Symbolic domains and reachability for nets with trajectories

Loïc Hélouët, Prerak Contractor

Symbolic domains and reachability for nets with trajectories.

Loïc Hélouët[1] and Prerak Contractor[2]

[1] Univ. Rennes, IRISA, CