mathcal{D}_{Y,alg}(z, y),$$

where the *alg* subscripts are used to indicate that the distance matrices are those that existed during the computation of μ_{ij} by the algorithm (i.e., prior to replacing or removing the edge E_{ij}). However, this implies that

$$\mathcal{D}^{-}_{X,alg}(x, x_i) + \mu_{ij} + \mathcal{D}^{-}_{X,alg}(x_j, z) + \mathcal{D}_{Y,alg}(z, y) \;\leq\; \delta_{xy},$$

which implies that the edge E_{xy} is dominated by a path through zero even *after* replacing the edge E_{ij} by the edge E_μ: $x_j - x_i \leq \mu_{ij}$.

Since E_{xy} was arbitrary in $\mathcal{C}^Z$, all of the tight, proper xy-edges in $\mathcal{C}^Z$ must be dominated by paths through zero even after replacing E_{ij} by E_μ. However, all of the tight, proper xy-edges in $\mathcal{C}$ that are *not* in $\mathcal{C}^Z$ do not require E_{ij} for their dominating paths (cf. Definition 5.34). Thus, every tight, proper xy-edge in $\mathcal{C}$ is dominated by a path through zero even after replacing E_{ij} by E_μ. (If $\mu_{ij} \leq \mathcal{D}(x_i, x_j)$, the above argument still applies, but there is no need to insert the edge E_μ into $\mathcal{C}^0_X$, since the corresponding edge in $\mathcal{C}_d|_{\mathcal{T}_X}$ is at least as strong as E_μ.)

Thus, in either case, the temporal decoupling property holds after the processing of the edge E_{ij}, which is a contradiction. Since there is no point in the algorithm at which the decoupling property first fails, the decoupling property must hold all the way through the algorithm, including at the very end.

(Proof that $\hat{\mathcal{S}}_X$ and $\hat{\mathcal{S}}_Y$ are a *minimal* temporal decoupling of $\mathcal{S}$) Let $\mathcal{S}'_X = (\mathcal{T}_X, \mathcal{C}'_X)$ and $\mathcal{S}'_Y = (\mathcal{T}_Y, \mathcal{C}'_Y)$ be arbitrary STNs such that the constraints in $\hat{\mathcal{C}}_X$ and $\hat{\mathcal{C}}_Y$ entail the constraints in $\mathcal{C}'_X$ and $\mathcal{C}'_Y$, respectively, but that $\hat{\mathcal{C}}_X \not\equiv \mathcal{C}'_X$ or $\hat{\mathcal{C}}_Y \not\equiv \mathcal{C}'_Y$. Without loss of generality, assume that $\hat{\mathcal{C}}_X \not\equiv \mathcal{C}'_X$. Without loss of generality, further assume that $\hat{\mathcal{C}}^0_X$ only contains tight edges. Finally, assume that the constraints in $\mathcal{C}'_X$ and $\mathcal{C}'_Y$ entail the constraints in $\mathcal{C}_d|_{\mathcal{T}_X}$ and $\mathcal{C}_d|_{\mathcal{T}_Y}$, respectively, since otherwise, by Theorem 5.10, $\mathcal{S}'_X$ and $\mathcal{S}'_Y$ could not form a temporal decoupling of $\mathcal{S}$.

It is sufficient to show that $\mathcal{S}'_X$ and $\mathcal{S}'_Y$ do *not* temporally decouple $\mathcal{S}$. Thus, it is sufficient to show that there is some tight, proper xy-edge in $\mathcal{C}$ that is *not* dominated by a path through zero with respect to the subnetworks $\mathcal{S}'_X$ and $\mathcal{S}'_Y$.

Since $\hat{\mathcal{C}}_X \not\equiv \mathcal{C}'_X$, and since the constraints in $\mathcal{C}'_X$ entail those in $\mathcal{C}_d|_{\mathcal{T}_X}$, there must exist some edge E_μ: $x_j - x_i \leq \mu_{ij}$ in $\hat{\mathcal{C}}^0_X$, where $x_i, x_j \in \mathcal{T}_X$ and

$$\mu_{ij} \;=\; \hat{\mathcal{D}}_X(x_i, x_j) \;<\; \mathcal{D}'_X(x_i, x_j).$$

Notice that since each edge in $\mathcal{C}^0_X$ is processed exactly once in the algorithm, the value μ_{ij} must be the value that was computed during the algorithm when the original edge E_{ij} from x_i to x_j in $\mathcal{C}^0_X$ was processed.

Let $\mathcal{C}^Z_{alg}$ be the set of xy-edges in $\mathcal{C}$ that required the presence of the edge E_{ij} for their dominating paths through zero with respect to the constraint sets $\mathcal{C}_{X,alg}$ and $\mathcal{C}_{Y,alg}$, where the alg subscripts indicate that the constraint sets are those that existed during the processing of E_{ij} by the algorithm. Notice that $\mathcal{C}^Z_{alg}$ cannot be empty since otherwise the edge E_{ij} would have been discarded without replacement, contradicting the existence of the edge E_μ in $\hat{\mathcal{C}}^0_X$.

Let $E_{xy} \in \mathcal{C}^Z_{alg} \subseteq \mathcal{C}$ be such that

$$\mu_{ij} \;=\; \delta_{xy} - \mathcal{D}^-_{X,alg}(x, x_i) - \mathcal{D}^-_{X,alg}(x_j, z) - \mathcal{D}_{Y,alg}(z, y),$$

where it has been assumed, without loss of generality, that E_{xy} has the form, $y - x \leq \delta_{xy}$. Rearranging terms yields the following:

$$\delta_{xy} \;=\; \mathcal{D}^-_{X,alg}(x, x_i) + \mu_{ij} + \mathcal{D}^-_{X,alg}(x_j, z) + \mathcal{D}_{Y,alg}(z, y).$$

Since the replacement edges are never stronger than the edges they replace (cf. Lemma 5.35), the distance matrix entries can only ever stay the same or increase during the course of the algorithm. Therefore, the above equality implies the following:

$$\delta_{xy} \;\leq\; \hat{\mathcal{D}}^-_X(x, x_i) + \mu_{ij} + \hat{\mathcal{D}}^-_X(x_j, z) + \hat{\mathcal{D}}_Y(z, y),$$

where $\hat{\mathcal{C}}^-_X = \mathcal{C}_d|_{\mathcal{T}_X} \cup (\hat{\mathcal{C}}^0_X \setminus \{E_\mu\})$ and $\hat{\mathcal{S}}^-_X = (\mathcal{T}_X, \hat{\mathcal{C}}^-_X)$, and where $\hat{\mathcal{D}}^-_X$ is the corresponding distance matrix. Since $\mu_{ij} < \mathcal{D}'_X(x_i, x_j)$, the above inequality implies the following *strict* inequality:

$$\delta_{xy} \;<\; \hat{\mathcal{D}}^-_X(x, x_i) + \mathcal{D}'_X(x_i, x_j) + \hat{\mathcal{D}}^-_X(x_j, z) + \hat{\mathcal{D}}_Y(z, y).$$

Furthermore, since the constraints in $\hat{\mathcal{C}}_X$ entail the constraints in $\mathcal{C}'_X$, the above inequality yields the following:

$$\delta_{xy} \;<\; \mathcal{D}'^-_X(x, x_i) + \mathcal{D}'_X(x_i, x_j) + \mathcal{D}'^-_X(x_j, z) + \mathcal{D}'_Y(z, y),$$

where ${\mathcal{C}'_X}^-$ is the constraint set $\mathcal{C}'_X$ minus any edge from x_i to x_j that might exist in $\mathcal{C}'_X$ and ${\mathcal{S}'_X}^- = (\mathcal{T}_X, {\mathcal{C}'_X}^-)$, and where ${\mathcal{D}'_X}^-$ is the corresponding distance matrix. However, this implies that the edge E_{xy} is *not* dominated by any path P through zero, using edges in $\mathcal{C}'_X \cup \mathcal{C}'_Y$ and such that P contains a subpath from x_i to x_j. However, the definition of $\mathcal{C}^Z$ (of which E_{xy} is a member) implies that E_{xy} is not dominated by any paths through zero using edges only from $\hat{\mathcal{C}}_X^-$ and $\hat{\mathcal{C}}_Y$. Since the constraints in $\hat{\mathcal{C}}_X^-$ and $\hat{\mathcal{C}}_Y$ entail the constraints in ${\mathcal{C}'_X}^-$ and $\mathcal{C}'_Y$, respectively, it follows that E_{xy} is not dominated by any paths through zero using edges only from ${\mathcal{C}'_X}^-$ and $\mathcal{C}'_Y$. Thus, E_{xy} is not dominated by any path through zero, using edges in $\mathcal{C}'_X \cup \mathcal{C}'_Y$, whether including a subpath from x_i to x_j or not, which implies that $\mathcal{S}'_X$ and $\mathcal{S}'_Y$ do *not* temporally decouple $\mathcal{S}$.

Since $\mathcal{S}'_X$ and $\mathcal{S}'_Y$ were arbitrary (such that their constraint sets were entailed by $\hat{\mathcal{C}}_X$ and $\hat{\mathcal{C}}_Y$, and such that $\hat{\mathcal{C}}_X \not\equiv \mathcal{C}'_X$ or $\hat{\mathcal{C}}_Y \not\equiv \mathcal{C}'_Y$), the STNs $\hat{\mathcal{S}}_X$ and $\hat{\mathcal{S}}_Y$ have been shown to be a *minimal* temporal decoupling of $\mathcal{S}$. ■

Returning to the non-minimal decoupling shown in Figure 5.15, notice that the Iterative Weakening algorithm, when applied to the edge from x_1 to z, would compute the constraint set $\mathcal{C}^Z = \{(y - x \leq 4)\}$ and $\mu = 4 - 0 - 0 - 1 = 3$. Thus, the edge $z - x_1 \leq 1$ would be replaced by the strictly weaker edge $z - x_1 \leq 3$, resulting in the minimal decoupling shown in Figure 5.16.

5.2.7 The Optimal Temporal Decoupling Problem

This section defines the Optimal Temporal Decoupling Problem in which a temporal decoupling that maximizes some metric h of solution quality is sought. A candidate metric might measure the flexibility of the decoupled network. Variations on this theme might take into account the flexibility of the decoupled subnetworks as well. In the case of an agent decoupling a portion of its Master STN in preparation for its participation in a task-allocation auction, the metric of solution quality might need to include some measure of how much *space* is available for tasks the agent might want to bid on.

Definition 5.37 (Temporal Decoupling Metric) *A function h is a* metric of the quality of temporal decouplings *if for any STNs $\mathcal{S}$, $\mathcal{S}_X$ and $\mathcal{S}_Y$ such that $\mathcal{S}_X$ and $\mathcal{S}_Y$ temporally decouple $\mathcal{S}$, $h(\mathcal{S}, \mathcal{S}_X, \mathcal{S}_Y)$ specifies some real number.*

A sample metric h_r, based on the measure of STN rigidity defined in Section 4.1.4, is given by:

$$h_r((\mathcal{T}, \mathcal{C}),\ (\mathcal{T}_X, \mathcal{C}_X),\ (\mathcal{T}_Y, \mathcal{C}_Y)) \ = \ -Rig((\mathcal{T}, \mathcal{C} \cup \mathcal{C}_X \cup \mathcal{C}_Y)),$$

where the negative sign ensures that decouplings with greater flexibility have higher quality.

Definition 5.38 (Optimal Temporal Decoupling Problem) *Let $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ be an STN whose time-points $\mathcal{T}$ are z-partitioned by the sets $\mathcal{T}_X$ and $\mathcal{T}_Y$. Let h be a metric of the quality of temporal decouplings. Find sets $\mathcal{C}_X$ and $\mathcal{C}_Y$ of constraints over time-points in $\mathcal{T}_X$ and $\mathcal{T}_Y$, respectively, that maximize $h((\mathcal{T}, \mathcal{C}),\ (\mathcal{T}_X, \mathcal{C}_X),\ (\mathcal{T}_Y, \mathcal{C}_Y))$ subject to the constraint that $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$ temporally decouple $\mathcal{S}$.*

Definition 5.39 (Metric Strongly Biased Toward Flexibility) *Let* $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ *be an STN whose time-points* $\mathcal{T}$ *are z-partitioned by the sets* $\mathcal{T}_X$ *and* $\mathcal{T}_Y$*. A metric* h *of the quality of temporal decouplings is said to be* strongly biased toward flexibility *if*

$$h((\mathcal{T},\mathcal{C}),\ (\mathcal{T}_X,\mathcal{C}_X),\ (\mathcal{T}_Y,\mathcal{C}_Y)) \quad < \quad h((\mathcal{T},\mathcal{C}),\ (\mathcal{T}_X,\mathcal{C}'_X),\ (\mathcal{T}_Y,\mathcal{C}'_Y))$$

whenever the constraints in $\mathcal{C}_X$ *and* $\mathcal{C}_Y$ *entail the constraints in* $\mathcal{C}'_X$ *and* $\mathcal{C}'_Y$*, respectively.*

Notice that for any metric that is strongly biased toward flexibility, including the metric h_r defined above, any *minimal* temporal decoupling (cf. Definition 5.30) represents a *locally* optimal temporal decoupling.

Finding globally optimal temporal decouplings (i.e., solutions to the Optimal Temporal Decoupling Problem) is beyond the scope of this thesis. That would enable comparing the solution quality attained by the algorithm varieties tested in Section 5.2.5 against the quality of the optimal solution, according to various metrics.

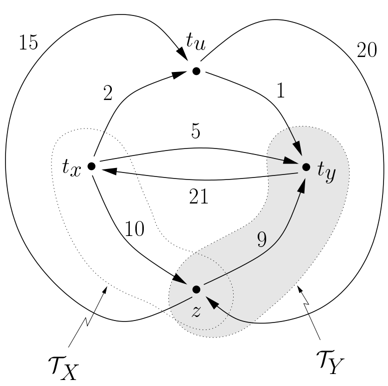
5.3 The Allocative Temporal Decoupling Problem

In certain applications (e.g., when an agent needs to decouple a portion of its Master STN in preparation for its participation in a task-allocation auction), the initial contents of the sets $\mathcal{T}_X$ and $\mathcal{T}_Y$ (and hence the z-partition for the subsequent decoupling) may be only partially specified. In such scenarios, the Temporal Decoupling Problem requires not only the addition of intra-subnetwork constraints sufficient to decouple the network, but also the completion of the partially specified z-partition.

This section defines the Allocative Temporal Decoupling Problem in which (1) a partially specified z-partition must be completed, and (2) a temporal decoupling must be constructed using that z-partition. Like the Optimal TDP in Section 5.2.7, the Allocative Temporal Decoupling Problem is cast as an optimization problem, based on a metric h of the quality of temporal decouplings.

This section also presents a greedy, incremental algorithm for finding approximate solutions to the Allocative TDP. The algorithm interleaves (1) the incremental allocation of time-points to $\mathcal{T}_X$ and $\mathcal{T}_Y$ with (2) the incremental addition of intra-subnetwork temporal constraints aimed at providing the eventual decoupling. The incremental-allocation decision-making functionality is encapsulated by an *oracle* function $\mathcal{F}$ which, given a partially decoupled network (based on a partially specified z-partition), generates sets $\Delta\mathcal{T}_X$ and $\Delta\mathcal{T}_Y$ of time-points to add to $\mathcal{T}_X$ and $\mathcal{T}_Y$, respectively. The incremental addition of intra-subnetwork constraints is handled by the basic TDP algorithm, modified to focus on a restricted set of tight, proper xy-edges. For guidance, each process requires the metric h to be able to measure the quality not only of fully decoupled networks, but also of *partially* decoupled networks.

The rest of this section is organized as follows. Section 5.3.1 introduces the Allocative Temporal Decoupling Problem by walking through a sample instance; Section 5.3.2



	z	t_x	t_y	t_u
z	0	30	9	15
t_x	10	0	3	2
t_y	31	21	0	23
t_u	20	22	1	0

$$\begin{aligned} \text{RMS}_X &= 0.024 \\ \text{RMS}_Y &= 0.024 \\ \text{RMS}_{XY} &= 0.040 \\ \text{RMS}_{\text{all}} &= 0.034 \end{aligned}$$

Figure 5.19: A sample STN with an initial allocation of time-points to $\mathcal{T}_X$ and $\mathcal{T}_Y$

formally defines the Allocative TDP; Section 5.3.3 presents an algorithm for finding approximate solutions to the Allocative TDP; and Section 5.3.4 applies the Allocative TDP algorithm to the Auction-Participation-Context-Generation Problem from Chapter 3.

5.3.1 A Sample Instance of the Allocative TDP

Figure 5.19 shows a sample STN containing four time-points: z, t_x, t_y and t_u. Initially, $\mathcal{T}_X$ contains z and t_x; $\mathcal{T}_Y$ contains z and t_y; and t_u is unallocated. The only proper xy-edges[2] are:

$$E_{xy}\colon t_y - t_x \le 5 \quad \text{and} \quad E_{yx}\colon t_x - t_y \le 21.$$

E_{yx} is a tight edge; E_{xy} is not (because there is a path from t_x to t_y via t_u that has length 3, which is less than the length of E_{xy}). The corresponding distance matrix is also shown in the figure, as are several relevant measures of rigidity (cf. Definition 4.38). RMS$_X$ measures the RMS rigidity of edges among points in $\mathcal{T}_X$; RMS$_Y$ measures the RMS rigidity of edges among points in $\mathcal{T}_Y$; RMS$_{XY}$ measures the RMS rigidity of edges joining points in $\mathcal{T}_X$ to points in $\mathcal{T}_Y$; and RMS$_{\text{all}}$ measures the RMS rigidity of the entire network.

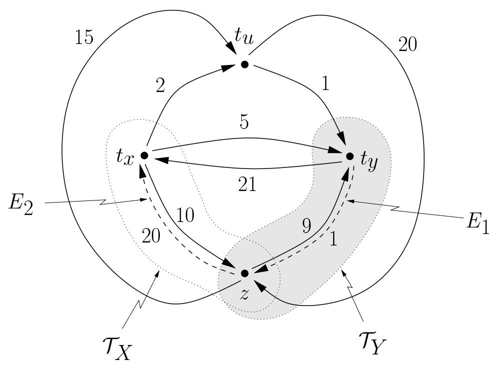
Adding the edges

$$E_1\colon z - t_y \le 1 \ \text{(i.e., } \delta_1 = 1\text{)} \quad \text{and} \quad E_2\colon t_x - z \le 20 \ \text{(i.e., } \delta_2 = 20\text{)},$$

shown as dashed arcs in Figure 5.20, provides a path through zero that dominates the edge E_{yx}.[3] Adding the edges E_1 and E_2 to the network causes its RMS rigidity to increase from

[2]Throughout this section, it is presumed that the notion of an xy-edge (cf. Definition 5.13) is extended (in the obvious way) to include the case where the sets $\mathcal{T}_X$ and $\mathcal{T}_Y$ do not form a proper z-partition of $\mathcal{T}$.

[3]Notice that the updated distance matrix entries satisfy, $\mathcal{D}(t_y, z) = \delta_1 = 1$ and $\mathcal{D}(z, t_x) = \delta_2 = 20$, as predicted by Corollary 5.20.



	z	t_x	t_y	t_u
z	0	20	9	15
t_x	10	0	3	2
t_y	1	21	0	16
t_u	2	22	1	0

$$\begin{aligned} \text{RMS}_X &= 0.040 \\ \text{RMS}_Y &= 0.091 \\ \text{RMS}_{XY} &= 0.040 \\ \text{RMS}_{\text{all}} &= 0.057 \end{aligned}$$

Figure 5.20: A partial decoupling of the STN from Figure 5.19

0.034 to 0.057.[4]

At this point, there are two ways to proceed: the unallocated time-point t_u may be allocated to either $\mathcal{T}_X$ or $\mathcal{T}_Y$. Allocating t_u to $\mathcal{T}_X$, as shown in Figure 5.21, introduces the new tight, proper xy-edge, E_{uy}: $t_y - t_u \leq 1$. Adding the edges

$$E_1'\colon z - t_u \leq -5.5 \quad \text{and} \quad E_2'\colon t_y - z \leq 6.5,$$

shown as dashed arcs in Figure 5.22, provides a path through zero that dominates the edge E_{uy}. The subnetworks corresponding to $\mathcal{T}_X$ and $\mathcal{T}_Y$ are now fully decoupled. The RMS rigidity of the entire network has increased to 0.075.

On the other hand, allocating t_u to $\mathcal{T}_Y$, as shown in Figure 5.23, introduces the new tight, proper xy-edge, E_{xu}: $t_u - t_x \leq 2$. Adding the edges

$$E_1''\colon z - t_x \leq 3.15 \quad \text{and} \quad E_2''\colon t_u - z \leq -1.15,$$

shown as dashed arcs in Figure 5.24, provides a path through zero that dominates the edge E_{xu}. Once again, the subnetworks corresponding to $\mathcal{T}_X$ and $\mathcal{T}_Y$ are temporally decoupled. This time, the RMS rigidity of the entire network is 0.383, which is substantially larger.

5.3.2 Definition of the Allocative TDP

Definition 5.40 (The Allocative TDP) *Let $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ be a consistent STN. Let $\mathcal{T}_X^0$ and $\mathcal{T}_Y^0$ be sets of time-points such that $\mathcal{T}_X^0 \cap \mathcal{T}_Y^0 = \{z\}$ and $\mathcal{T}_X^0 \cup \mathcal{T}_Y^0 \subseteq \mathcal{T}$. Find:*

[4] Notice that the inequality $\mathcal{D}(t_x, z) + \mathcal{D}(z, t_y) \leq \mathcal{D}(t_x, t_y)$ (cf. Lemma 5.18) does *not* hold, even though (t_x, t_y) is a proper xy-pair. Thus, the subnetworks corresponding to $\mathcal{T}_X$ and $\mathcal{T}_Y$ are not yet fully decoupled, even though all tight, proper xy-edges (with respect to $\mathcal{T}_X$ and $\mathcal{T}_Y$) are dominated by paths through zero. This does not contradict Lemma 5.18 because that lemma relies on a strict definition of tight, proper xy-edges arising from sets $\mathcal{T}_X$ and $\mathcal{T}_Y$ that properly z-partition $\mathcal{T}$, which is not the case here.

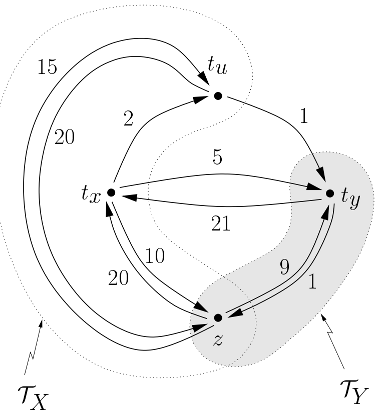
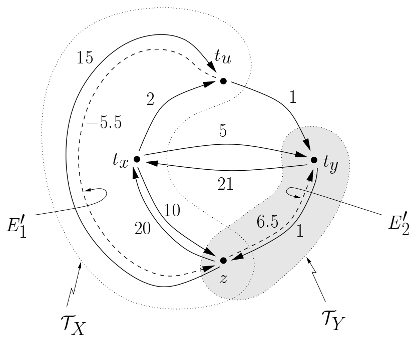


Figure 5.21: Allocating t_u to $\mathcal{T}_X$



	z	t_x	t_y	t_u
z	0	20	6.5	15
t_x	-3.5	0	3	2
t_y	1	21	0	16
t_u	-5.5	14.5	1	0

$$\begin{aligned} \mathrm{RMS}_X &= 0.072 \\ \mathrm{RMS}_Y &= 0.118 \\ \mathrm{RMS}_{XY} &= 0.048 \\ \mathrm{RMS}_{\mathrm{all}} &= 0.075 \end{aligned}$$

Figure 5.22: A decoupling in which $t_u \in \mathcal{T}_X$

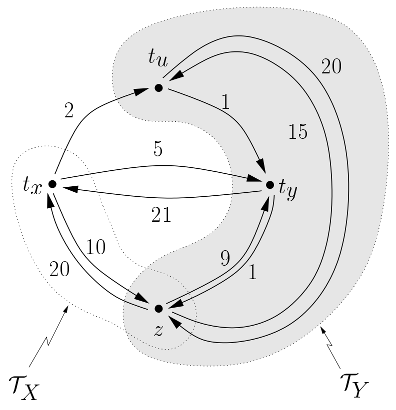
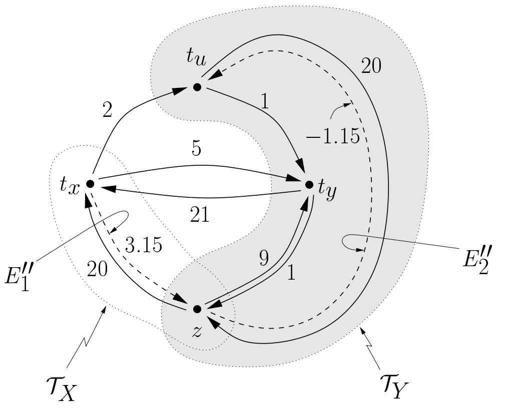


Figure 5.23: Allocating t_u to $\mathcal{T}_Y$



	z	t_x	t_y	t_u
z	0	20	0.15	-1.15
t_x	-3.5	0	3	2
t_y	1	21	0	-0.15
t_u	2	22	1	0

$$\begin{aligned} \text{RMS}_X &= 0.163 \\ \text{RMS}_Y &= 0.517 \\ \text{RMS}_{XY} &= 0.040 \\ \text{RMS}_{\text{all}} &= 0.383 \end{aligned}$$

Figure 5.24: A decoupling in which $t_u \in \mathcal{T}_Y$

- *sets* $\mathcal{T}_X$ *and* $\mathcal{T}_Y$ *of time-points such that* $\mathcal{T}_X^0 \subseteq \mathcal{T}_X$, $\mathcal{T}_Y^0 \subseteq \mathcal{T}_Y$, *and* $\mathcal{T}_X$ *and* $\mathcal{T}_Y$ *z-partition* $\mathcal{T}$; *and*
- *sets* $\mathcal{C}_X$ *and* $\mathcal{C}_Y$ *of constraints over the time-points in* $\mathcal{T}_X$ *and* $\mathcal{T}_Y$ *such that* $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ *and* $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$ *temporally decouple* $\mathcal{S}$,

such that the quantity $h(\mathcal{S}, \mathcal{S}_X, \mathcal{S}_Y)$ *is maximized.*

5.3.3 An Algorithm for Finding Approximate Solutions to the Allocative TDP

This section presents a greedy, incremental algorithm for finding approximate solutions to the Allocative TDP. The algorithm interleaves two processes: (1) the incremental allocation of time-points to the sets $\mathcal{T}_X$ and $\mathcal{T}_Y$; and (2) the incremental addition of intra-subnetwork temporal constraints. Each process requires the metric h to be able to measure the quality not only of fully decoupled networks, but also of partially decoupled networks. The incremental allocation of time-points to $\mathcal{T}_X$ and $\mathcal{T}_Y$ is encapsulated by an oracle function $\mathcal{F}$. The incremental addition of intra-subnetwork temporal constraints is handled by the basic TDP algorithm from Section 5.2.4, modified to restrict attention to a limited pool of tight, proper xy-edges.

Since the contents of $\mathcal{T}_X$ and $\mathcal{T}_Y$ determine whether any given edge is classified as an xy-edge, every new allocation of a time-point to $\mathcal{T}_X$ or $\mathcal{T}_Y$ has the potential to cause previously unclassified edges to become newly classified as xy-edges. Thus, the pool of tight, proper xy-edges that is the central focus of the basic TDP algorithm (cf. Section 5.2.4) has the potential to grow over time. The algorithm presented in this section lazily adds intra-subnetwork constraints sufficient to ensure that every edge in the *current* pool of tight, proper xy-edges is dominated by a path through zero. This is accomplished by modifying the initialization step of the basic TDP algorithm such that only the most recent additions to the pool of tight, proper xy-edges are included in the set $\mathcal{E}$. No further modification of the basic TDP algorithm is required. By the time the specification of the z-partition is completed, and the latest additions to the pool of tight, proper xy-edges have been dominated by paths through zero, the network is guaranteed to be decoupled.

The greedy, incremental approach to finding approximate solutions to the Allocative TDP requires that the metric h be able to measure the quality not only of fully decoupled networks, but also of *partially* decoupled networks. The metric h can be used not only to guide the Step 2 and Step 3 choice functions of the basic TDP algorithm (cf. Section 5.2.4), but also to guide incremental-allocation decision-making of the oracle function $\mathcal{F}$.

Definition 5.41 (Requirements for the Oracle Function) *Given an STN* $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ *and subnetworks* $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ *and* $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$, *where* $\mathcal{T}_X \cap \mathcal{T}_Y = \{z\}$ *and* $\mathcal{T}_X \cup \mathcal{T}_Y \subset \mathcal{T}$, *find sets* $\Delta\mathcal{T}_X$ *and* $\Delta\mathcal{T}_Y$ *such that:*

$$\left.\begin{array}{l} \Delta\mathcal{T}_X \subseteq (\mathcal{T} \setminus (\mathcal{T}_X \cup \mathcal{T}_Y)) \\ \Delta\mathcal{T}_Y \subseteq (\mathcal{T} \setminus (\mathcal{T}_X \cup \mathcal{T}_Y)) \end{array}\right\} \quad \textit{Can only allocate currently unallocated time-points.}$$

$(\Delta\mathcal{T}_X \cup \Delta\mathcal{T}_Y) \neq \emptyset$ Must *allocate at least one time-point*

$(\Delta\mathcal{T}_X \cap \Delta\mathcal{T}_Y) = \emptyset$ *Each time-point must be* uniquely *allocated*

Pseudo-code for the Allocative TDP algorithm is given in Figure 5.25. The algorithm takes as input a metric h, an oracle function $\mathcal{F}$, a consistent STN $\mathcal{S} = (\mathcal{T}, \mathcal{C})$, and sets $\mathcal{T}_X^0$ and $\mathcal{T}_Y^0$ that partially specify a z-partition of $\mathcal{T}$. The output of the algorithm is a pair of subnetworks, $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$, where $\mathcal{T}_X^0 \subseteq \mathcal{T}_X$ and $\mathcal{T}_Y^0 \subseteq \mathcal{T}_Y$, such that $\mathcal{S}_X$ and $\mathcal{S}_Y$ temporally decouple $\mathcal{S}$.

Each iteration of the algorithm (Steps 1 through 5) assumes that all pre-existing tight, proper xy-edges (i.e., tight, proper xy-edges with respect to the sets $\mathcal{T}_X$ and $\mathcal{T}_Y$) are already dominated by paths through zero. The two-part goal of each iteration is (1) to allocate additional time-points to $\mathcal{T}_X$ and $\mathcal{T}_Y$ and (2) to add intra-subnetwork constraints sufficient to ensure that all tight, proper xy-edges with respect to the sets $(\mathcal{T}_X \cup \Delta\mathcal{T}_X)$ and $(\mathcal{T}_Y \cup \Delta\mathcal{T}_Y)$ are dominated by paths through zero.

Since the input sets $\mathcal{T}_X^0$ and $\mathcal{T}_Y^0$ represent an initial, incremental allocation of time-points, the algorithm prepares for the first iteration by converting this initial allocation into the form expected in Step 2. In particular, $\mathcal{T}_X$ and $\mathcal{T}_Y$ are set to $\{z\}$, $\Delta\mathcal{T}_X$ is set to $(\mathcal{T}_X^0 \setminus \{z\})$, and $\Delta\mathcal{T}_Y$ is set to $(\mathcal{T}_Y^0 \setminus \{z\})$. Notice that, given this setup, there are *no* pre-existing tight, proper xy-edges (with respect to $\mathcal{T}_X$ and $\mathcal{T}_Y$). Thus, the assumption that all pre-existing tight, proper xy-edges are dominated by paths through zero holds trivially. Because the oracle function is not needed for the first iteration, the algorithm then jumps to Step 2. Subsequent iterations begin at Step 1, using the oracle function to generate the sets $\Delta\mathcal{T}_X$ and $\Delta\mathcal{T}_Y$ of time-points to add to $\mathcal{T}_X$ and $\mathcal{T}_Y$.

The rest of an iteration proceeds as follows. In Step 2, $\mathcal{E}$ is assigned to the set of all tight, proper xy-edges in $\mathcal{C}$ with respect to the sets $(\mathcal{T}_X \cup \Delta\mathcal{T}_X)$ and $(\mathcal{T}_Y \cup \Delta\mathcal{T}_Y)$ whose ZPS values are positive and which are *not* pre-existing xy-edges (i.e., are *not* xy-edges with respect to $\mathcal{T}_X$ and $\mathcal{T}_Y$). Thus, $\mathcal{E}$ contains edges only of the form $t_j - t_i \leq \delta$, where:

$$\begin{array}{lcll} t_i \in \mathcal{T}_X & \text{and} & t_j \in \Delta\mathcal{T}_Y & ; \\ t_i \in \Delta\mathcal{T}_X & \text{and} & t_j \in \mathcal{T}_Y & ; \text{ or} \\ t_i \in \Delta\mathcal{T}_X & \text{and} & t_j \in \Delta\mathcal{T}_Y & . \end{array}$$

The number of edges in $\mathcal{E}$ is bounded above by

$$|\mathcal{T}_X||\Delta\mathcal{T}_Y| + |\Delta\mathcal{T}_X||\mathcal{T}_Y| + |\Delta\mathcal{T}_X||\Delta\mathcal{T}_Y|$$

which, assuming $|\Delta\mathcal{T}_X| \ll |\mathcal{T}_X|$ and $|\Delta\mathcal{T}_Y| \ll |\mathcal{T}_Y|$, represents a small fraction of the xy-edges with respect to $(\mathcal{T}_X \cup \Delta\mathcal{T}_X)$ and $(\mathcal{T}_Y \cup \Delta\mathcal{T}_Y)$.

Once $\mathcal{E}$ has been initialized, Steps 2 through 6 of the basic TDP algorithm (Figure 5.8) are performed, after which all tight, proper xy-edges with respect to $(\mathcal{T}_X \cup \Delta\mathcal{T}_X)$ and $(\mathcal{T}_Y \cup \Delta\mathcal{T}_Y)$ are guaranteed to be dominated by paths through zero. (Recall that the basic TDP algorithm destructively modifies the STN $\mathcal{S}$ and its distance matrix $\mathcal{D}$.) The final step of the iteration is to add the time-points in $\Delta\mathcal{T}_X$ and $\Delta\mathcal{T}_Y$ to $\mathcal{T}_X$ and $\mathcal{T}_Y$, respectively.

Given: A metric h, an oracle function $\mathcal{F}$, a consistent STN $\mathcal{S} = (\mathcal{T}, \mathcal{C})$, and sets $\mathcal{T}_X^0$ and $\mathcal{T}_Y^0$ such that $\mathcal{T}_X^0 \cap \mathcal{T}_Y^0 = \{z\}$ and $\mathcal{T}_X^0 \cup \mathcal{T}_Y^0 \subseteq \mathcal{T}$.

Do the following:

(0) Initialization: $\Delta\mathcal{T}_X = (\mathcal{T}_X^0 \setminus \{z\}) \qquad \mathcal{T}_X = \{z\}$
$\Delta\mathcal{T}_Y = (\mathcal{T}_Y^0 \setminus \{z\}) \qquad \mathcal{T}_Y = \{z\}$

Go to Step 2.

(1) Apply the oracle function $\mathcal{F}$ to $\langle \mathcal{S},\ (\mathcal{T}_X, \mathcal{C}_d|_{\mathcal{T}_X}),\ (\mathcal{T}_Y, \mathcal{C}_d|_{\mathcal{T}_Y})\rangle$ to generate new sets $\Delta\mathcal{T}_X$ and $\Delta\mathcal{T}_Y$ of time-points that will be added to $\mathcal{T}_X$ and $\mathcal{T}_Y$ in Step 4.

(2) Set $\mathcal{E}$ to the set of *tight* edges in $\mathcal{C}$ having positive ZPS values and restricted to the form, $t_j - t_i \leq \delta$, where:

$$\begin{array}{lll} t_i \in \mathcal{T}_X & \text{and} & t_j \in \Delta\mathcal{T}_Y \ ;\ \text{or} \\ t_i \in \Delta\mathcal{T}_X & \text{and} & t_j \in \mathcal{T}_Y \ ;\ \text{or} \\ t_i \in \Delta\mathcal{T}_X & \text{and} & t_j \in \Delta\mathcal{T}_Y \ . \end{array}$$

(3) Run the basic TDP algorithm from Figure 5.8 starting at Step 2, using the metric h, if desired, to guide the Step 2 and Step 3 choice functions. (In the process, the STN $\mathcal{S}$ and its distance matrix $\mathcal{D}$ may be modified. The outputs $\mathcal{C}_X$ and $\mathcal{C}_Y$ of the basic TDP algorithm may be ignored.)

(4) Add the time-points in $\Delta\mathcal{T}_X$ and $\Delta\mathcal{T}_Y$ to $\mathcal{T}_X$ and $\mathcal{T}_Y$, respectively.

(5) If $(\mathcal{T}_X \cup \mathcal{T}_Y) \neq \mathcal{T}$, go back to Step 1; otherwise ...

Return: $(\mathcal{T}_X, \mathcal{C}_d|_{\mathcal{T}_X})$ and $(\mathcal{T}_Y, \mathcal{C}_d|_{\mathcal{T}_Y})$, where $\mathcal{C}$ includes all of the constraints added during the various calls to the basic TDP algorithm (from Step 3 above), and $\mathcal{D}$ has been updated accordingly. If desired, the Iterative Weakening algorithm from Section 5.2.6 may be applied to ensure that a minimal temporal decoupling is returned.

Figure 5.25: An algorithm for finding approximate solutions to the Allocative TDP

If some time-points in $\mathcal{T}$ have not yet been allocated, then at least one more iteration needs to be performed; otherwise, the algorithm is done and returns the subnetworks $(\mathcal{T}_X, \mathcal{C}_d|_{\mathcal{T}_X})$ and $(\mathcal{T}_Y, \mathcal{C}_d|_{\mathcal{T}_Y})$, which are guaranteed to temporally decouple $\mathcal{S}$.

Theorem 5.42 (Soundness, Completeness) *Given an oracle function $\mathcal{F}$ satisfying the requirements in Definition 5.41, the algorithm for finding approximate solutions to the Allocative Temporal Decoupling Problem is guaranteed to terminate with a temporal decoupling after at most $|\mathcal{T}|$ iterations.*

Proof At the end of each iteration of the Allocative TDP algorithm, edges in the current pool of tight, proper xy-edges are dominated by paths through zero. Since ZPS values cannot ever increase (cf. Lemma 5.17), the edges processed during one iteration never need to be revisited during any subsequent iteration. Thus, the main difference between the Allocative TDP algorithm and the basic TDP algorithm (cf. Section 5.2.4) is the *order* in which the tight, proper xy-edges are processed. Thus, the soundness result for the basic TDP algorithm (cf. Theorem 5.23) ensures that the Allocative TDP algorithm is sound.

Since the oracle function is required to allocate at least one time-point per iteration, the Allocative TDP algorithm is guaranteed to terminate after at most $|\mathcal{T}|$ iterations. ■

The greedy, incremental algorithm for finding approximate solutions to the Allocative Temporal Decoupling Problem is guided by the metric h of the quality of possibly-partial temporal decouplings and the oracle function $\mathcal{F}$ for making incremental allocation decisions. The metric h can be used to guide the Step 2 and Step 3 choice functions of the basic TDP algorithm. In addition, it can be used to guide the oracle function. The next section applies the Allocative TDP algorithm to the Auction-Participation-Context-Generation Problem described in Chapter 3. It provides a sample oracle function and discusses the dependence of the metric h on the application domain.

5.3.4 The APC-Generation Problem

Prior to participating in the type of task-allocation auction described in Chapter 3, an agent must decide what portion (i.e., subnetwork) of its Master STN to use as the basis for its auction-related computations. To reduce the computational burden of participating in an auction, the selected subnetwork, referred to as the agent's Auction-Participation-Context (APC), should be kept as small as possible. To allow the auction-related computations to be carried out independently, the APC subnetwork must be temporally decoupled from the rest of the Master STN.

In this section, $\mathcal{T}_X$ represents the time-points allocated to the APC subnetwork, $\mathcal{T}_Y$ the time-points allocated to remain with the Master STN, and $\mathcal{T}_U$ the time-points about which an allocation decision has not yet been made.

An Oracle Function for the APC-Generation Problem

A sample oracle function, $\mathcal{F}_{APC}$, for an application of the Allocative TDP algorithm to the APC-Generation Problem is given in Figure 5.26. The algorithm takes as input a partially

Given: STNs $\mathcal{S} = (\mathcal{T}, \mathcal{C})$, $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ and $\mathcal{S}_Y = (\mathcal{T}_Y, \mathcal{C}_Y)$ such that:

$$\mathcal{T}_X \subseteq \mathcal{T}; \qquad \mathcal{T}_Y \subseteq \mathcal{T}; \qquad \mathcal{T}_X \cap \mathcal{T}_Y = \{z\}; \qquad \text{and} \qquad \mathcal{T}_X \cup \mathcal{T}_Y \subset \mathcal{T},$$

and a metric h that measures the quality of (possibly partial) temporal decouplings.

(0) Let $\mathcal{T}_U = \mathcal{T} \setminus (\mathcal{T}_X \cup \mathcal{T}_Y)$ be the set of currently unallocated time-points in $\mathcal{T}$.

(1) Randomly select a time-point t_u from $\mathcal{T}_U$.

(2) Do Steps 2 and 3 of the Allocative TDP algorithm (Figure 5.25) using $\Delta\mathcal{T}_X = \{t_u\}$ and $\Delta\mathcal{T}_Y = \emptyset$. (Notice that constraints may be added to $\mathcal{C}$ in the process.)

(3) Let $h_X = h(\mathcal{S},\ (\mathcal{T}_X \cup \{t_u\}, \mathcal{C}_d|_{(\mathcal{T}_X \cup \{t_u\})}),\ (\mathcal{T}_Y, \mathcal{C}_d|_{\mathcal{T}_Y}))$.

(4) Restore $\mathcal{S}$ by removing any constraints from $\mathcal{C}$ that may have been added during Step 2 and recomputing the distance matrix $\mathcal{D}$.

(5) Do Steps 2 and 3 of the Allocative TDP algorithm (Figure 5.25) using $\Delta\mathcal{T}_X = \emptyset$ and $\Delta\mathcal{T}_Y = \{t_u\}$. (Once again, constraints may be added to $\mathcal{C}$ in the process.)

(6) Let $h_Y = h(\mathcal{S},\ (\mathcal{T}_X, \mathcal{C}_d|_{\mathcal{T}_X}),\ (\mathcal{T}_Y \cup \{t_u\}, \mathcal{C}_d|_{(\mathcal{T}_Y \cup \{t_u\})}))$.

(7) Restore $\mathcal{S}$ by removing any constraints from $\mathcal{C}$ that may have been added during Step 5 and recomputing the distance matrix $\mathcal{D}$.

Return: $\begin{cases} \Delta\mathcal{T}_X = \{t_U\} \text{ and } \Delta\mathcal{T}_Y = \emptyset \ , & \text{if } h_X \geq h_Y \\ \Delta\mathcal{T}_X = \emptyset \text{ and } \Delta\mathcal{T}_Y = \{t_U\} \ , & \text{otherwise.} \end{cases}$

Figure 5.26: An oracle function for an application of the Allocative TDP algorithm to the APC-Generation Problem

decoupled network (based on a partially specified z-partition) and returns as output sets, $\Delta\mathcal{T}_X$ and $\Delta\mathcal{T}_Y$, such that $|\Delta\mathcal{T}_X| + |\Delta\mathcal{T}_Y| = 1$. In other words, each time it is called, $\mathcal{F}_{APC}$ allocates a single time-point to either $\Delta\mathcal{T}_X$ or $\Delta\mathcal{T}_Y$.

In Step 1, a time-point t_u is randomly selected from the pool of currently unallocated time-points $\mathcal{T}_U$. In Step 2, an iteration of the Allocative TDP algorithm is executed to determine the effect of allocating t_u to $\Delta\mathcal{T}_X$. Notice that doing so typically involves adding constraints to $\mathcal{C}$ to ensure that any newly created tight, proper xy-edges are dominated by paths through zero. The quality of this (typically partial) decoupling is scored by the metric h and stored in the variable h_X (Step 3). In Step 4, $\mathcal{S}$ is restored to its original state. In Step 5, an iteration of the Allocative TDP algorithm is again executed, this time to determine the effect of allocating t_u to $\mathcal{T}_Y$. Once again, the metric h is used to score the quality of the resulting (typically partial) decoupling, this time storing the result in the variable h_Y. In Step 7, $\mathcal{S}$ is once again restored to its original state. Finally, $\mathcal{F}_{APC}$ returns the allocation decision that received a better score (i.e., $t_u \in \Delta\mathcal{T}_X$ or $t_u \in \Delta\mathcal{T}_Y$).

Choosing a Metric for the APC-Generation Problem

The purpose of the metric h is to measure the goodness of (typically partial) decouplings based on (possibly partial) allocations represented by the sets $\mathcal{T}_X$ and $\mathcal{T}_Y$. In general, the choice of metric depends on the preferences of the agent. In the context of the APC-Generation Problem, reasonable criteria to be considered by a metric include:

- The flexibility of the decoupled subnetworks; and
- The ability of the subnetwork corresponding to the auction-participation context to accommodate new tasks that the agent might be interested in bidding on.

If the APC subnetwork is highly flexible, then the agent will likely be better able to generate bids containing flexible temporal constraints, thereby increasing the likelihood of its bids being awardable. Similarly, if the rest of its Master STN is highly flexible, then an agent will likely be better able to accept offers to do other activities should they arise. However, the APC subnetwork must also be able to accommodate new tasks that the agent might want to bid on. The following example, illustrated in Figure 5.27, highlights some of the relevant issues.

Example. Suppose that $\mathcal{T}_X = \{z, t_x\}$ is the set of points currently allocated to the auction-participation context, and that $\mathcal{T}_Y = \{z, t_{y_1}, t_{y_2}\}$ is the set of points currently allocated to remain with the Master STN. Further suppose that t_{y_1} and t_{y_2} are rigidly connected such that t_{y_2} is constrained to occur precisely 10 minutes after t_{y_1}. (In Figure 5.27, the rigid constraint among t_{y_1} and t_{y_2} is represented by a thick, horizontal bar of length 10.) Let $I = [I_1, I_2]$ be a fixed interval on the time-line such that all of the tasks associated with the auction are constrained to lie within the interval I. Suppose that t_{y_1} is constrained to lie outside of I, but that t_{y_2} (although allocated to $\mathcal{T}_Y$) is constrained to lie within I. Let a and b be the lower and upper bounds on t_{y_1}. Similarly, let c and d be the lower and upper bounds on t_{y_2}. Notice that the rigid constraint among t_{y_1} and t_{y_2} implies that $c = a + 10$

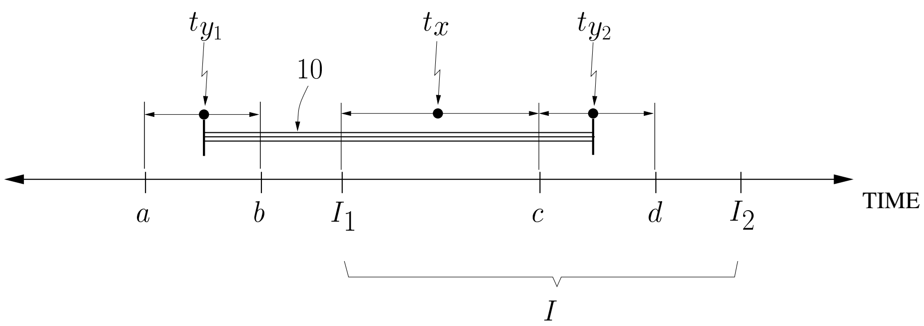


Figure 5.27: Sample scenario for the APC-Generation Problem

and $d = b + 10$. Finally, suppose that the lower and upper bounds for t_x are I_1 and c, respectively.

Notice that, as shown in Figure 5.27, the subnetworks corresponding to $\mathcal{T}_X$ and $\mathcal{T}_Y$ are temporally decoupled. Notice further that the flexibility afforded to t_{y_2} within the interval $[c, d]$ (subject to t_{y_1} moving in parallel within the interval $[a, b]$) is due to t_{y_2} having been allocated to $\mathcal{T}_Y$. If t_{y_2} had instead been allocated to $\mathcal{T}_X$, then temporally decoupling $\mathcal{T}_X$ and $\mathcal{T}_Y$ would have required fixing both t_{y_1} and t_{y_2} on the time-line. (In general, rigidly connected time-points may be decoupled only by fixing them on the time-line.)

Nonetheless, the flexibility afforded to t_{y_2} by having allocated it to $\mathcal{T}_Y$ comes with a cost, namely: it reduces the amount of space within the interval I for accommodating new tasks that the agent might want to bid on. For example, assuming that the agent cannot do two things at once, any time-point t^* associated with a new task that the agent might want to bid on must be constrained to lie outside the interval $[c, d]$. Otherwise, t^*, which is required to belong to $\mathcal{T}_X$, would introduce a dependence among $\mathcal{T}_X$ and $\mathcal{T}_Y$, thus subverting the decoupling. Thus, a metric h must consider the tradeoff between the flexibility of the decoupled subnetworks and the ability of the subnetwork representing the auction-participation context to accommodate new tasks that the agent might want to bid on. The Allocative TDP algorithm provides a structure within which a variety of candidate metrics may be applied.

Given a suitable metric h and the oracle function $\mathcal{F}_{APC}$, the Allocative TDP algorithm enables an agent to incrementally decide which points from its Master STN it will allocate to its auction-participation context. The basis for making each allocation decision may include not only the immediate impact on the flexibility of the decoupled networks, but also the amount of space in the auction-participation context for accommodating tasks that the agent might be interested in bidding on.

5.4 The General Temporal Decoupling Problem

This section generalizes the temporal decoupling problem presented in Section 5.2 to the case where the z-partition of $\mathcal{T}$ contains arbitrarily many subsets, $\mathcal{T}_1, \ldots, \mathcal{T}_n$. The treatment of the General TDP in Sections 5.4.1 through 5.4.4 (below) parallels the treatment of the $n = 2$ special case of the TDP presented earlier in Sections 5.2.1 through 5.2.4. For completeness, Sections 5.4.1 and 5.4.2 provide the definitions and theorems in the more general setting. However, the proofs, which are entirely analogous to their counterparts in Sections 5.2.1 and 5.2.2, are omitted. Since most of the definitions, theorems and proofs from Sections 5.2.3 and 5.2.4 (concerning xy-edges and xy-pairs) require only a minor terminological change to carry over to the more general setting, only the required terminological change is presented. Those results from Sections 5.2.3 and 5.2.4 that require more involved translation are presented explicitly, along with the General TDP algorithm itself, in Sections 5.4.3 and 5.4.4.

The General TDP algorithm has applications to the Post-Auction Coordination Problem for the sorts of task-allocation auctions described in Chapter 3. Such applications will be discussed at the end of this section.

5.4.1 Formal Definition of the General TDP

To simplify the presentation, the variables r and s are assumed to be restricted to the set $\{1, 2, \ldots, n\}$ in all that follows.

Definition 5.43 (z-Partition) *Let $\mathcal{T}, \mathcal{T}_1, \ldots, \mathcal{T}_n$ be sets of time-point variables. We say that $\mathcal{T}_1, \ldots, \mathcal{T}_n$* z-partition *$\mathcal{T}$ if:*

- $\mathcal{T}_r \cap \mathcal{T}_s = \{z\}$, *for all $r \neq s$; and*
- $\mathcal{T}_1 \cup \ldots \cup \mathcal{T}_n = \mathcal{T}$.

Figure 5.28 shows a sample z-partition involving four subsets of time-points.

Definition 5.44 (General Temporal Decoupling) $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1), \ldots, \mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$ *are called a* temporal decoupling *of the STN $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ if:*

- $\mathcal{S}_1, \ldots, \mathcal{S}_n$ *are consistent STNs;*
- $\mathcal{T}_1, \ldots, \mathcal{T}_n$ *z-partition $\mathcal{T}$; and*
- *(Mergeable Solutions Property)* Any *solutions $\mathcal{X}_1, \ldots, \mathcal{X}_n$ for $\mathcal{S}_1, \ldots, \mathcal{S}_n$, respectively, may be merged to form a solution for $\mathcal{S}$.*

Lemma 5.45 *If $\mathcal{S}_1, \ldots, \mathcal{S}_n$ are a temporal decoupling of $\mathcal{S}$, then $\mathcal{S}$ is consistent.*

Lemma 5.46 *If $\mathcal{S}_1, \ldots, \mathcal{S}_n$ are a temporal decoupling of $(\mathcal{T}, \mathcal{C} \cup \mathcal{C}')$, then $\mathcal{S}_1, \ldots, \mathcal{S}_n$ are also a temporal decoupling of $(\mathcal{T}, \mathcal{C})$.*

Lemma 5.47 *If $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1), \ldots, \mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$ are a temporal decoupling of $(\mathcal{T}, \mathcal{C})$, then $\mathcal{S}_1, \ldots, \mathcal{S}_n$ are also a temporal decoupling of $(\mathcal{T}, \mathcal{C} \cup \mathcal{C}_1 \cup \ldots \cup \mathcal{C}_n)$.*

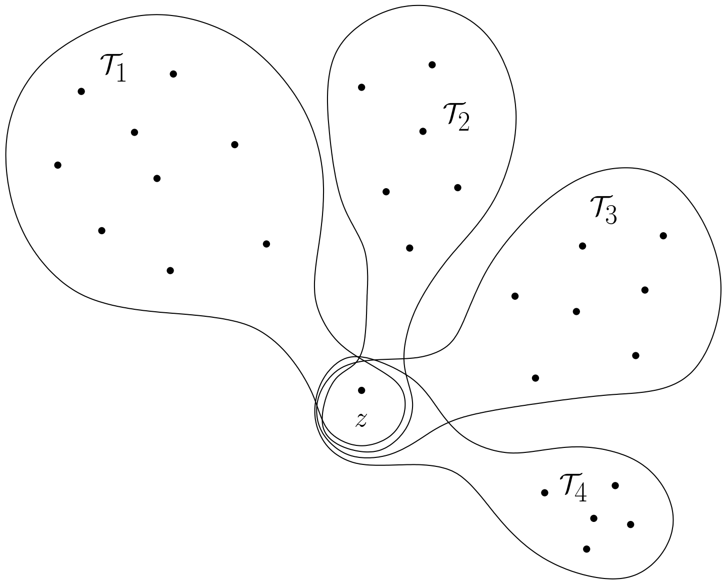


Figure 5.28: A Sample z-Partition

As with the case $n = 2$, the Mergeable Solutions Property is equivalent to the corresponding Mergeable Constraints Property.

Definition 5.48 (Mergeable Constraints Property) *Let* $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1)$, $\ldots$, $\mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$ *be consistent STNs such that* $\mathcal{T}_1, \ldots, \mathcal{T}_n$ *z-partition a set of time-points* $\mathcal{T}$*. We say that* $\mathcal{S}_1, \ldots, \mathcal{S}_n$ *have the* Mergeable Constraints Property *in the STN* $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ *if for any sets of constraints* $\Delta\mathcal{C}_1, \ldots, \Delta\mathcal{C}_n$ *over* $\mathcal{T}_1, \ldots, \mathcal{T}_n$*, respectively, if* $\Delta\mathcal{C}_1, \ldots, \Delta\mathcal{C}_n$ *are consistent with* $\mathcal{S}_1, \ldots, \mathcal{S}_n$*, respectively, then their union* $\Delta\mathcal{C}_1 \cup \ldots \cup \Delta\mathcal{C}_n$ *is consistent with* $\mathcal{S}$*.*

Theorem 5.49 (MSP $\equiv$ MCP) *Let* $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1)$, $\ldots$, $\mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$ *be consistent STNs such that* $\mathcal{T}_1, \ldots, \mathcal{T}_n$ *z-partition a set of time-points* $\mathcal{T}$*.* $\mathcal{S}_1, \ldots, \mathcal{S}_n$ *have the Mergeable Solutions Property in the STN* $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ *if and only if they have the Mergeable Constraints Property in* $\mathcal{S}$*.*

Definition 5.50 (The General Temporal Decoupling Problem) *Given an STN* $\mathcal{S}$ *whose time-points* $\mathcal{T}$ *are z-partitioned by* $\mathcal{T}_1, \ldots, \mathcal{T}_n$*, find sets of constraints* $\mathcal{C}_1, \ldots, \mathcal{C}_n$ *such that the STNs* $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1)$, $\ldots$, $\mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$ *temporally decouple* $\mathcal{S}$*.*

Lemma 5.51 *Any instance of the General TDP in which* $\mathcal{S}$ *is consistent has a solution.*

5.4.2 Necessary and Sufficient Characterizations of Solutions to the General TDP

Theorem 5.52 (Necessary Conditions) *If the STNs* $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1)$, $\ldots$, $\mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$ *are a temporal decoupling of the STN* $\mathcal{S} = (\mathcal{T}, \mathcal{C})$*, then the following must hold:*

(Property 1) For each r: $\mathcal{D}_r(x_i, x_j) \leq \mathcal{D}(x_i, x_j)$, for all $x_i, x_j \in \mathcal{T}_r$; and

(Property 2) For each r, s such that $r \neq s$: $\mathcal{D}_r(x_i, z) + \mathcal{D}_s(z, x_j) \leq \mathcal{D}(x_i, x_j)$, for all $x_i \in \mathcal{T}_r$ and $x_j \in \mathcal{T}_s$,

where $\mathcal{D}_1, \ldots, \mathcal{D}_n$ and $\mathcal{D}$ are the respective distance matrices for $\mathcal{S}_1, \ldots, \mathcal{S}_n$ and $\mathcal{S}$. If, in addition to the above, Property 1 holds with equality in all instances (i.e., $\mathcal{S}_r \equiv \mathcal{S}_d|_{\mathcal{T}_r}$ for each r), then Property 2 also holds with equality in all instances.

Corollary 5.53 *If $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1)$, $\ldots$, $\mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$ temporally decouple $(\mathcal{T}, \mathcal{C})$, then $\mathcal{S}_1, \ldots, \mathcal{S}_n$ temporally decouple $(\mathcal{T}, \mathcal{C} \cup \mathcal{C}_1 \cup \ldots \cup \mathcal{C}_n)$ such that Property 1 (and hence Property 2) from Theorem 5.52 holds with equality in all instances.*

Theorem 5.54 (Sufficient Conditions) *Let $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1)$, $\ldots$, $\mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$ and $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ be consistent STNs such that $\mathcal{T}_1, \ldots, \mathcal{T}_n$ z-partition $\mathcal{T}$. If Properties 1 and 2 of Theorem 5.52 hold, then $\mathcal{S}_1, \ldots, \mathcal{S}_n$ temporally decouple $\mathcal{S}$.*

5.4.3 Toward a General TDP Algorithm

In this section, the notions of xy-pairs and xy-edges are generalized to the notions of mixed pairs and mixed edges, respectively. Since the focus of the new definitions remains on pairs and edges, most of the definitions, results and proofs from Sections 5.2.3 and 5.2.4 carry over to the more general setting simply by replacing instances of the terms "xy-pairs" and "xy-edges" with the terms "mixed pairs" and "mixed edges", respectively.

Definition 5.55 (Mixed Pairs, Mixed Edges) *Let $\mathcal{T}_1, \ldots, \mathcal{T}_n$ and $\mathcal{T}$ be sets of time-points such that $\mathcal{T}_1, \ldots, \mathcal{T}_n$ 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 a* mixed pair *(with respect to that z-partition) if for some $r \neq s$:*

$$t_i \in \mathcal{T}_r \quad \text{and} \quad t_j \in \mathcal{T}_s.$$

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

Definition 5.14 (Zero-Path Shortfall), Lemma 5.15 (ZPS ≥ 0), Definition 5.16 (Dominated by a path through zero), Lemma 5.17 (ZPS values cannot increase), and Lemma 5.18 (sufficient to consider only tight, proper xy-edges) from Section 5.2.3 all carry over to the more general setting simply by replacing instances of the terms "xy-pairs" and "xy-edges" with the terms "mixed pairs" and "mixed edges", respectively. (This holds for the proofs, too.) Thus, those definitions and results are not repeated here.

5.4.4 Algorithms for Solving the General TDP

Pseudo-code for the basic TDP algorithm (general case) is given in Figure 5.29. The algorithm returns sets of constraints $\mathcal{C}_1, \ldots, \mathcal{C}_n$ such that the corresponding STNs $\mathcal{S}_1, \ldots, \mathcal{S}_n$ are d-subnetworks of $\mathcal{S}$ with respect to $\mathcal{T}_1, \ldots, \mathcal{T}_n$, respectively.

Given: An STN $\mathcal{S}$ whose time-points $\mathcal{T}$ are z-partitioned by the sets $\mathcal{T}_1, \ldots, \mathcal{T}_n$.

(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, mixed edges in $\mathcal{S}$, and continue.

(2) Select a tight, proper, mixed edge $E = (t_j - t_i \leq \delta)$ from $\mathcal{E}$ 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 the STN $\mathcal{S}$, updating $\mathcal{D}$ accordingly.

(5) Go to Step 2.

(6) Return $\mathcal{C}_1, \ldots, \mathcal{C}_n$ where for each r,

$$\mathcal{C}_r \;=\; \mathcal{C}_d|_{\mathcal{T}_r} \;=\; \{(t_j - t_i \leq \mathcal{D}(t_i, t_j)) : t_i, t_j \in \mathcal{T}_r\},$$

where $\mathcal{C}$ includes all constraints added in passes of Step 4, and $\mathcal{D}$ has been updated accordingly.

Figure 5.29: Pseudo-code for the basic TDP algorithm (general case)

The same terminological change ("xy-pairs/edges" $\mapsto$ "mixed pairs/edges") that enabled most of the definitions, results and proofs from Section 5.2.3 to carry over to Section 5.4.3 also enables the following results from Section 5.2.4 to carry over to this section:

- Theorem 5.19: adding the Step 4 constraints does not threaten the STN's consistency;
- Corollary 5.20: after adding the Step 4 constraints, δ_1 and δ_2 are the updated distance-matrix entries;
- Lemma 5.21: the regions Ω and Θ are related by invertible transformations;
- Corollary 5.22: it is always possible to choose values of δ_1 and δ_2 in Step 3 to achieve any legal R and α values; and
- Proposition 5.25: any progress made by the algorithm is never lost.

The theorems stipulating the soundness and completeness of the TDP algorithm in the general setting are given below.

Lemma 5.56 (Soundness) *If the General TDP algorithm terminates at Step 6, then the constraint sets* $\mathcal{C}_1, \ldots, \mathcal{C}_n$ *returned by the algorithm are such that* $(\mathcal{T}_1, \mathcal{C}_1), \ldots, (\mathcal{T}_n, \mathcal{C}_n)$ *are a general temporal decoupling of the input STN* $\mathcal{S}$.

The Greedy and Less-Greedy strategies for ensuring that substantial progress is made during each iteration of the algorithm are unchanged. However, the maximum number of iterations are:

Greedy Strategy	At most $2\Pi_{i=1}^{n}\vert\mathcal{T}_i\vert$	iterations
Less-Greedy Strategy	At most $2\Pi_{i=1}^{n}\vert\mathcal{T}_i\vert \left\lceil \left\lceil \frac{log(Z/\epsilon)}{log(1/(1-r))} \right\rceil + 1 \right\rceil$	iterations

Corollary 5.57 (Sound and Complete) *Using the Greedy Strategy (cf. Definition 5.24) or the Less-Greedy Strategy (cf. Definition 5.27), the General TDP algorithm is sound and complete.*

5.4.5 An Application of the General TDP Algorithm to the Post-Auction-Coordination Problem

As described in Chapter 3, when bids in a task-allocation auction are awarded, the temporal constraints from all of the bids are guaranteed to be consistent with the temporal constraints associated with the original opportunity; however, there is no requirement that the tasks being awarded to different agents be temporally decoupled. For example, if I have been awarded a bid concerning a dish-washing task constrained to occur in the interval $[0, 20]$ and you have been awarded a bid concerning the corresponding dish-drying task, similarly constrained to occur in the interval $[0, 20]$, but also constrained to occur after the dish-washing task, then our tasks are not independent. Figure 5.30 shows an STN representing our activities. Notice that the edge from y to x is not dominated by a path through zero. (For simplicity, the tasks are assumed to have zero duration.)

The General TDP algorithm may be directly applied in such situations to achieve a complete decoupling of the tasks being done by different agents. The advantages of such an approach include the following:

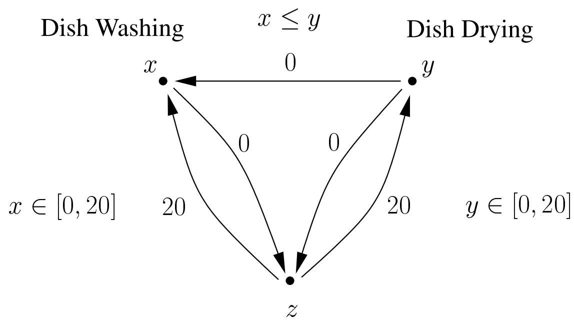


Figure 5.30: An STN representing dependent dish washing and drying activities

- The agents may henceforth operate independently, each confident that no matter what constraints it might choose to add to its local subnetwork, the global constraints will necessarily be satisfied (cf. the Mergeable Solutions Property, Definition 5.44).
- The General TDP algorithm is sound and complete, easy to implement, and runs in polynomial time.

Furthermore, an extension of the Iterative Weakening algorithm (cf. Section 5.2.6) to the case of multiple subnetworks can be applied to any decoupling to generate a *minimal* decoupling.

Despite these advantages, fully decoupling the tasks being done by different agents does require adding constraints to the network. In some cases, agents might prefer to accept some amount of dependence among tasks (and the requisite coordination overhead) in exchange for greater flexibility in their local subnetworks. The Relative Temporal Decoupling Problem described in the next section, formally addresses this issue.

5.5 The Relative Temporal Decoupling Problem

In certain applications (e.g., the Post-Auction-Coordination Problem), it can be advantageous to fully decouple a set of subnetworks that, taken together, do not constitute the entire network. For example, if agents G_1 and G_2 both must finish their tasks β_1 and β_2 prior to agent G_3 being able to start its task β_3, it may be advantageous to decouple β_1 and β_2 from one another, but not from β_3. Doing so would leave G_1 and G_2 able to act independently of one another, while leaving G_3 dependent on both of them, as illustrated in Figure 5.31. Such a decoupling is called a *Relative Temporal Decoupling*. In such a decoupling, the subnetworks being decoupled, taken together, do not constitute the entire network. The leftover time-points (i.e., those that do not belong to any of the subnetworks being decoupled) constitute what is herein called the *relativizing set*, denoted $\mathcal{T}_W$. In a relative temporal decoupling, the subnetworks are said to be fully decoupled *relative* to the set $\mathcal{T}_W$ of leftover time-points.

The Relative Temporal Decoupling Problem bears a superficial resemblance to a single iteration of the Allocative TDP algorithm in that each treats a set of subnetworks that, taken

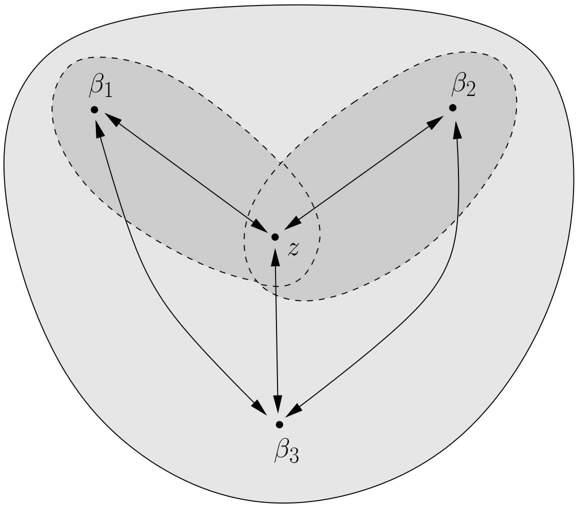


Figure 5.31: A sample relative temporal decoupling

together, do not constitute the entire network. However, after a single iteration of the Allocative TDP algorithm, the subnetworks are typically only *partially* decoupled from one another, whereas a solution to the Relative TDP requires the subnetworks to be *fully* decoupled from one another. Stated differently, a solution to the Relative TDP is required to have a Mergeable Solutions Property (relative to the set $\mathcal{T}_W$), whereas the partially decoupled subnetworks generated by a single iteration of the Allocative TDP algorithm typically have no such property.[5] In addition, the goal of the Allocative TDP algorithm is to incrementally construct a set of fully decoupled subnetworks that, taken together, *do* constitute the entire network, whereas the subnetworks generated by a solution to the Relative TDP, taken together, do *not* constitute the entire network. The set of leftover time-points in $\mathcal{T}_W$ remain "unallocated" in any solution to the Relative TDP.

Because the set of time-points in $\mathcal{T}_W$ are not decoupled from the other subnetworks, there is an inherent asymmetry in a relative temporal decoupling (which mirrors an inherent asymmetry present in many applications). In particular, the agents controlling the subnetworks that are fully decoupled from one another relative to $\mathcal{T}_W$ may act independently of one another—and independently of the agent G_W controlling the time-points in the set $\mathcal{T}_W$; however, the agent G_W does not have the same freedom. In particular, G_W is not allowed to impose any additional constrainedness on any of the other subnetworks. As a result, G_W is not free to blindly apply the ordinary STN rules for adding new constraints (e.g., as specified in Theorem 4.33). G_W must also observe an additional set of bounds, called *lambda bounds*, which are explicated below, in Section 5.5.5.

[5] At the end of an intermediate iteration of the Allocative TDP algorithm, even though every tight, proper xy-edge *that currently exists* is dominated by a path through zero, there might yet be some shortest *path* from some $t_x \in \mathcal{T}_X$ to some $t_y \in \mathcal{T}_Y$ that passes through some of the points not yet allocated to $\mathcal{T}_X$ or $\mathcal{T}_Y$, but that is not dominated by a path through zero (cf. the path from t_x to t_u to t_y in Figure 5.20). In such cases, the Mergeable Solutions Property does not hold.

In addition to its application to the Post-Auction-Coordination Problem, the tighter set of bounds that the agent G_W must observe in a relative temporal decoupling provides a solution to the question of how an agent should accommodate the situation of having outstanding bids. In that case, the auctioneer, who is authorized to apply additional constraints to an agent's bid, plays the role of an independent actor, while the bidder plays the role of the agent G_W, who is in part dependent on the decisions of the auctioneer.

In contrast to the General TDP, where it only makes sense to consider decouplings involving two or more subnetworks (i.e., $n \geq 2$), for the Relative TDP, the presence of the relativizing set $\mathcal{T}_W$ makes the case $n = 1$ non-trivial. In fact, the case $n = 1$ is directly applicable to the Temporal-Constraint-Generation problem, as discussed in Section 5.5.6.

Finally, the analysis in this section reduces the problem of finding an algorithm for the Relative Temporal Decoupling Problem to the problem of finding a set of *redundant* constraints to add to the global network to trick the General TDP algorithm into doing the right thing.

The rest of this section is organized as follows. Section 5.5.1 formally defines the Relative TDP. Section 5.5.2 presents theorems characterizing solutions to instances of the Relative TDP. Sections 5.5.3 and 5.5.4 develop a sound and complete algorithm for solving the Relative TDP. Section 5.5.5 presents the *lambda bounds* relevant to the agent controlling the leftover time-points in $\mathcal{T}_W$. Section 5.5.6 shows how the relative TDP algorithm applies to the Temporal-Constraint-Generation problem. Section 5.5.7 shows how the relative TDP algorithm can be used to provide more flexible solutions to the Post-Auction-Coordination Problem.

Although the formal presentation in Sections 5.5.1 through 5.5.4 parallels the corresponding presentations for the TDP and the General TDP, the presence of the relativizing set $\mathcal{T}_W$ introduces additional complications. As a result, full proofs are provided for many of the theorems.

5.5.1 Formal Definition of the Relative TDP

We begin by defining a z-partition relative to a set $\mathcal{T}_W$ of time-points. Unlike the other sets involved in a relative z-partition, the relativizing set $\mathcal{T}_W$ does *not* contain z. Thus, $\mathcal{T}_W$ is given a distinguished place in the notation.

Definition 5.58 (Relative z-Partition) *Let $\mathcal{T}, \mathcal{T}_1, \ldots, \mathcal{T}_n$ and $\mathcal{T}_W$ be sets of time-point variables. A* z-partition *of $\mathcal{T}$* relative to *$\mathcal{T}_W$ is an ordered sequence $(\mathcal{T}_1, \ldots, \mathcal{T}_n;\ \mathcal{T}_W)$ such that the sets $\mathcal{T}_1, \ldots, \mathcal{T}_n$ z-partition the set $\mathcal{T} \setminus \mathcal{T}_W$. The sets $\mathcal{T}_1, \ldots, \mathcal{T}_n$ are said to z-partition $\mathcal{T}$ relative to $\mathcal{T}_W$.*

Figure 5.32 shows a z-partition involving the sets $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3$ and $\mathcal{T}_W$.

Definition 5.59 (Extendible Partial Solution) *Let $\mathcal{S}$ be an STN over the time-points in $\mathcal{T}$. Let $\mathcal{T}' \subseteq \mathcal{T}$ be a set of time-points. An* extendible partial solution for *$\mathcal{S}$* over *$\mathcal{T}'$ is a set of assignments for the time-point variables in $\mathcal{T}'$ that is extendible to a solution for $\mathcal{S}$. In the case where $\mathcal{T}_W = \mathcal{T} - \mathcal{T}'$, then $(\mathcal{T}'; \mathcal{T}_W)$ is a relative z-partition of $\mathcal{T}$ and the above set of*

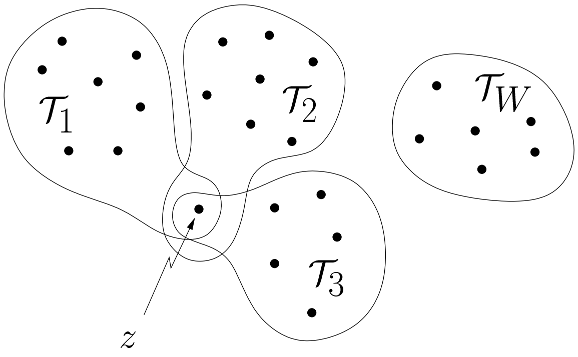


Figure 5.32: A sample relative z-partition

variable assignments may be called a partial solution for $\mathcal{S}$ relative to $\mathcal{T}_W$ *(i.e., highlighting the points* not *yet assigned values).*

Proposition 5.60 *By Theorems 4.26 and 4.33, a set $\mathcal{X}|_{\mathcal{T}'}$ of assignments to time-point variables in $\mathcal{T}'$ is an extendible partial solution for $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ over $\mathcal{T}$ if and only if $\mathcal{X}|_{\mathcal{T}'}$ is locally consistent with respect to the constraints in $\mathcal{C}_d$.*

Notice that an extendible partial solution for $\mathcal{S}$ relative to the empty set is simply a solution for $\mathcal{S}$. Notice also that an extendible partial solution for $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ over $\mathcal{T}'$ is necessarily a locally consistent assignment with respect to $\mathcal{C}$ (cf. Definition 4.26), but that the converse need not hold.

Definition 5.61 (Relative Temporal Decoupling) $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1), \ldots, \mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$ *are said to* temporally decouple $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ relative to $\mathcal{T}_W$ *if:*

- $(\mathcal{T}_1, \ldots, \mathcal{T}_n; \mathcal{T}_W)$ *is a relative z-partition of* $\mathcal{T}$*;*
- $\mathcal{S}_1, \ldots, \mathcal{S}_n$ *are consistent STNs; and*
- *(Relative Mergeable Solutions Property) Any solutions $\mathcal{X}_1, \ldots, \mathcal{X}_n$ to $\mathcal{S}_1, \ldots, \mathcal{S}_n$, may be merged to form an extendible partial solution for $\mathcal{S}$ relative to $\mathcal{T}_W$.*

Notice that a temporal decoupling relative to the empty set is simply a temporal decoupling (cf. Definition 5.44).

Lemma 5.62 *If $\mathcal{S}_1, \ldots, \mathcal{S}_n$ temporally decouple $\mathcal{S}$ relative to $\mathcal{T}_W$, then $\mathcal{S}$ is consistent.*

Lemma 5.63 *If $\mathcal{S}_1, \ldots, \mathcal{S}_n$ temporally decouple $(\mathcal{T}, \mathcal{C} \cup \mathcal{C}')$ relative to $\mathcal{T}_W$, then $\mathcal{S}_1, \ldots, \mathcal{S}_n$ also temporally decouple $(\mathcal{T}, \mathcal{C})$ relative to $\mathcal{T}_W$.*

Lemma 5.64 *If $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1), \ldots, \mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$ temporally decouple $(\mathcal{T}, \mathcal{C})$ relative to $\mathcal{T}_W$, then $\mathcal{S}_1, \ldots, \mathcal{S}_n$ also temporally decouple $(\mathcal{T}, \mathcal{C} \cup \mathcal{C}_1 \cup \ldots \cup \mathcal{C}_n)$ relative to $\mathcal{T}_W$.*

Definition 5.65 (Relative Mergeable Constraints Property) *Given consistent STNs, $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1),\ \ldots,\ \mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$, where $(\mathcal{T}_1, \ldots, \mathcal{T}_n; \mathcal{T}_W)$ is a relative z-partition of $\mathcal{T}$, then $\mathcal{S}_1, \ldots, \mathcal{S}_n$ are said to have the Mergeable Constraints Property in the STN $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ relative to $\mathcal{T}_W$ if whenever sets of constraints $\Delta\mathcal{C}_1, \ldots, \Delta\mathcal{C}_n$ over the time-points in $\mathcal{T}_1, \ldots, \mathcal{T}_n$, respectively, are consistent with $\mathcal{S}_1, \ldots, \mathcal{S}_n$, respectively, their union $\Delta\mathcal{C}_1 \cup \ldots \cup \Delta\mathcal{C}_n$ is consistent with $\mathcal{S}$.*

Theorem 5.66 (Relative MSP $\equiv$ Relative MCP) *Let $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1),\ \ldots,\ \mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$ be consistent STNs such that $(\mathcal{T}_1, \ldots, \mathcal{T}_n; \mathcal{T}_W)$ is a relative z-partition of $\mathcal{T}$. Then $\mathcal{S}_1, \ldots, \mathcal{S}_n$ have the Mergeable Solutions Property in $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ relative to $\mathcal{T}_W$ if and only if they have the Mergeable Constraints Property in $\mathcal{S}$ relative to $\mathcal{T}_W$.*

Definition 5.67 (The Relative Temporal Decoupling Problem) *Given an STN $\mathcal{S}$ whose time-points $\mathcal{T}$ are z-partitioned by $\mathcal{T}_1, \ldots, \mathcal{T}_n$ relative to $\mathcal{T}_W$, find sets of constraints $\mathcal{C}_1,\ \ldots,\ \mathcal{C}_n$ such that the STNs $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1),\ \ldots,\ \mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$ temporally decouple $\mathcal{S}$ relative to $\mathcal{T}_W$.*

Notice that the Relative TDP in the case where $\mathcal{T}_W = \emptyset$ is simply the General TDP (cf. Definition 5.50).

Lemma 5.68 *Any instance of the Relative TDP in which $\mathcal{S}$ is consistent has a solution.*

Although it is easy to construct solutions to the Relative TDP by employing constraints that *fix* all of the time-points in $\mathcal{T}_1, \ldots, \mathcal{T}_n$, it would be preferable to find more flexible solutions.

5.5.2 Necessary and Sufficient Characterizations of Solutions to the Relative TDP

Theorem 5.69 (Necessary Conditions) *If $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1), \ldots, \mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$ temporally decouple $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ relative to the set $\mathcal{T}_W$, then the following properties hold:*

(Property 1) For each r: $\mathcal{D}_r(t_i, t_j) \leq \mathcal{D}(t_i, t_j)$, for each $t_i, t_j \in \mathcal{T}_r$; and

*(Property 2) For each r, s such that $r \neq s$: $\mathcal{D}_r(t_i, z) + \mathcal{D}_s(z, t_j) \leq \mathcal{D}(t_i, t_j)$,
for each $t_i \in \mathcal{T}_r$ and each $t_j \in \mathcal{T}_s$,*

where (for any r and s), $\mathcal{D}_r, \mathcal{D}_s$ and $\mathcal{D}$ are the distance matrices for the respective STNs $\mathcal{S}_r, \mathcal{S}_s$ and $\mathcal{S}$.

Proof

(Property 1) Given any r and any $t_i, t_j \in \mathcal{T}_r$, let $\mathcal{X}_r$ be a solution to $\mathcal{S}_r$ in which $t_j - t_i = \mathcal{D}_r(t_i, t_j)$ (cf. Theorem 4.33). For each $s \neq r$, let $\mathcal{X}_s$ be an arbitrary solution for $\mathcal{S}_s$. The Relative Mergeable Solutions Property ensures that the union of the solutions $\mathcal{X}_1, \ldots, \mathcal{X}_n$ is extendible to a solution for $\mathcal{S}$. Let $\mathcal{X}$ be such a solution. In that solution, $t_j - t_i = \mathcal{D}_r(t_i, t_j)$. In addition, being a solution for $\mathcal{S}$ implies that $t_j - t_i \leq \mathcal{D}(t_i, t_j)$. Thus, $\mathcal{D}_r(t_i, t_j) = t_j - t_i \leq \mathcal{D}(t_i, t_j)$.

(Property 2) Given any r and s with $r \neq s$, and any $t_i \in \mathcal{T}_r$ and $t_j \in \mathcal{T}_s$, let $\mathcal{X}_r$ and $\mathcal{X}_s$ be solutions to $\mathcal{S}_r$ and $\mathcal{S}_s$, respectively, in which $z - t_i = \mathcal{D}_r(t_i, z)$ and $t_j - z = \mathcal{D}_s(z, t_j)$. Let $\mathcal{X}$ be a solution for $\mathcal{S}$ extended from the merger of $\mathcal{X}_r$, $\mathcal{X}_s$, and arbitrary solutions for all $\mathcal{S}_q$, $q \notin \{r, s\}$. In the solution, $\mathcal{X}$:

$$t_j - t_i \;=\; (z - t_i) + (t_j - z) \;=\; \mathcal{D}_r(t_i, z) + \mathcal{D}_s(z, t_j).$$

In addition, being a solution for $\mathcal{S}$ implies that $t_j - t_i \leq \mathcal{D}(t_i, t_j)$. Thus,

$$\mathcal{D}_r(t_i, z) + \mathcal{D}_s(z, t_j) \leq \mathcal{D}(t_i, t_j). \;\blacksquare$$

Theorem 5.70 (Sufficient Conditions) *Let* $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1)$, $\ldots$, $\mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$ *and* $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ *be consistent STNs, and* $\mathcal{T}_W$ *a set of time-points such that* $\mathcal{T}_1, \ldots, \mathcal{T}_n$ *z-partition* $\mathcal{T}$ *relative to* $\mathcal{T}_W$. *If Properties 1 and 2 of Theorem 5.69 hold in all instances, then* $\mathcal{S}_1, \ldots, \mathcal{S}_n$ *temporally decouple* $\mathcal{S}$ *relative to* $\mathcal{T}_W$.

Proof Given the conditions of the theorem, it is sufficient to verify that the Relative Mergeable Solutions Property holds (i.e., that the union of arbitrary solutions for $\mathcal{S}_1, \ldots, \mathcal{S}_n$ is extendible to a solution for $\mathcal{S}$). Let $\mathcal{X}_1, \ldots, \mathcal{X}_n$ be arbitrary solutions for $\mathcal{S}_1, \ldots, \mathcal{S}_n$, respectively. Let $\mathcal{X} = \mathcal{X}_1 \cup \ldots \cup \mathcal{X}_n$. For any r and s, and any $t_i \in \mathcal{T}_r$ and $t_j \in \mathcal{T}_s$:

- If $r = s$, then $t_j - t_i \leq \mathcal{D}_r(t_i, t_j)$ (since $\mathcal{X}_r$ is a solution for $\mathcal{S}_r$). Property 1 then gives that $t_j - t_i \leq \mathcal{D}(t_i, t_j)$.
- If $r \neq s$, then $t_j - t_i = (z - t_i) + (t_j - z) \leq \mathcal{D}_r(t_i, z) + \mathcal{D}_s(z, t_j)$ (since $\mathcal{X}_r$ and $\mathcal{X}_s$ are solutions for $\mathcal{S}_r$ and $\mathcal{S}_s$, respectively). Property 2 then gives that $t_j - t_i \leq \mathcal{D}(t_i, t_j)$.

Thus, for any t_i, t_j drawn from the set $\mathcal{T}' = \mathcal{T}_1 \cup \ldots \cup \mathcal{T}_n$, we have that $t_j - t_i \leq \mathcal{D}(t_i, t_j)$. In other words, $\mathcal{X}$ is a locally consistent assignment of the variables in $\mathcal{T}'$ with respect to the constraints in $\mathcal{C}_d$. Thus, by Theorem 4.27, the set $\mathcal{X}$ is extendible to a solution for $\mathcal{S}$. Since $\mathcal{X}_1, \ldots, \mathcal{X}_n$ were arbitrary solutions for $\mathcal{S}_1, \ldots, \mathcal{S}_n$, the Relative Mergeable Solutions Property necessarily holds. $\blacksquare$

5.5.3 Toward a Relative TDP Algorithm

The presence of the relativizing set $\mathcal{T}_W$ makes a noticeable difference in the generalization of the notion of a mixed edge with respect to a z-partition (cf. Definition 5.55) to the notion of a mixed *path* with respect to a *relative* z-partition. Particularly important is that a relative mixed path joining some $t_i \in \mathcal{T}_r$ to some $t_j \in \mathcal{T}_s$ may contain time-points in the relativizing set $\mathcal{T}_W$. In fact, the only points allowed to be outside the set $\mathcal{T}_W$ are the endpoints of a mixed path, as illustrated in Figure 5.33.

Definition 5.71 (Relative Mixed Path / Relative Mixed Pair) *Let* $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ *be an STN. Let* $\mathcal{T}_1, \ldots, \mathcal{T}_n$ *and* $\mathcal{T}_W$ *be sets of time-points such that* $(\mathcal{T}_1, \ldots, \mathcal{T}_n; \mathcal{T}_W)$ *is a relative z-partition of* $\mathcal{T}$. *Let* r *and* s *be arbitrary such that* $r \neq s$. *Let* $t_i \in \mathcal{T}_r$ *and* $t_j \in \mathcal{T}_s$ *be arbitrary. A path* P *of the form* $(t_i, w_1, w_2, \ldots, w_k, t_j)$ *is called a* mixed path in $\mathcal{S}$ *relative to* $\mathcal{T}_W$ *if:* $k \geq 0$ *and* $\{w_1, w_2, \ldots, w_k\} \subseteq \mathcal{T}_W$.

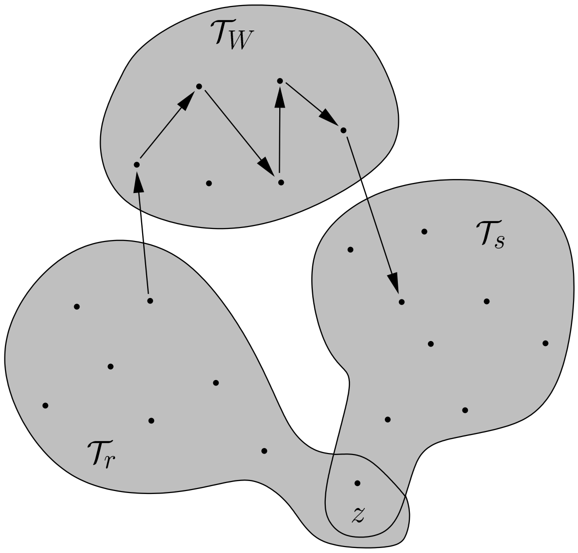


Figure 5.33: A sample *mixed path*

If neither t_i nor t_j is the z time-point, then P is called a proper, *mixed path in $\mathcal{S}$ relative to $\mathcal{T}_W$. (Since $z \notin \mathcal{T}_W$, z cannot appear anywhere along a proper, mixed path relative to $\mathcal{T}_W$.) If, in addition, P is a shortest path from t_i to t_j in $\mathcal{S}$, then P is called a* tight, proper, mixed path *in $\mathcal{S}$ relative to $\mathcal{T}_W$. The endpoints, (t_i, t_j), of a (proper) mixed path P relative to $\mathcal{T}_W$ are called a (resp., proper) mixed pair relative to $\mathcal{T}_W$.*

Notice that if $k = 0$, then a mixed path corresponds to a mixed edge (cf. Definition 5.55).

Note. Although we speak of mixed paths relative to some set $\mathcal{T}_W$, it must be kept in mind that the definition of a mixed path depends on a specified relative z-partition (because the endpoints of each mixed path must belong to different sets $\mathcal{T}_r$ and $\mathcal{T}_s$ from the relative z-partition). Since this relative z-partition is usually clear from context, we typically omit the reference to it when speaking of mixed paths.

Definition 5.72 (Relative Zero-Path Shortfall) *The* zero-path shortfall *associated with a proper, mixed path P from t_i to t_j is given by:*

$$\mathrm{ZPS}(P) \;=\; [\mathcal{D}(t_i, z) + \mathcal{D}(z, t_j)] - |P|.$$

Definition 5.73 (Dominance by a Path through Zero) *If the ZPS value for a tight, proper, mixed path P relative to the set $\mathcal{T}_W$ is zero, then we say that P is dominated by a path through zero.*

Lemma 5.74 shows that it suffices to consider only tight, proper, mixed paths when seeking a solution to an instance of the Relative TDP.

Lemma 5.74 *Let* $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ *be a consistent STN and let* $(\mathcal{T}_1, \ldots, \mathcal{T}_n; \mathcal{T}_W)$ *be some relative z-partition of* $\mathcal{T}$ *that defines the mixed paths and mixed pairs in* $\mathcal{S}$*. If the inequality* $\mathcal{D}(t_i, z) + \mathcal{D}(z, t_j) \leq |P|$ *holds for every tight, proper, mixed* path P*, where* t_i *and* t_j *are the first and last time-points in* P*, respectively, then* $\mathcal{D}(t_p, z) + \mathcal{D}(z, t_q) \leq \mathcal{D}(t_p, t_q)$ *holds for every mixed* pair (t_p, t_q) *in* $\mathcal{S}$*.*

Proof Let (t_p, t_q) be an arbitrary mixed pair in $\mathcal{S}$ relative to $\mathcal{T}_W$ where $t_p \in \mathcal{T}_r$ and $t_q \in \mathcal{T}_s$. It suffices to show that $\mathcal{D}(t_p, z) + \mathcal{D}(z, t_q) \leq \mathcal{D}(t_p, t_q)$ holds.

Let $P = (v_0, v_1, \ldots, v_c, v_{c+1})$ be an arbitrary *shortest* path from t_p to t_q in $\mathcal{S}$. If z is on this path, then the desired inequality holds (because the subpaths from v_0 to z and from z to v_{c+1} must be shortest paths by Proposition 4.20). Suppose z is not on P. Let a be the largest index such that $v_a \in \mathcal{T}_x$ for some $x \neq s$. (a is well-defined since $v_0 \in \mathcal{T}_r$ and $r \neq s$.) Similarly, let b be the smallest index such that $a < b$, $v_b \in \mathcal{T}_y$ and $y \neq x$. (b is well-defined since $a < c + 1$, $v_{c+1} \in \mathcal{T}_s$ and $s \neq x$.) Let P' be the subpath of P from v_a to v_b. By construction, any time-points between v_a and v_b in P must be in $\mathcal{T}_W$; thus, P' is a mixed path relative to $\mathcal{T}_W$. Since z is not on P (by assumption), P' is a proper, mixed path. Since P' is a subpath of a shortest path, P' must itself be a shortest path (by Proposition 4.20). Thus, P' is a tight, proper, mixed path from v_a to v_b relative to $\mathcal{T}_W$. Thus, by the premise of the Lemma, $\mathcal{D}(v_a, z) + \mathcal{D}(z, v_b) \leq |P'|$. However, since P' is a shortest path, the opposite inequality also holds. Thus, $\mathcal{D}(v_a, z) + \mathcal{D}(z, v_b) = |P'|$. But this implies that P' may be replaced in P by a pair of shortest paths, one from v_a to z, one from z to v_b, without changing the length of P. Since z is on this version of the shortest path P from v_0 to v_{c+1} (i.e., from t_p to t_q), the desired inequality holds (as argued in the earlier case). ∎

5.5.4 Algorithms for Solving the Relative TDP

Theorem 5.75 shows how to use the General TDP algorithm to solve instances of the Relative TDP.

Theorem 5.75 *Let* $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ *be a consistent STN whose time-points* $\mathcal{T}$ *are z-partitioned by* $\mathcal{T}_1, \ldots, \mathcal{T}_n$ *relative to* $\mathcal{T}_W$*. If to each tight, proper, mixed path* P *(with respect to the specified relative z-partition) in* $\mathcal{S}$*, there corresponds an explicit constraint of the form* $t_j - t_i \leq |P|$ *in* $\mathcal{C}$*, (where* t_i *and* t_j *are the first and last time-points in* P*, respectively), then running the General TDP algorithm on* $\mathcal{S}$ *(where mixed edges are determined according to the sets* $\mathcal{T}_1, \ldots, \mathcal{T}_n$*, which do* not *form a z-partition of* $\mathcal{T}$*) will produce a* relative *temporal decoupling of* $\mathcal{S}$ *with respect to the relative z-partition* $(\mathcal{T}_1, \ldots, \mathcal{T}_n; \mathcal{T}_W)$*.*

Proof Step 1 of the General TDP algorithm (Figure 5.29) initializes $\mathcal{E}$ to the set of tight, proper, mixed edges in $\mathcal{S}$ whose ZPS values are positive. Crucially, the determination of what constitutes a mixed edge is made, in this theorem, with respect to the sets, $\mathcal{T}_1, \ldots, \mathcal{T}_n$, which do *not* form a z-partition of $\mathcal{T}$.[6] The algorithm then processes the edges in $\mathcal{E}$ until

[6]Because the set $\mathcal{E}$ is "improperly initialized", this use of the General TDP algorithm is non-standard and will typically *not* result in a solution to any General TDP—which, for one thing, requires having a specified z-partition. To generate a temporal decoupling with respect to the z-partition of $\mathcal{T}$ formed by the sets $\mathcal{T}_1, \ldots, \mathcal{T}_n$ and $(\mathcal{T}_W \cup \{z\})$, would require processing not only the tight, proper, mixed edges described in the theorem, but also any such edges having endpoints in $\mathcal{T}_W$, which are ignored in the theorem.

each is dominated by a path through zero. Using the Greedy or Less-Greedy Strategy (cf. Definitions 5.24 and 5.27), the algorithm is guaranteed to terminate at Step 6.

Suppose P was some tight, proper, mixed path (with respect to the specified relative z-partition) from t_i to t_j in $\mathcal{S}$ prior to running the General TDP algorithm as described in the theorem. By assumption, there is an edge E of the form $t_j - t_i \leq |P|$ in $\mathcal{S}$. Since P is tight, so is the edge E. Furthermore, since P is a proper, mixed path with respect to the specified relative z-partition, E is a proper, mixed edge with respect to the sets $\mathcal{T}_1, \ldots, \mathcal{T}_n$. By the time the algorithm terminates, the edge E is guaranteed to be dominated by a path through zero. Thus, so is the path P. Since P was an arbitrary tight, proper, mixed path, the premise of Lemma 5.74 holds. Thus, the conclusion of the lemma also holds:

$$\mathcal{D}(t_p, z) + \mathcal{D}(z, t_q) \ \leq \ \mathcal{D}(t_p, t_q),$$

for every mixed *pair* (t_p, t_q) with respect to the specified relative z-partition. Since the constraint sets $\mathcal{C}_1, \ldots, \mathcal{C}_n$ returned by the General TDP algorithm (cf. Figure 5.29) are such that the subnetworks $\mathcal{S}_1, \ldots, \mathcal{S}_n$ are *d-subnetworks* of $\mathcal{S}$, Properties 1 and 2 of Theorem 5.69 follow (using an argument similar to that used in the proof of Theorem 5.23). Thus, by Theorem 5.70, the subnetworks $\mathcal{S}_1, \ldots, \mathcal{S}_n$ returned by the algorithm are a temporal decoupling of $\mathcal{S}$ relative to $\mathcal{T}_W$. ■

Theorem 5.75 reduces the problem of finding an algorithm for the Relative TDP to the problem of finding a set of *redundant* constraints to add to the STN to trick the General TDP algorithm into doing the right thing. The pseudo-code for the resulting Relative TDP algorithm is given in Figure 5.34. The algorithm uses a helper STN $\mathcal{S}_{Wx} = (\mathcal{T}, \mathcal{C}_{Wx})$, where

$$\mathcal{C}_{Wx} \ = \ \{(t_j - t_i \leq \delta) \in \mathcal{C} : t_j \neq z \text{ and } t_i \in \mathcal{T}_W\}.$$

Once constructed, the helper STN is fixed. The paths in the distance graph of the helper STN have the form: $(w_1, w_2, \ldots, w_k, t_j)$ where $k \geq 1$, each $w_i \in \mathcal{T}_W$, and $z \neq t_j \in \mathcal{T}$. For any $w \in \mathcal{T}_W$, and any $t_i \in \mathcal{T}$, where $t_i \neq z$, the length of the shortest mixed path (cf. Definition 5.71) from t_i to t_j having the form $(t_i, w, \ldots, t_j)$, if such exists, is given by: $\mathcal{D}(t_i, w) + \mathcal{D}_{Wx}(w, t_j)$. Except for the possibility of a tight *edge* from t_i to t_j, any tight, proper, mixed path from t_i to t_j *must* have the form $(t_i, w, \ldots, t_j)$, for some w. Thus, the algorithm checks for tight edges and tight paths of the specified form.[7] Anytime it finds such an edge or path, it adds a triple of the form $\langle t_i, t_j, \mathcal{D}(t_i, t_j)\rangle$ to the set $\mathcal{E}$. This information is sufficient to generate the desired constraints needed to trick the General TDP algorithm into processing the tight, proper, mixed paths required to generate a solution to the Relative TDP.

5.5.5 Lambda Bounds

Given a relative temporal decoupling, the agents $G_1, \ldots, G_n$ controlling the decoupled subnetworks $\mathcal{S}_1, \ldots, \mathcal{S}_n$ may operate independently. In other words, as long as each agent

[7] It is only necessary to check $w \in \mathcal{T}_W$ for which there is a tight edge from t_i to w. It is trivial to associate with each time-point the set of tight edges (or "successor edges") emanating from that time-point, referred to as $\mathit{TightSuccs}(t_i)$ in the pseudo-code for the algorithm.

Given: A consistent STN $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ whose time-points $\mathcal{T}$ are z-partitioned by $\mathcal{T}_1, \ldots, \mathcal{T}_n$ relative to $\mathcal{T}_W$.

(0) Set $\mathcal{E} = \emptyset$.

(1) Let $\mathcal{C}_{Wx} = \{(t_j - t_i \leq \delta) \in \mathcal{C} : t_j \neq z \text{ and } t_i \in \mathcal{T}_W\}$.

(2) Let $\mathcal{D}_{Wx}$ be the distance matrix for $(\mathcal{T}, \mathcal{C}_{Wx})$.

(3) `FOR` each proper, mixed pair (t_p, t_q) (w.r.t. the relative z-partition specified above):

If there is a tight edge E from t_p to t_q in $\mathcal{C}$, such that ZPS$(E) > 0$, then add $\langle t_p, t_q, \mathcal{D}(t_p, t_q)\rangle$ to $\mathcal{E}$;

Otherwise, `FOR` each $w \in (\mathcal{T}_W \cap \mathit{TightSuccs}(t_p))$:

If: $\mathcal{D}(t_p, w) + \mathcal{D}_{Wx}(w, t_q) = \mathcal{D}(t_p, t_q)$

and: $[\mathcal{D}(t_p, z) + \mathcal{D}(z, t_q)] > \mathcal{D}(t_p, t_q)$

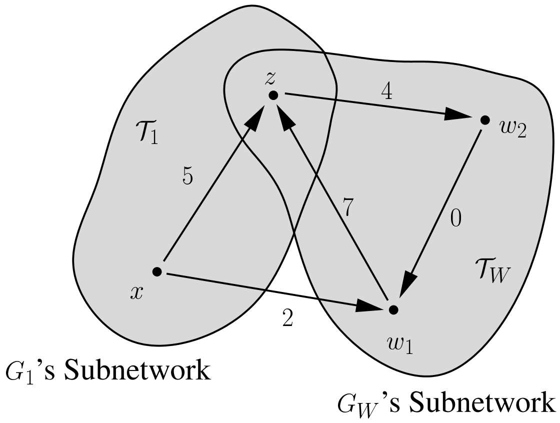
then: add $\langle t_p, t_q, \mathcal{D}(t_p, t_q)\rangle$ to $\mathcal{E}$

and break out of the inner `FOR` loop.

(4) Run Steps 2 through 6 of the General TDP Algorithm (cf. Figure 5.29).

Return: $\mathcal{C}_1 = \mathcal{C}_d|_{\mathcal{T}_1}, \ \ldots, \ \mathcal{C}_n = \mathcal{C}_d|_{\mathcal{T}_n}$ (generated by the General TDP Algorithm).

Figure 5.34: An algorithm for the Relative TDP based on the General TDP algorithm



Ordinary STN bound on edge from w_1 to z: -4

Lambda bound on edge from w_1 to z: 3

Figure 5.35: An illustration of the use of lambda bounds

G_i maintains the consistency of its own subnetwork $\mathcal{S}_i$, then each G_i is free to further constrain $\mathcal{S}_i$ in any way whatsoever without having to worry that it might threaten the consistency of the global network $\mathcal{S}$.

Now consider the agent G_W controlling the leftover time-points in $\mathcal{T}_W$. Since the time-points in $\mathcal{T}_W$ are *not* decoupled from $\mathcal{S}_1, \ldots, \mathcal{S}_n$, the agent G_W may *not* operate independently of the subnetworks, $\mathcal{S}_1, \ldots, \mathcal{S}_n$. In particular, G_W must refrain from adding any constraints that would impose any amount of additional constrainedness on $\mathcal{S}_1, \ldots, \mathcal{S}_n$. However, this does *not* imply that G_W must refrain altogether from adding constraints to the network.

Suppose G_W wants to add a constraint of the form, $t_j - t_i \leq \delta$, to the network. If G_W were in control of the entire network, then according to Theorem 4.33, G_W could add such a constraint if and only if $\delta \geq -\mathcal{D}(t_j, t_i)$. In other words, the negative-transpose entry in the distance matrix $\mathcal{D}$ for the global network would provide the lower bound for the length of the edge that G_W wants to add to the network (where shorter edges correspond to stronger constraints).

In the context of a relative temporal decoupling in which G_W controls the leftover time-points in $\mathcal{T}_W$, G_W must still observe the lower bound specified by the corresponding negative-transpose distance-matrix entry. However, because G_W must refrain from imposing any amount of additional constrainedness on the subnetworks $\mathcal{S}_1, \ldots, \mathcal{S}_n$, G_W must also observe an extra set of lower bounds which, in this section, are called *lambda bounds*.

Figure 5.35 illustrates the use of lambda bounds by the agent G_W. G_W is considering strengthening the edge from w_1 to z. If G_W controlled the entire network, then G_W could strengthen this edge by adding the constraint, $z - w_1 \leq -4$ (since $\mathcal{D}(z, w_1) = 4$). In other words, as long as G_W refrains from introducing any loops with negative path-length, all is well. However, in the context of a relative temporal decoupling in which

G_W must refrain from imposing any additional constrainedness on the subnetwork $\mathcal{S}_1$, G_W cannot add the constraint $z - w_1 \leq -4$ since it would contribute to a shorter path from x to z. In particular, it would introduce a path from x to w_1 to z of length -2, thus imposing additional constrainedness on the subnetwork $\mathcal{S}_1$. This does not mean that G_W must refrain altogether from strengthening the edge from w_1 to z, only that G_W must ensure that strengthening that edge does not impose on $\mathcal{S}_1$. In this case, G_W may add the constraint $z - w_1 \leq 3$ without imposing on the subnetwork $\mathcal{S}_1$.

This section presents theorems specifying necessary and sufficient lower bounds, called *lambda bounds*, that the agent G_W must observe to avoid imposing any amount of additional constrainedness on the subnetworks $\mathcal{S}_1, \ldots, \mathcal{S}_n$.

Unlike the agents $G_1, \ldots, G_n$, that are only allowed to add constraints among time-points within their own subnetworks, G_W is allowed not only to add constraints among time-points in $\mathcal{T}_W$, but also to add constraints that connect a time-point in $\mathcal{T}_W$ to a time-point in one of the subnetworks. Doing so will not impose any amount of additional constrainedness on the subnetworks $\mathcal{S}_1, \ldots, \mathcal{S}_n$ as long as the lambda bounds are obeyed.

It is important to keep in mind that the lambda bounds do not represent constraints in the network; instead, they represent restrictions on which constraints G_W may subsequently *add* to the network. In addition, whenever any agent G_i adds a constraint to its subnetwork $\mathcal{S}_i$, the corresponding lambda bounds for G_W are thereby *loosened*. In effect, when G_i adds a constraint to its subnetwork, it is restricting its own independence, thus making it easier for G_W to accommodate G_i's independence.

As will be discussed in Sections 5.5.6 and 5.5.7, the lambda bounds have natural interpretations when the Relative TDP is applied to the Post-Auction-Coordination and Temporal-Constraint-Generation problems that were outlined in Chapter 3. For example, when an agent has an outstanding bid, the lambda bounds specify which constraints the agent can add to its private schedule without threatening its ability to honor its bid should it ultimately be awarded. In that case, the bidder is represented by G_W, the auctioneer by G_1.

Finding the Lambda Bounds

If an agent controls all the time-points in an STN $\mathcal{S}$ and is concerned solely with maintaining the consistency of $\mathcal{S}$, then that agent may add any constraint of the form, $(t_j - t_i \leq \delta)$, where $\delta \geq -\mathcal{D}(t_j, t_i)$ (cf. Theorem 4.33). In other words, when contemplating strengthening an edge from t_i to t_j, the negative-transpose distance-matrix entry, $-\mathcal{D}(t_j, t_i)$, provides a necessary, sufficient, and readily available lower bound for δ.

In the context of a relative temporal decoupling, the agent G_W cannot rely solely on the negative-transpose distance-matrix entries to determine which constraints it may safely add to the network. It is not sufficient that the STN remain consistent; G_W must not add any constraints that would, directly or indirectly, add any additional amount of constrainedness to the subnetworks $\mathcal{S}_1, \ldots, \mathcal{S}_n$.

Theorem 5.76 shows that a characteristic set of inequalities hold in a consistent STN relative to the set $\mathcal{T}_W$, whether or not any temporal decoupling exists. These inequalities then motivate the definition of the lambda bounds (cf. Definition 5.77), which are computable in polynomial time. Theorems 5.79 and 5.80 verify that the lambda bounds are necessary and sufficient to ensure that G_W does not impose any additional constrainedness on $\mathcal{S}_1, \ldots, \mathcal{S}_n$. Theorem 5.80 shows that if G_W observes its lambda bounds, then the relative mergeable constraints property characterizing the relative temporal decoupling will necessarily be maintained.

Theorem 5.76 *Let $\mathcal{S} = (\mathcal{T}, \mathcal{C})$ be a consistent STN. Let $(\mathcal{T}'; \mathcal{T}_W)$ be a relative z-partition of $\mathcal{T}$ (i.e., $\mathcal{T} = \mathcal{T}' \cup \mathcal{T}_W$, $z \in \mathcal{T}'$, and $\mathcal{T}' \cap \mathcal{T}_W = \emptyset$). Let $\mathcal{C}_{xW}$ and $\mathcal{C}_{Wx}$ be given by:*

$$\begin{array}{rcl} \mathcal{C}_{xW} & = & \{(t_j - t_i \leq \delta) \in \mathcal{C} : t_j \in \mathcal{T}_W\} \\ \mathcal{C}_{Wx} & = & \{(t_j - t_i \leq \delta) \in \mathcal{C} : t_i \in \mathcal{T}_W\}. \end{array}$$

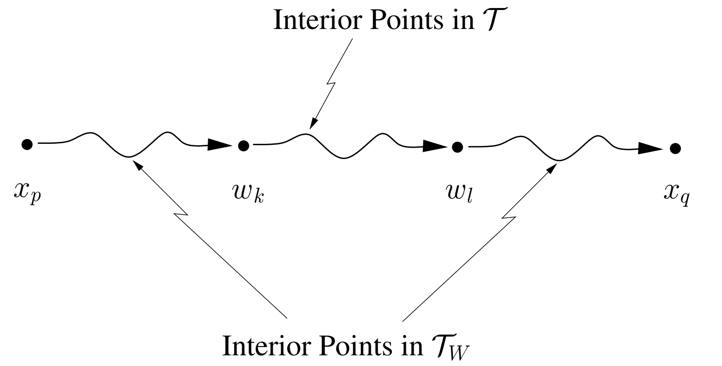
Let $\mathcal{D}_{xW}$ and $\mathcal{D}_{Wx}$ be the distance matrices for the respective STNs, $\mathcal{S}_{xW} = (\mathcal{T}, \mathcal{C}_{xW})$ and $\mathcal{S}_{Wx} = (\mathcal{T}, \mathcal{C}_{Wx})$. Then the following inequalities hold for all $w_k, w_l \in \mathcal{T}_W$ and $x_p, x_q \in \mathcal{T}'$:

$$\begin{array}{lrcl} (1) & \mathcal{D}(w_k, w_l) & \geq & [\mathcal{D}(x_p, x_q) - \mathcal{D}_{xW}(x_p, w_k) - \mathcal{D}_{Wx}(w_l, x_q)] \\ (2) & \mathcal{D}(x_p, w_l) & \geq & [\mathcal{D}(x_p, x_q) - \mathcal{D}_{Wx}(w_l, x_q)] \\ (3) & \mathcal{D}(w_k, x_q) & \geq & [\mathcal{D}(x_p, x_q) - \mathcal{D}_{xW}(x_p, w_k)] \end{array}$$

Proof First, notice that the paths in $\mathcal{S}_{xW}$ are precisely those paths in $\mathcal{S}$ having the form $(x, w_1, w_2, \ldots, w_k)$ or $(w_1, w_2, \ldots, w_k)$, where $x \in \mathcal{T}$ and $w_1, w_2, \ldots, w_k \in \mathcal{T}_W$. Similarly, the paths in $\mathcal{S}_{Wx}$ are precisely those paths in $\mathcal{S}$ having the form $(w_1, w_2, \ldots, w_k, x)$ or $(w_1, w_2, \ldots, w_k)$.

Let $w_k, w_l \in \mathcal{T}_W$ and $x_p, x_q \in \mathcal{T}'$ be arbitrary.

(1) Let $\mathcal{P}_1$ be the set of paths in $\mathcal{S}$ having the form:



where the interior points of the subpaths $x_p \rightsquigarrow w_k$ and $w_l \rightsquigarrow x_q$ are drawn exclusively from $\mathcal{T}_W$. If there are no such paths, then at least one of $\mathcal{D}_{xW}(x_p, w_k), \mathcal{D}(w_k, w_l)$ or $\mathcal{D}_{Wx}(w_l, x_q)$ is equal to positive infinity, in which case Inequality (1) necessarily holds. Otherwise, let P_1 be a path in $\mathcal{P}_1$ with minimal length, in which case:

$$|P_1| = \mathcal{D}_{xW}(x_p, w_k) + \mathcal{D}(w_k, w_l) + \mathcal{D}_{Wx}(w_l, x_q).$$

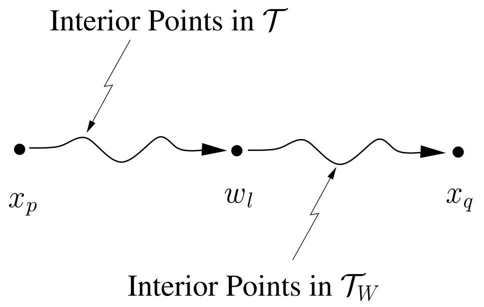
Since P_1 is a path from x_p to x_q, it must be that $\mathcal{D}(x_p, x_q) \leq |P_1|$. Thus:

$$\mathcal{D}(x_p, x_q) \;\leq\; \mathcal{D}_{xW}(x_p, w_k) + \mathcal{D}(w_k, w_l) + \mathcal{D}_{Wx}(w_l, x_q).$$

Hence:

$$\mathcal{D}(w_k, w_l) \;\geq\; \mathcal{D}(x_p, x_q) - \mathcal{D}_{xW}(x_p, w_k) - \mathcal{D}_{Wx}(w_l, x_q).$$

(2) Let $\mathcal{P}_2$ be the set of paths in $\mathcal{S}$ having the form:



If there are no such paths, then $\mathcal{D}(x_p, w_l)$ or $\mathcal{D}_{Wx}(w_l, x_q)$ is equal to positive infinity, in which case Inequality (2) necessarily holds. Otherwise, let P_2 be a path in $\mathcal{P}_2$ with minimal length, in which case:

$$|P_2| \;=\; \mathcal{D}(x_p, w_l) + \mathcal{D}_{Wx}(w_l, x_q).$$

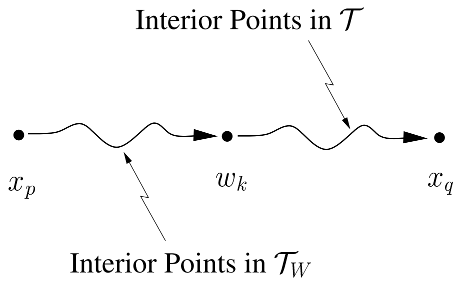
Since P_2 is a path from x_p to x_q, it must be that $\mathcal{D}(x_p, x_q) \leq |P_2|$. Thus:

$$\mathcal{D}(x_p, x_q) \;\leq\; \mathcal{D}(x_p, w_l) + \mathcal{D}_{Wx}(w_l, x_q).$$

Hence:

$$\mathcal{D}(x_p, w_l) \;\geq\; \mathcal{D}(x_p, x_q) - \mathcal{D}_{Wx}(w_l, x_q).$$

(3) Let $\mathcal{P}_3$ be the set of paths in $\mathcal{S}$ having the form:



If there are no such paths, then $\mathcal{D}_{xW}(x_p, w_k)$ or $\mathcal{D}(w_k, x_q)$ is equal to positive infinity, in which case Inequality (3) necessarily holds. Otherwise, let P_3 be a path in $\mathcal{P}_3$ with minimal length, in which case:

$$|P_3| \;=\; \mathcal{D}_{xW}(x_p, w_k) + \mathcal{D}(w_k, x_q).$$

Since P_3 is a path from x_p to x_q, it must be that $\mathcal{D}(x_p, x_q) \leq |P_3|$. Thus:

$$\mathcal{D}(x_p, x_q) \ \leq \ \mathcal{D}_{xW}(x_p, w_k) + \mathcal{D}(w_k, x_q).$$

Hence:

$$\mathcal{D}(w_k, x_q) \ \geq \ \mathcal{D}(x_p, x_q) - \mathcal{D}_{xW}(x_p, w_k). \ \blacksquare$$

Definition 5.77 (Lambda Bounds) *Given the same setup as in Theorem 5.76, let λ be defined as follows, where $w_k, w_l \in \mathcal{T}_W$ and $x_p, x_q \in \mathcal{T}'$ are arbitrary:*

$$\begin{array}{llcll}
(1) & \lambda(w_k, w_l) & = & \max\limits_{x_p, x_q \in \mathcal{T}'} & [\mathcal{D}(x_p, x_q) - \mathcal{D}_{xW}(x_p, w_k) - \mathcal{D}_{Wx}(w_l, x_q)] \\
(2) & \lambda(x_p, w_l) & = & \max\limits_{x_q \in \mathcal{T}'} & [\mathcal{D}(x_p, x_q) - \mathcal{D}_{Wx}(w_l, x_q)] \\
(3) & \lambda(w_k, x_q) & = & \max\limits_{x_p \in \mathcal{T}'} & [\mathcal{D}(x_p, x_q) - \mathcal{D}_{xW}(x_p, w_k)].
\end{array}$$

A constraint E of the form, $t_j - t_i \leq \delta$, where $t_i \in \mathcal{T}_W$ or $t_j \in \mathcal{T}_W$, is said to satisfy the appropriate lambda bound for $\mathcal{S}$ relative to $\mathcal{T}_W$ if $\delta \geq \lambda(t_i, t_j)$.

Proposition 5.78 *Given the distance matrices $\mathcal{D}, \mathcal{D}_{xW}$ and $\mathcal{D}_{Wx}$, and assuming constant matrix lookup time, each $\lambda(w_k, w_l)$ may be computed in $O(|\mathcal{T}'|^2)$ time, while each $\lambda(x_p, w_l)$ and each $\lambda(w_k, x_q)$ may be computed in $O(|\mathcal{T}'|)$ time. Since there are $O(|\mathcal{T}_W|^2)$ different values to compute for the first type and $O(|\mathcal{T}'||\mathcal{T}_W|)$ different values to compute for each of the other types, the entire set of lambda bounds may be computed in $O(|\mathcal{T}_W|^2|\mathcal{T}'|^2)$ time.*

Theorem 5.79 *Given the same setup as in Theorem 5.76, let E be a constraint of the form, $t_j - t_i \leq \delta$, where $t_i \in \mathcal{T}_W$ or $t_j \in \mathcal{T}_W$. Let $\mathcal{C}^+ = \mathcal{C} \cup \{E\}$ and $\mathcal{S}^+ = (\mathcal{T}, \mathcal{C}^+)$. (In other words, $\mathcal{S}^+$ is the STN resulting from adding E to $\mathcal{S}$.) Under the assumption that E is consistent with $\mathcal{S}$ (i.e., that $\mathcal{S}^+$ is consistent), the following are equivalent:*

- *E satisfies the appropriate lambda bound for $\mathcal{S}$ relative to $\mathcal{T}_W$.*
- *$\mathcal{D}^+(x_p, x_q) = \mathcal{D}(x_p, x_q)$, for all $x_p, x_q \in \mathcal{T}'$.*

Proof Suppose E has the form $w_l - w_k \leq \delta$. (The other forms of E are handled similarly.) Thus, for E, the appropriate lambda bound is:

$$\lambda(w_k, w_l) \ = \ \max_{x_p, x_q \in \mathcal{T}'} [\mathcal{D}(x_p, x_q) - \mathcal{D}_{xW}(x_p, w_k) - \mathcal{D}_{Wx}(w_l, x_q)].$$

($\Leftarrow$) Suppose E fails to satisfy the above lambda bound. Then, for some $x_p, x_q \in \mathcal{T}'$,

$$\delta \ < \ \mathcal{D}(x_p, x_q) - \mathcal{D}_{xW}(x_p, w_k) - \mathcal{D}_{Wx}(w_l, x_q),$$

which implies that

$$\mathcal{D}_{xW}(x_p, w_k) + \delta + \mathcal{D}_{Wx}(w_l, x_q) \ < \ \mathcal{D}(x_p, x_q).$$

Since the left-hand side of this inequality measures the length of some path in $\mathcal{S}^+$ from x_p to x_q (via w_k and w_l), it must be that:

$$\mathcal{D}^+(x_p, x_q) \ \leq\ \mathcal{D}_{xW}(x_p, w_k) + \delta + \mathcal{D}_{Wx}(w_l, x_q),$$

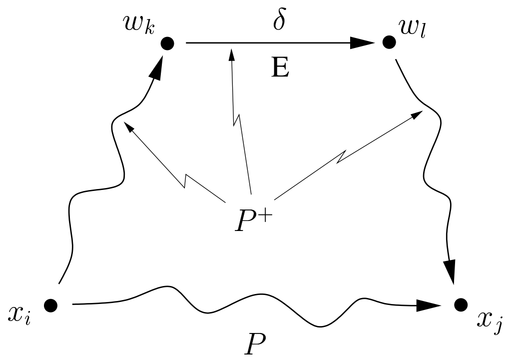
whence $\mathcal{D}^+(x_p, x_q) < \mathcal{D}(x_p, x_q)$. Thus, $\mathcal{D}^+(x_p, x_q) \neq \mathcal{D}(x_p, x_q)$.

($\Rightarrow$) Suppose $\mathcal{D}^+(x_i, x_j) \neq \mathcal{D}(x_i, x_j)$ for some $x_i, x_j \in \mathcal{T}'$. Since each constraint in $\mathcal{S}$ is present in $\mathcal{S}^+$, $\mathcal{D}^+(x_i, x_j) \leq \mathcal{D}(x_i, x_j)$. Thus, it must be that $\mathcal{D}^+(x_i, x_j) < \mathcal{D}(x_i, x_j)$.

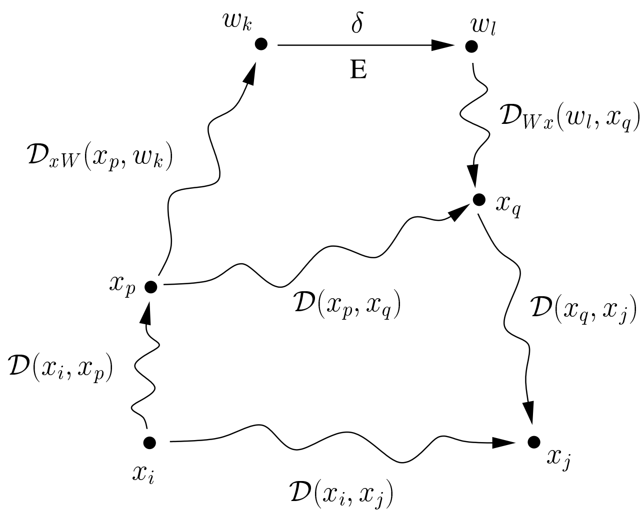
Let P^+ be a shortest path from x_i to x_j in $\mathcal{S}^+$, and let P be a shortest path from x_i to x_j in $\mathcal{S}$. Thus,

$$|P^+| \ =\ \mathcal{D}^+(x_i, x_j) \ <\ \mathcal{D}(x_i, x_j) \ <\ |P|.$$

Since $\mathcal{S}$ and $\mathcal{S}^+$ differ only in that $\mathcal{S}^+$ contains E, it must be that E is on the path P^+. Since removing loops from a shortest path does not change its length, we can assume that E occurs only once in P^+. Thus, the situation is as shown below.



Now, let x_p be the latest point in P^+ such that x_p is before w_k, but $x_p \notin \mathcal{T}_W$. Similarly, let x_q be the earliest point in P^+ such that x_q is after w_l, but $x_q \notin \mathcal{T}_W$. Note that it might be that x_i and x_p are the same time-point variable; ditto for x_j and x_q. The picture below illustrates the situation,



where the interior points of the subpaths from x_p to w_k, and from w_l to x_q, lie within $\mathcal{T}_W$ (by choice of x_p and x_q); and where the shortest path in $\mathcal{S}$ from x_p to x_q is shown for later reference.

Now, since $|P^+| < \mathcal{D}(x_i, x_j)$, we have that:

$$\mathcal{D}(x_i, x_p) + \mathcal{D}_{xW}(x_p, w_k) + \delta + \mathcal{D}_{Wx}(w_l, x_q) + \mathcal{D}(x_q, x_j) \ = \ |P^+| \ < \ \mathcal{D}(x_i, x_j).$$

But we also have, by two applications of the Triangle Inequality to $\mathcal{D}(x_i, x_j)$, that:

$$\mathcal{D}(x_i, x_j) \ \leq \ \mathcal{D}(x_i, x_p) + \mathcal{D}(x_p, x_q) + \mathcal{D}(x_q, x_j).$$

Putting the above inequalities together and canceling like terms yields the following:

$$\mathcal{D}_{xW}(x_p, w_k) + \delta + \mathcal{D}_{Wx}(w_l, x_q) \ < \ \mathcal{D}(x_p, x_q),$$

whence:

$$\delta \ < \ \mathcal{D}(x_p, x_q) - \mathcal{D}_{xW}(x_p, w_k) - \mathcal{D}_{Wx}(w_l, x_q),$$

which implies that E does not satisfy the appropriate lambda bound. ■

Theorem 5.80 shows that if a constraint E is consistent with $\mathcal{S}$ and satisfies the appropriate lambda bound, then adding E to $\mathcal{S}$ will not threaten the mergeable constraints property that characterizes a relative temporal decoupling.

Theorem 5.80 *Let $\mathcal{S} = (\mathcal{T}, \mathcal{C})$. Let $\mathcal{S}_1 = (\mathcal{T}_1, \mathcal{C}_1)$, ..., $\mathcal{S}_n = (\mathcal{T}_n, \mathcal{C}_n)$ be a temporal decoupling of $\mathcal{S}$ relative to $\mathcal{T}_W$. Let E be a constraint of the form, $t_j - t_i \leq \delta$, where $t_i \in \mathcal{T}_W$ or $t_j \in \mathcal{T}_W$. Let $\mathcal{C}^+ = \mathcal{C} \cup \{E\}$ and $\mathcal{S}^+ = (\mathcal{T}, \mathcal{C}^+)$. (In other words, $\mathcal{S}^+$ is the STN resulting from adding E to $\mathcal{S}$.) If E is consistent with $\mathcal{S}$ and satisfies the appropriate λ bound, then the STNs $\mathcal{S}_1, \ldots, \mathcal{S}_n$ temporally decouple $\mathcal{S}^+$ relative to $\mathcal{T}_W$. Furthermore, if Property 1 of Theorem 5.69 holds with equality in all instances for the temporal decoupling of $\mathcal{S}$, then it also does so for the temporal decoupling of $\mathcal{S}^+$.*

Proof To show that $\mathcal{S}_1, \ldots, \mathcal{S}_n$ temporally decouple $\mathcal{S}^+$ relative to $\mathcal{T}_W$, it suffices, by Theorem 5.70, to show that $\mathcal{S}^+$ is consistent and that Properties 1 and 2 from Theorem 5.69 hold. That $\mathcal{S}^+$ is consistent follows from E being consistent with $\mathcal{S}$.

To show that Properties 1 and 2 hold, let $\mathcal{T}' = \mathcal{T}_1 \cup \ldots \cup \mathcal{T}_n$. By Theorem 5.79, since E satisfies the appropriate lambda bound, $\mathcal{D}^+(x_p, x_q) = \mathcal{D}(x_p, x_q)$, for all $x_p, x_q \in \mathcal{T}'$. This in turn implies that Properties 1 and 2 hold with respect to $\mathcal{S}^+$ if and only if they hold with respect to $\mathcal{S}$. Furthermore, equality in Properties 1 and 2 holds in all instances for $\mathcal{S}^+$ if and only if it holds in all instances for $\mathcal{S}$. ■

5.5.6 The Temporal-Constraint-Generation Problem

This section addresses the Temporal-Constraint-Generation problem that an agent must solve when generating bids in the type of task-allocation auction described in Chapter 3. This section formally defines the TCG problem, presents necessary and sufficient characterizations of solutions to the TCG problem, and provides an algorithm that an agent can

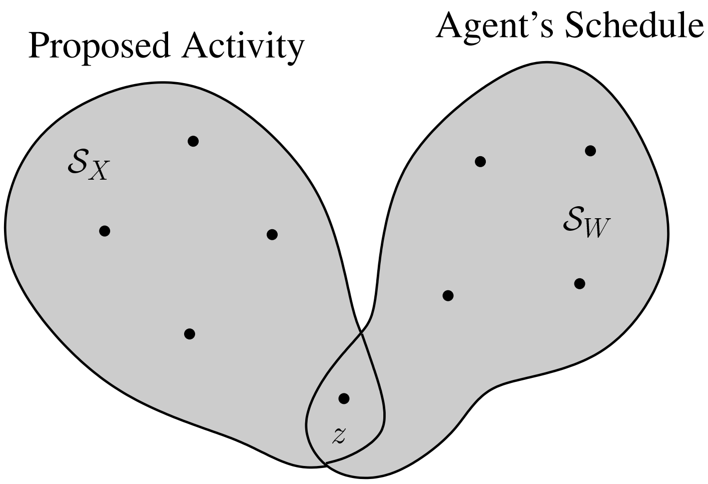


Figure 5.36: The pre-bidding situation for the TCG problem

use to solve instances of the TCG problem. The main result is proven by showing that the TCG problem is a special instance of the Relative TDP in which $n = 1$.

Let $\mathcal{S}_W = (\mathcal{T}_W \cup \{z\}, \mathcal{C}_W)$ be an STN representing the agent's private schedule of pre-existing commitments.[8] Let $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ be an STN representing the time-points and temporal constraints corresponding to a set of tasks in some proposed group activity. Since the activity is only in the proposal stage, it cannot share any time-points with the agent's pre-existing schedule of commitments. Thus, $\mathcal{T}_X \cap \mathcal{T}_W = \emptyset$. Notice, however, that $\mathcal{S}_X$ and $\mathcal{S}_W$ both include the zero time-point variable z. Figure 5.36 illustrates the pre-bidding situation.

Now suppose that the agent is considering submitting a bid on a set of tasks associated with the proposed group activity. Let $\mathcal{T}'_X \subseteq \mathcal{T}_X$ be the set of time-points associated with the tasks that the agent wants to bid on, as illustrated in Figure 5.37. To ensure that the tasks being bid on do not conflict with its schedule of pre-existing commitments, the agent must add some set $\mathcal{C}_{XW}$ of temporal constraints joining points in $\mathcal{T}_X$ and $\mathcal{T}_W$, as illustrated in Figure 5.38. Although in practice the edges in $\mathcal{C}_{XW}$ join points in $\mathcal{T}'_X$ and $\mathcal{T}_W$, the analysis below allows the constraints in $\mathcal{C}_{XW}$ to join points in $\mathcal{T}_X$ and $\mathcal{T}_W$. Let $\mathcal{S} = (\mathcal{T}, \mathcal{C}) = (\mathcal{T}_X \cup \mathcal{T}_W, \mathcal{C}_X \cup \mathcal{C}_W \cup \mathcal{C}_{XW})$ (i.e., the entire network represented in Figure 5.38).

Note. The process of generating the constraint set, $\mathcal{C}_{XW}$, is called the Task-Integration-Scheduling (TIS) problem (Hunsberger, 2002b). There are many possible algorithms for solving the TIS problem, but they are beyond the scope of this thesis. The TCG problem discussed in this section does not depend on the manner in which the constraint set $\mathcal{C}_{XW}$ is

[8]The set of time-points in $\mathcal{S}_W$ is written as $\mathcal{T}_W \cup \{z\}$ to facilitate the analogy with the corresponding instance of the Relative Temporal Decoupling Problem. The set $\mathcal{T}_W$ will be the set of leftover points for the relative temporal decoupling. The constraints in $\mathcal{C}_W$ may, of course, involve $\{z\}$.

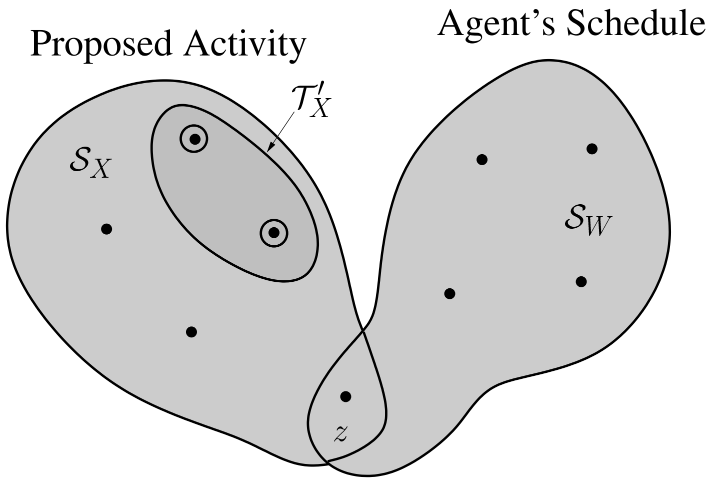


Figure 5.37: Highlighting the time-points associated with tasks the agent wants to bid on

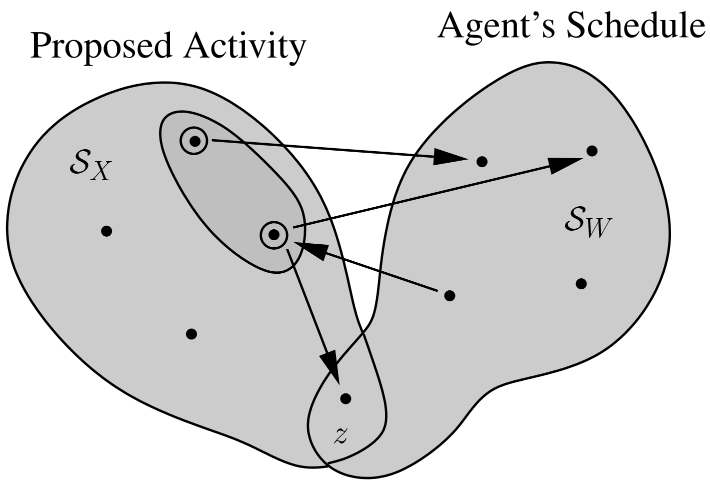


Figure 5.38: The set $\mathcal{C}_{XW}$ of temporal constraints for the TCG problem

generated.

As described in Chapter 3, the agent is allowed to include temporal constraints in its bid. However, the temporal constraints included with the bid can only refer to time-points associated with the proposed activity (i.e., time-points in the set $\mathcal{T}_X$). Thus, the agent is not allowed to simply make its bid conditioned on the entire contents of the STN $\mathcal{S}$.[9] The Temporal-Constraint-Generation problem (cf. Definition 5.81, below) is the problem of finding a set $\mathcal{C}_B$ of constraints to include with a bid such that no matter what additional constraints $\mathcal{C}''$ the auctioneer might add (in accordance with the rules of the auction), the agent's STN $\mathcal{S}$ will necessarily remain consistent.

The auctioneer is allowed to award any combination of bids that are mutually consistent and consistent with the constraints in the proposed activity. Thus, from the perspective of the bidding agent, the auctioneer is, in effect, allowed to add any set of constraints consistent with the constraints in $(\mathcal{C}_X \cup \mathcal{C}_B)$ (i.e., the constraints associated with the proposed activity and the constraints in the agent's bid).

Definition 5.81 (The Temporal-Constraint-Generation Problem) *Let $\mathcal{T}_X$ and $\mathcal{T}_W$ be sets of time-points such that $z \in \mathcal{T}_X$ and $\mathcal{T}_X \cap \mathcal{T}_W = \emptyset$. Let $\mathcal{C}_X$, $\mathcal{C}_W$ and $\mathcal{C}_{XW}$ be sets of constraints over the time-points in $\mathcal{T}_X$, $\{z\} \cup \mathcal{T}_W$, and $\mathcal{T}_X \cup \mathcal{T}_W$, respectively, such that $\mathcal{S} = (\mathcal{T}, \mathcal{C}) = (\mathcal{T}_X \cup \mathcal{T}_W, \mathcal{C}_X \cup \mathcal{C}_W \cup \mathcal{C}_{XW})$ is a consistent STN. Find a set of constraints $\mathcal{C}_B$ over $\mathcal{T}_X$ such that:*

(1) $(\mathcal{T}_X, \mathcal{C}_X \cup \mathcal{C}_B)$ is consistent; and

(2) for any set of constraints $\mathcal{C}''$ over $\mathcal{T}_X$, if the constraints in $\mathcal{C}''$ are consistent with $(\mathcal{T}_X, \mathcal{C}_X \cup \mathcal{C}_B)$, then they are also consistent with $\mathcal{S}$.

Notice that in Definition 5.81, $\mathcal{S}$ being consistent implies that both $\mathcal{S}_W = (\{z\} \cup \mathcal{T}_W, \mathcal{C}_W)$ and $\mathcal{S}_X = (\mathcal{T}_X, \mathcal{C}_X)$ are consistent. In addition, condition (1) is equivalent to saying that the constraints in $\mathcal{C}_B$ are consistent with $\mathcal{S}_X$.

Theorem 5.82 (Necessary and Sufficient Conditions for TCG Solution) *Given the same setup as in Definition 5.81, let $\mathcal{C}_B$ be a set of constraints over the time-points in $\mathcal{T}_X$ such that $\mathcal{S}_1 = (\mathcal{T}_X, \mathcal{C}_X \cup \mathcal{C}_B)$ is consistent. Then $\mathcal{C}_B$ is a solution to the TCG problem if and only if the constraints in $\mathcal{C}_X \cup \mathcal{C}_B$ entail the constraints in $\mathcal{C}_d|_{\mathcal{T}_X}$, where $\mathcal{C} = \mathcal{C}_X \cup \mathcal{C}_W \cup \mathcal{C}_{XW}$.*

Proof Let $\mathcal{D}_1$ be the distance matrix for $\mathcal{S}_1$.

($\Rightarrow$) Suppose $\mathcal{C}_B$ is a solution to the TCG problem. By Definition 5.61, the following imply that $\mathcal{S}_1$ is a temporal decoupling of $\mathcal{S} = (\mathcal{T}, \mathcal{C}) = (\mathcal{T}_X \cup \mathcal{T}_W, \mathcal{C}_X \cup \mathcal{C}_W \cup \mathcal{C}_{XW})$ relative to $\mathcal{T}_W$ in the case where $n = 1$.

- $\mathcal{S}_1$ and $\mathcal{S}$ are both consistent.

[9]This might be undesirable for other reasons as well. For example, the agent might not be *allowed* to reveal the contents of its private schedule. In addition, making the bid conditioned on its entire schedule could greatly restrict the agent's ability to take on additional commitments while the bid is outstanding.

- $(\mathcal{T}_X; \mathcal{T}_W)$ is a z-partition of $\mathcal{T} = (\mathcal{T}_X \cup \mathcal{T}_W)$, since $z \in \mathcal{T}_X$ and $\mathcal{T}_X \cap \mathcal{T}_W = \emptyset$.

- Since $\mathcal{C}_B$ is a solution to the TCG problem, Condition (2) of Definition 5.81 holds. Since Condition (2) is precisely the Relative Mergeable Constraints Property (cf. Definition 5.65 in the case, $n = 1$) which, by Theorem 5.66, is equivalent to the Relative Mergeable Solutions Property, the Relative MSP necessarily holds.

Since $\mathcal{S}_1$ temporally decouples $\mathcal{S}$ relative to $\mathcal{T}_W$, Property (1) of Theorem 5.69 holds:

$$\mathcal{D}_1(t_i, t_j) \;\leq\; \mathcal{D}(t_i, t_j), \text{ for all } t_i, t_j \in \mathcal{T}_X.$$

By Proposition 4.29, $\mathcal{D} = \mathcal{D}_d$. Furthermore, by Lemma 4.42, $(\mathcal{D}_d)|_{\mathcal{T}_X}$ is identical to the distance matrix, $\mathcal{D}_d|_{\mathcal{T}_X}$, for the d-subnetwork $\mathcal{S}_d|_{\mathcal{T}_X} = (\mathcal{T}_X, \mathcal{C}_d|_{\mathcal{T}_X})$. Thus,

$$\mathcal{D}_1(t_i, t_j) \;\leq\; \mathcal{D}_d|_{\mathcal{T}_X}(t_i, t_j), \text{ for all } t_i, t_j \in \mathcal{T}_X,$$

which, by Corollary 4.34, implies that the constraints in $\mathcal{C}_X \cup \mathcal{C}_B$ entail the constraints in $\mathcal{C}_d|_{\mathcal{T}_X}$.

($\Leftarrow$) Suppose the constraints in $\mathcal{C}_X \cup \mathcal{C}_B$ entail the constraints in $\mathcal{C}_d|_{\mathcal{T}_X}$. Reversing the last few steps of the above argument implies that Property 1 from Theorem 5.69 holds. Since Property 2 is vacuous in the case, $n = 1$, Theorem 5.70 gives us that $\mathcal{S}_1$ is a temporal decoupling of $\mathcal{S}$ relative to $\mathcal{T}_W$. Thus, the Mergeable Solutions Property holds relative to $\mathcal{T}_W$ (cf. Definition 5.61). Hence, the equivalent Mergeable Constraints Property holds relative to $\mathcal{T}_W$ which, as already argued, is equivalent to Condition (2) of Definition 5.81. Since $\mathcal{S}_1$ being consistent is Condition (1) of Definition 5.81, $\mathcal{C}_B$ is a solution to the TCG problem. ■

Corollary 5.83 *If $\mathcal{C}_B$ is a solution to the TCG problem and $\Delta\mathcal{C}_B$ is a set of constraints over the time-points in $\mathcal{T}_X$ such that $\Delta\mathcal{C}_B$ is consistent with $\mathcal{C}_X \cup \mathcal{C}_B$, then $\mathcal{C}_B \cup \Delta\mathcal{C}_B$ is also a solution to the TCG problem.*

Theorem 5.82 implies that if $\mathcal{C}_B$ is a set of constraints over $\mathcal{T}_X$ such that $(\mathcal{C}_X \cup \mathcal{C}_B)$ is equivalent to $\mathcal{C}_d|_{\mathcal{T}_X}$, then $\mathcal{C}_B$ is a solution to the TCG problem representing the weakest constraints that are sufficient to protect the bidder's private schedule of pre-existing commitments against the possibility of the bid subsequently being awarded. Stated differently, such a solution places the weakest possible constraints on the auctioneer while still protecting the bidder's private schedule of pre-existing commitments.

An Algorithm for Solving the TCG Problem

An algorithm for solving the TCG problem is given in Figure 5.39. The inputs to the algorithm are equivalent to the setup of Definition 5.81. In Step 1, the algorithm sets $\tilde{\mathcal{S}}$ to the d-subnetwork of $\mathcal{S}$ relative to $\mathcal{T}_X$. By Theorem 5.82, the constraints in $\mathcal{C}_d|_{\mathcal{T}_X}$ would provide a solution to the TCG problem, but one that contains $|\mathcal{T}_X|^2$ explicit constraints. The rest of the algorithm is devoted to generating an equivalent constraint set, $\mathcal{C}_B$, with fewer

Given:

- sets $\mathcal{T}_X$ and $\mathcal{T}_W$ of time-points such that $z \in \mathcal{T}_X$ and $\mathcal{T}_X \cap \mathcal{T}_W = \emptyset$; and
- sets $\mathcal{C}_X, \mathcal{C}_W$ and $\mathcal{C}_{XW}$ of constraints over the time-points in $\mathcal{T}_X$, $\{z\} \cup \mathcal{T}_W$, and $\mathcal{T}_X \cup \mathcal{T}_W$, such that $\mathcal{S} = (\mathcal{T}, \mathcal{C}) = (\mathcal{T}_X \cup \mathcal{T}_W, \mathcal{C}_X \cup \mathcal{C}_W \cup \mathcal{C}_{XW})$ is a consistent STN.

(1) Let $\tilde{\mathcal{S}} = (\mathcal{T}_X, \mathcal{C}_d|_{\mathcal{T}_X})$.

(2) Let $\tilde{\mathcal{S}}^{nrc} = (\tilde{\mathcal{T}}^{nrc}, \tilde{\mathcal{C}}^{nrc})$ and $\tilde{\mathcal{S}}_1^{rc} = (\tilde{\mathcal{T}}_1^{rc}, \tilde{\mathcal{C}}_1^{rc})$, $\ldots$, $\tilde{\mathcal{S}}_n^{rc} = (\tilde{\mathcal{T}}_n^{rc}, \tilde{\mathcal{C}}_n^{rc})$ be the subnetworks resulting from decoupling the rigid components from $\tilde{\mathcal{S}}$, as described in Section 4.1 (cf. Definition 4.50). (If $\tilde{\mathcal{S}}$ has no rigid components, then $\tilde{\mathcal{S}}^{nrc} = \tilde{\mathcal{S}}$ and $n = 0$.)

(3) Let $\mathcal{C}_B = \{ (t_j - t_i \leq \delta) \in ((\tilde{\mathcal{C}}^{nrc})_d^u \cup \tilde{\mathcal{C}}_1^{rc} \cup \ldots \cup \tilde{\mathcal{C}}_n^{rc}) : \delta < \mathcal{D}_X(t_i, t_j) \}$.

Return: $\mathcal{C}_B$.

Figure 5.39: An algorithm for solving the TCG problem

explicit constraints. Step 2 decouples any rigid components from the rest of $\tilde{\mathcal{S}}$, as described in Section 4.1. By Theorem 4.48, Lemma 4.49 and Theorem 4.53, the constraint set

$$\mathcal{C}^* = (\tilde{\mathcal{C}}^{nrc})_d^u \cup \tilde{\mathcal{C}}_1^{rc} \cup \ldots \cup \tilde{\mathcal{C}}_n^{rc}$$

is equivalent to the constraints in $\tilde{\mathcal{S}}$, and furthermore, has the fewest number of edges in any constraint set equivalent to $\tilde{\mathcal{S}}$. Step 3 of the algorithm simply removes any constraints from $\mathcal{C}^*$ that are not strictly stronger than the corresponding strongest implicit constraints in $\mathcal{C}_X$.

Adding Constraints in the Context of Outstanding Bids

In the case of an agent (G_W) having an outstanding bid in the sort of auction described in Chapter 3, the lambda bounds described in Section 5.5.5 specify which constraints the agent may add to its private schedule without threatening its ability to carry out the tasks in that bid should it ultimately be awarded. For example, suppose Bob has an outstanding bid to do a dish-washing activity between 3:00 and 6:00 p.m. Because the auctioneer (played by G_i) is, in effect, free to make Bob do the dishes any time between 3:00 and 6:00, Bob must refrain from committing to a house-painting activity scheduled to run from 4:30 to 5:30. (The lambda bounds preclude Bob from imposing any additional constrainedness on the dish-washing activity.) However, should the auctioneer subsequently inform Bob that the dish-washing activity will occur between 3:30 and 4:00, then Bob would thereby become free to commit to the house-painting activity.

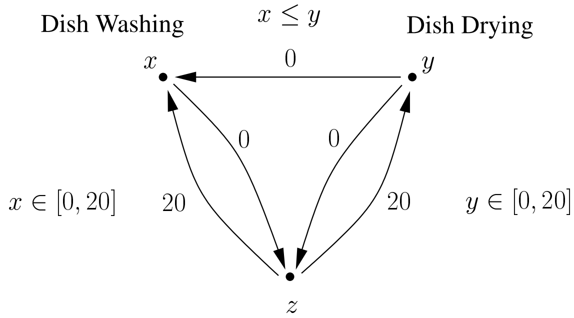


Figure 5.40: An STN representing dependent dish washing and drying activities

5.5.7 The Post-Auction Coordination Problem

One of the original motivations for analyzing the temporal decoupling problem was that agents working on a set of temporally dependent tasks must do something to ensure that the temporal constraints among tasks being done by different agents are satisfied. Three possible strategies they might follow are: (1) impose additional constraints on the network sufficient to decouple the tasks being done by different agents, thereby enabling agents to work independently, without the need for further coordination or communication; (2) decouple the tasks being done by *some* of the agents (so that they may work independently), while leaving the rest of the agents dependent on them; and (3) create a hierarchical decoupling in which each decoupled subnetwork is itself partitioned into decoupled subnetworks.

The first strategy can be realized by a direct application of the General TDP algorithm, as described in Section 5.4.5. By decoupling the tasks being done by all of the different agents, those agents are thereby free to operate independently, as long as each maintains the consistency of its local subnetwork. However, in some cases, fully decoupling the network may require the imposition of constraints beyond those that the agents are willing to accept.

For example, in the simple dish-washing/dish-drying scenario described in Section 5.4.5, and repeated in Figure 5.40, the tasks might be fully decoupled by adding such constraints as

$$x \le 12 \qquad \text{and} \qquad y \ge 12;$$

however, it might instead be preferable to give the dish-washing agent G_x complete freedom to select the time for its task, while making the dish-drying agent G_y dependent on the choice made by G_x. Doing this would correspond to a *relative* temporal decoupling where

- $n = 1$;
- the subnetwork controlled by G_x is: $\mathcal{S}_1 = (\{z, x\},\ \{(x - z \le 20), (z - x \le 0)\});$
- the set of leftover time-points controlled by G_y is given by: $\mathcal{T}_W = \{y\}$.

While G_y is waiting for G_x to make up its mind about when it will wash the dishes, G_y has less flexibility, and must use the lambda bounds described in Section 5.5.5 to decide which constraints it can safely add to the network. However, giving G_x complete freedom

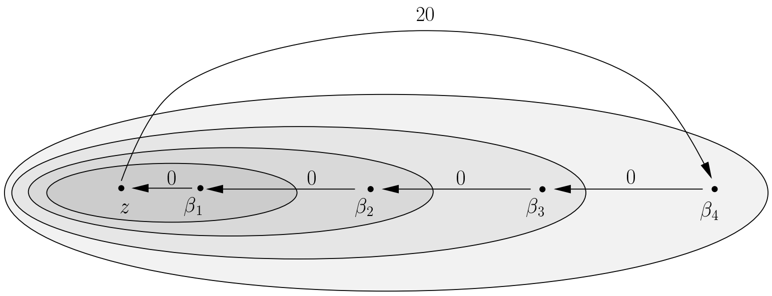


Figure 5.41: A hierarchical decoupling for a sequence of tasks

might end up giving G_y more freedom, too. For example, should G_x eventually decide to wash the dishes at time 4, then G_y could decide to dry the dishes immediately thereafter instead of having to wait until time 12. When G_x decides to set x to 4, it causes the corresponding lambda bound for G_y to *decrease*, as described in Section 5.5.5, resulting in greater flexibility for G_y. Furthermore, it may be that G_x simply *requires* complete flexibility in making its dish-washing decision; for instance, it may not know when it will have access to a sink.

In general, the Relative TDP algorithm described in Section 5.5.4 enables a more flexible approach to solving the Post-Auction Coordination problem. In this approach, some agents are given greater freedom while others are required to be dependent on them. However, as seen in the above example, when the agents controlling the decoupled subnetworks further constrain their subnetworks (e.g., by deciding when to execute tasks under their control), it may translate into greater freedom of choice for the dependent agents as well, as evidenced by decreasing lambda bounds.

The third strategy calls for a structured, hierarchical approach to the allocation of temporal flexibility and dependence among the agents. To illustrate this strategy, consider the case of agents, G_1, G_2, G_3 and G_4, assigned to do tasks, $\beta_1, \beta_2, \beta_3$ and β_4, respectively, where the tasks are constrained to be executed consecutively within the interval $[0, 20]$. For simplicity, we suppose that the tasks have zero duration. In this case, it may be preferable to allow G_1 to have complete freedom to execute its task β_1 at any time within $[0, 20]$. Although G_2 would be dependent on G_1, once G_1 made up its mind to execute β_1 at some specified point, say, at time 3, then G_2 could act independently. Similarly, G_3 would be dependent on G_2 until G_2 made a decision to execute β_2, say, at time 7. Finally, G_4 would be dependent on G_3 until G_3 made a decision to execute β_3, say, at time 14—at which point, G_4 could act independently.

The simple hierarchy for the above example is illustrated in Figure 5.41. It can be generated by recursive applications of the Relative TDP algorithm, as follows. First, the subnetwork $\mathcal{S}'$ comprising the time-points $\{z, \beta_1, \beta_2, \beta_3\}$ is decoupled (in the case $n = 1$) relative to the set of leftover time-points $\{\beta_4\}$. Next, within the independent subnetwork $\mathcal{S}'$, the subnetwork $\mathcal{S}''$ comprising the time-points $\{z, \beta_1, \beta_2\}$ is decoupled (in the case $n = 1$)

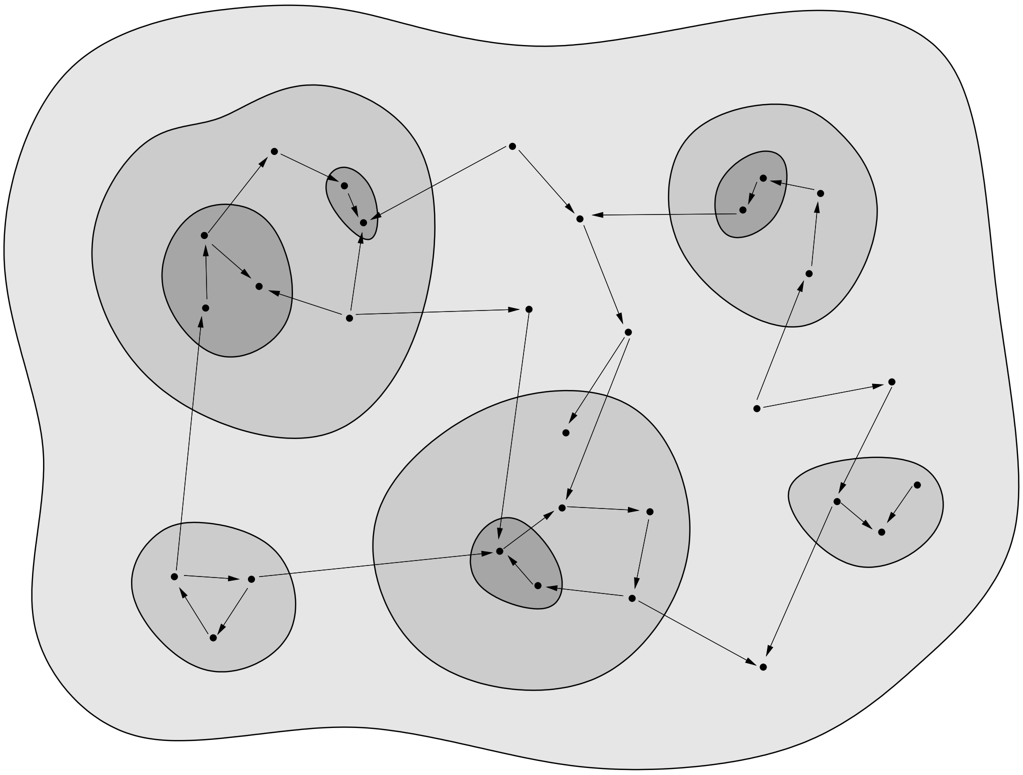


Figure 5.42: Sample Hierarchy of Partially Decoupled Subnetworks

relative to the set of leftover time-points $\{\beta_3\}$. Finally, within the independent subnetwork $\mathcal{S}''$, the subnetwork $\mathcal{S}'''$ comprising the time-points $\{z, \beta_1\}$ is decoupled (in the case $n = 1$) relative to the set of leftover time-points $\{\beta_2\}$.

Of course, relative temporal decouplings can be used to generate hierarchies with more complex structure, as illustrated in Figure 5.42. In general, given an STN $(\mathcal{T}, \mathcal{C})$, the procedure for generating a hierarchical decoupling is as follows.

- First, choose some relative z-partition $(\mathcal{T}_1, \ldots, \mathcal{T}_n; \mathcal{T}_W)$ of $\mathcal{T}$ and then use the Relative TDP algorithm to generate STNs $(\mathcal{T}_1, \mathcal{C}_1), \ldots, (\mathcal{T}_n, \mathcal{C}_n)$ that temporally decouple $(\mathcal{T}, \mathcal{C})$ relative to $\mathcal{T}_W$.

- Next, for each decoupled subnetwork $(\mathcal{T}_i, \mathcal{C}_i)$, choose some relative z-partition $(\mathcal{T}_{i1}, \ldots, \mathcal{T}_{in_i}; \mathcal{T}_{W_i})$ and then use the Relative TDP algorithm to generate STNs that temporally decouple $(\mathcal{T}_i, \mathcal{C}_i)$ relative to $\mathcal{T}_{W_i}$.

- Continue recursively down the hierarchy.

5.6 Related Work

5.6.1 Separation Vertices

The temporal decoupling problems presented in this chapter are related to *separation vertices*, as defined by Dechter et al. (1991).

Definition 5.84 *(Dechter, Meiri, and Pearl, 1991) A connected graph $G = (V, E)$ is said to have a* separation vertex *v (sometimes also called an articulation point) if there exist vertices a and b, $a \neq v$ and $b \neq v$, such that all paths connecting a and b pass through v. In this case we also say that v separates a from b. A graph which has a separation vertex is called* separable*, and one which has none is called* non-separable*. Let $V' \subseteq V$. The induced subgraph $G' = (V', E')$ is called a* non-separable component *if G' is non-separable and if for every larger V'', $V \subset V' \subseteq V''$, the induced subgraph $G'' = (V'', E'')$ is separable.*

If the STNs $\mathcal{S}_1, \ldots, \mathcal{S}_n$ temporally decouple $\mathcal{S}$ relative to some set $\mathcal{T}_W$, then, *disregarding edges dominated by paths through zero* (cf. Definition 5.16), z is a separation vertex for every proper, mixed pair in $\mathcal{S}$ (cf. Definition 5.71). The time-points in $\mathcal{T}_W$ need not participate in any such separation relationships.

Whereas Dechter et al. focus on finding the non-separable components of a fixed temporal network, the temporal decoupling problems presented in this chapter focus on finding a set of constraints to *add* to a given network, with the goal of temporally decoupling the network into independent subnetworks, any of which may be separable.

5.6.2 Simple Temporal Networks with Uncertainty

When a single agent *controls* an STN (i.e., has the authority to tighten constraints among any pair of time-points in the network), it is easy for that agent to maintain the consistency of the network. For example, when adding a new constraint, $t_j - t_i \leq \delta$, to the network, the agent need only ensure that δ satisfies $\delta \geq -\mathcal{D}(t_i, t_j)$ (cf. Theorem 4.33).

When different agents control different portions of the network, agents may need to observe a different set of requirements when adding new constraints to the network. For example, as seen in Section 5.5.5, when the agent controlling the leftover set of time-points $\mathcal{T}_W$ in a relative temporal decoupling adds a new constraint to the network, it must observe the *lambda bounds* defined in that section to avoid imposing any additional amount of constrainedness on the decoupled subnetworks controlled by other agents.

Several researchers (Vidal and Fargier, 1999; Morris and Muscettola, 1999; Morris and Muscettola, 2000; Morris, Muscettola, and Vidal, 2001) have investigated temporal networks in single-agent settings in which some of the temporal differences, $t_j - t_i$, are controlled not by the agent, but by nature. The temporal differences controlled by nature are called *contingent durations*, each of which is used to represent some causal process that an agent might initiate but, once initiated, does not control. For example, I can control when my web browser begins the process of fetching baseball scores; and I know that on a good day it will take my browser no more than twenty seconds to fetch the scores; but I cannot control how long it will actually take.

Each contingent duration is constrained to lie within some interval of the form $[\delta_1, \delta_2]$, where $0 < \delta_1 < \delta_2 < \infty$. Thus, associated with each contingent duration is a pair of edges in the distance graph. The *forward edge* is an edge, E_f, of the form, $t_j - t_i \leq \delta_1$; the *reverse edge* is an edge, E_r, of the form, $t_i - t_j \leq -\delta_2$. The pair of edges, E_f and E_r, are also referred to as a *contingent link*. It is assumed that each contingent link corresponds to an *independent* causal process.

An STN extended to include contingent durations is called a *Simple Temporal Network with Uncertainty (STNU)*.[10] If an agent must execute a set of tasks that are dependent on causal processes controlled by nature, as represented by an STNU, then the challenge for the agent is to select execution times for its tasks such that no matter what numbers nature assigns to the contingent durations, which cannot be known in advance, the network will remain consistent over time until all of the tasks are completed. To ensure global consistency, the agent must not impose any additional constrainedness on nature. If it is possible for an agent to respond successfully to this challenge, the network is called *controllable*.

Strong, Dynamic and Weak Controllability of STNUs. Different varieties of STNU controllability arise from different assumptions about when execution times are selected and when information about contingent durations becomes available to an agent. Vidal and Fargier (1999) define the following:

- Strong Controllability: there exists a *single* set of execution times for the agent's tasks that guarantees that the network will remain consistent no matter how nature sets the contingent durations;
- Dynamic Controllability: there is a dynamic strategy for selecting execution times for the agent's tasks in real time, as information about contingent durations becomes available, such that the network will remain consistent; and
- Weak Controllability: for each possible set of values that nature might select for the contingent durations, there is *some* set of execution times for the agent's tasks such that all temporal constraints will be satisfied.

Vidal and Fargier show that the above properties stand in the following relation:

Strong Controllability $\Rightarrow$ Dynamic Controllability $\Rightarrow$ Weak Controllability.

Interestingly, although checking whether a given network has the weak controllability property is Co-NP-complete (Morris and Muscettola, 1999), checking for the strong controllability property can be done in deterministic polynomial time (Vidal and Fargier, 1999), as can checking for the dynamic controllability property (Morris, Muscettola, and Vidal, 2001). Furthermore, Morris, Muscettola and Vidal (2001) provide an algorithm for efficiently *executing* a network that is dynamically controllable (i.e., for selecting execution values for time-point variables in real-time, as information about contingent durations becomes available, such that all constraints are satisfied).

[10]Vidal and Fargier (1999) define a *Simple Temporal Problem under Uncertainty (STPU)* that is essentially equivalent to an STNU.

Waypoint Controllability. Morris and Muscettola (1999) define a *waypoint controllability* property that generalizes both weak and strong controllability. An STNU is called waypoint controllable with respect to a distinguished set W of time-points (called *waypoints*) "if there is a fixed assignment of time values to the [time-points] in W that can be extended to a solution in every projection of [the STNU]." They show that the definition of waypoint controllability reduces to that of weak controllability in the case where the set of waypoints is an arbitrary singleton. Similarly, they show that the definition of waypoint controllability reduces to that of strong controllability in the case where the set of waypoints is the entire set of time-points except for those time-points that end a contingent link without starting a subsequent contingent link (e.g., in a chain of contingent links). The value of having a waypoint controllable network is that it "may be effectively decomposed into (1) an induced STN involving only the waypoints, and (2) several STNUs corresponding to the subnetworks *between* the waypoints" (Morris and Muscettola, 2000). Morris and Muscettola also show that in certain special cases, being waypoint controllable implies being dynamically controllable. Finally, they provide an exponential propagation algorithm that an agent can use in real time when executing tasks in a waypoint controllable STNU (Morris and Muscettola, 1999).

Safe and Potentially Safe Networks. The goal of the above algorithms is to determine whether a given network has some controllability property and, if so, to provide a strategy for executing that network. In contrast, Morris and Muscettola (2000) define *safe* and *potentially safe* networks, and provide algorithms not only for determining whether a given network is potentially safe, but also for converting a potentially safe network into a safe network by adding new constraints to the network.

Morris and Muscettola define an STNU to be *safe*, if "for every valid execution and for each contingent link, the only effective propagations to the contingent link finishing point are those that propagate through the contingent link itself." They point out that safe networks are necessarily dynamically controllable. Morris and Muscettola provide a polynomial algorithm for verifying the safety of an STNU that is based on the notion of dominance presented by Tsamardinos et al. (2000)—which differs from the notion of dominance defined in Section 4.2 in that it is tied to the *dispatchability* of the network (i.e., the ability of the network to be executed in real-time). They show that checking whether a network is *potentially safe* is in $\mathcal{NP}$ and provide an algorithm for converting potentially safe networks into safe networks. They conjecture that the property of being potentially safe is closely related to the dynamic controllability property.

Figure 5.43 summarizes the known relationships among the various controllability properties.

Weak Controllability and Relative Temporal Decouplings. This section shows that an STNU that is weakly controllable is equivalent to an STN whose contingent subnetworks are temporally decoupled relative to the set of time-points directly controlled by the agent. The equivalence depends on modifying the definition of a relative temporal decoupling (cf. Section 5.5.1) to allow separation vertices other than z.

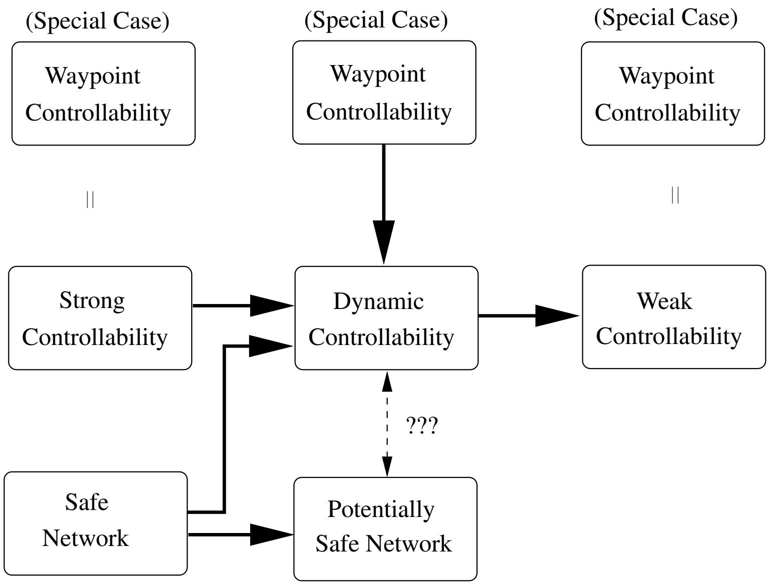


Figure 5.43: Relationships among network controllability properties

Let $\mathcal{S}_U = (\mathcal{T}, \mathcal{C}, \mathcal{C}^c)$ be an STNU that is equivalent to the STN $\mathcal{S} = (\mathcal{T}, \mathcal{C})$, except that the set $\mathcal{C}^c \subseteq \mathcal{C}$ contains constraints representing the contingent links in $\mathcal{S}_U$. As described above, the constraints in $\mathcal{C}^c$ come in pairs,

$$E_f\colon t_j - t_i \leq \delta_1 \qquad \text{and} \qquad E_r\colon t_i \leq t_j \leq -\delta_2,$$

such that $0 < \delta_1 < \delta_2 < \infty$. Notice that the length of the forward edge is positive, whereas the length of the reverse edge negative.

STNUs are not allowed to have multiple contingent links terminating at the same time-point (since doing so would make the STNU uncontrollable). However, multiple contingent links may start at the same point. In addition, contingent links may form chains. Thus, the set of contingent links in an STNU form a disjoint set of *trees*, $T_1, \ldots, T_n$, each tree having a unique root node.

For each tree T_i, let the *contingent subnetwork* corresponding to T_i be the subnetwork $\mathcal{S}_i^c = (\mathcal{T}_i^c, \mathcal{C}_i^c)$, where $\mathcal{T}_i^c \subseteq \mathcal{T}$ is the set of time-points in the tree T_i, and $\mathcal{C}_i^c \subseteq \mathcal{C}_c$ is the set of constraints corresponding to the contingent links in T_i. Notice that, like rc-subnetworks (cf. Definition 4.50), contingent subnetworks need not contain the zero time-point variable, z. Let $\mathcal{T}_W$ be the set of time-points remaining when all of the non-root time-points from the contingent subnetworks are removed from $\mathcal{T}$. Notice that $\mathcal{T}_W$ contains the root time-point for each tree of contingent links.

If the definition of a relative temporal decoupling is modified to allow decouplings in which time-points other than z are permitted to serve as separation vertices, then $\mathcal{S}_U$ is weakly controllable if and only if the contingent subnetworks $\mathcal{S}_1^c, \ldots, \mathcal{S}_n^c$ temporally decouple $\mathcal{S}$ relative to $\mathcal{T}_W$.[11] In such a decoupling, the root node of each tree serves as a

[11]Thus, a corresponding modification to the definition of the lambda bounds in Section 5.5.5 would allow

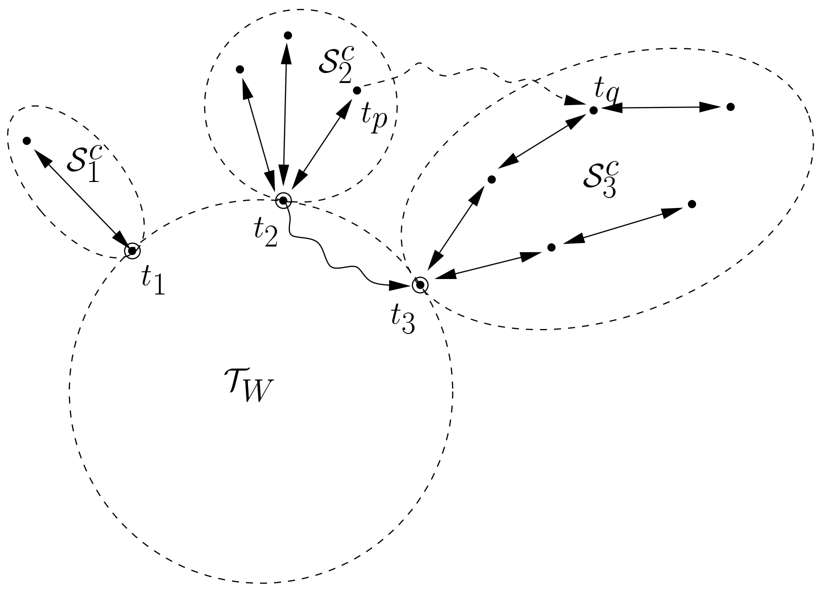


Figure 5.44: A weakly controllable STNU as a temporally decoupled STN.

separation vertex. If the network is temporally decoupled relative to $\mathcal{T}_W$, then any path joining non-root time-points, t_p and t_q, where t_p and t_q belong to different contingent subnetworks, is dominated by a path through the root nodes of the trees containing t_p and t_q, as illustrated in Figure 5.44, where the path from t_p to t_q (shown as a dashed path) is dominated by the path from t_p to t_2 to t_3 to t_q. For convenience, contingent links are shown as double-headed arrows in the figure.

Summary. Although other connections may exist between STNUs and the temporal decoupling problems presented in this chapter, there are substantial differences. First, STNUs presume a single agent, whereas this chapter accommodates arbitrarily many agents. Second, the contingent subnetworks in an STNU have a restricted form, whereas the subnetworks that temporally decouple an STN are not restricted. Third, when adding constraints to an STNU, for example, to make it *safe* (Morris and Muscettola, 2000), an agent is not allowed to impose any additional constrainedness on contingent links. There is no negotiating with nature! However, when seeking to decouple a network involving tasks being done by multiple agents, constraints that impose on each participant may be added if the participants are willing.

Finally, this chapter presumes that the decoupling is done in advance of execution, based on the negotiation of multiple agents, whereas the work cited above considers real-time execution issues, but involves only a single agent. Combining these threads of research to enable groups of negotiating agents to handle real-time execution issues is also left to future work.

---

them to apply to an agent controlling the non-contingent durations in an STNU.

5.6.3 Other Types of Temporal Networks

One of the most attractive features of Simple Temporal Networks is that the operations required to maintain them can be done in polynomial time. However, STNs do not accommodate either disjunctive or conditional constraints, which limits their applicability. Schwalb and Dechter (1997) present polynomial approximation algorithms for temporal networks involving disjunction. Tsamardinos (2001) presents consistency checking algorithms for temporal networks involving disjunctive or conditional constraints. Extending the work in this chapter to cover temporal networks with disjunctive or conditional constraints is left to future work.

Chapter 6

Conclusion

This thesis contributes to both the theory and the implementation of collaborative, multi-agent systems. It contributes to the theory by providing a model of the dynamic function of group decision-making in collaborative activity. It contributes to the implementation by providing an auction-based group decision-making mechanism for solving the Initial-Commitment Decision Problem and a suite of sound, complete and polynomial-time temporal-reasoning algorithms to enable agents to participate effectively in that mechanism. In addition, the formal analysis behind the temporal-reasoning algorithms extends the theory of Simple Temporal Networks (Dechter, Meiri, and Pearl, 1991).

6.1 The Coordinated Cultivation of Group-Related Intentions

In the context of single-agent activity, the process of intention cultivation is under the control of a single agent. In the context of collaborative, multi-agent activity, the process of intention cultivation is more complicated. This thesis presents the CCGI model of the coordinated cultivation of group-related intentions which (1) explicitly represents the general prohibition against unilateral intention updating, (2) specifies how group decisions establish obligations that authorize and oblige member agents to update their intentions together; and (3) shows how the intentions of group members motivate them to participate in group decision-making mechanisms. By explicating the dynamic function of group decision making in the coordinated cultivation of group-related intentions, the CCGI model provides a unifying framework for the web of commitments to group planning processes specified by the SharedPlans formalization of collaborative activity.

To reason effectively about the obligations that might be established by group decision-making mechanisms, such mechanisms must be rigorously defined. This thesis provides the GDMM framework for rigorously specifying such mechanisms in Dynamic Deontic Linear Time Temporal Logic (Dignum and Kuiper, 1997). For each mechanism, the specification includes the classes of declarative speech-acts used in the mechanism and the conditions under which such speech-acts are authorized. To illustrate the framework, a sample, proposal-based mechanism is specified and its properties formally analyzed.

6.2 The Initial Commitment Decision Problem

This thesis defines the Initial-Commitment Decision Problem and specifies a group decision-making mechanism based on a combinatorial auction that agents can use to solve instances of the ICDP. In the auction-based mechanism, which is specified according to the GDMM framework, agents bid on sets of tasks in a proposed activity using temporal constraints in their bids to protect the feasibility of their private schedules of pre-existing commitments. The auction thus serves to mediate between computations that are best done locally, by individuals, and those that require global computations. To participate effectively in such a mechanism, agents need to solve several temporal reasoning problems. The definitions of, and solutions to, these problems are the focus of the second half of the thesis.

6.3 Temporal Reasoning for Collaborative Group Activities

This thesis defines and formally analyzes an important family of temporal decoupling problems. The basic Temporal Decoupling Problem involves searching for constraints to add to a temporal network to partition it into independent subnetworks. The Relative TDP involves partitioning a portion of a temporal network into independent subnetworks. The Allocative TDP involves not only adding temporal constraints, but also allocating time-point to various subnetworks.

For each type of Temporal Decoupling Problem, the problem is formally defined in terms of Simple Temporal Networks (Dechter, Meiri, and Pearl, 1991), theorems characterizing solutions are proven, and a family of sound, complete and polynomial-time algorithms is given. For the special case of two decoupled subnetworks, a *minimal* temporal decoupling is defined and an Iterative Weakening algorithm is provided that, given some arbitrary decoupling, generates a minimal decoupling by iteratively weakening intra-subnetwork constraints.

The algorithms for solving the different Temporal Decoupling Problems may be directly applied to the sorts of temporal reasoning problems that collaborating agents must solve. For example, to generate temporal constraints for bids in the auction-based mechanism, agents can use an algorithm based on the Relative TDP algorithm. Furthermore, when an agent has outstanding bids, it is subject to tighter restrictions on the temporal constraints that it can safely add to its private schedule. These tighter restrictions are specified by the *lambda bounds* that this thesis derives for relative temporal decouplings.

The computations associated with generating bids for an auction-based mechanism depend on the size of the temporal network upon which the bid-generation computations are based. To reduce the burden of bid-generation computations, an agent must be able to limit the portion of its private schedule of pre-existing commitments upon which its bid-generation computations are based. This thesis provides an algorithm for finding approximate solutions to the Allocative TDP that can be directly applied to this problem.

Evan after allocating tasks to various group members as determined by a successful instance of an ICDP auction, there may be temporal dependencies among tasks assigned to different agents. To avoid violating such temporal constraints, agents must be able to coordinate their post-auction activity. This thesis presents several strategies for doing so.

The first strategy involves adding new constraints to the network sufficient to temporally decouple the tasks being done by different agents. This method, which enables each agent to operate independently, is directly related to the General Temporal Decoupling Problem.

The second strategy involves temporally decoupling the tasks being done by *some* of the agents, while the tasks being done by the rest of the agents remain dependent. This method, which allows the agents controlling the decoupled tasks to operate independently, while restricting the freedom of the rest of the agents, is directly related to the Relative Temporal Decoupling Problem.

The restrictions on the freedom of the rest of the agents are specified by the corresponding lambda bounds for the relative temporal decoupling. The properties of the lambda bounds indicate that although the agents controlling the non-decoupled tasks may initially be restricted in the constraints they are allowed to add to the network, once the agents controlling the decoupled tasks decide when they will execute their tasks, the rest of the agents may end up having greater flexibility than they would have had under the first strategy.

The third strategy involves applying the second strategy recursively, resulting in a task-dependency hierarchy that can take advantage of the structure of the temporal dependencies among tasks in the group activity to provide flexibility to the agents that need it the most.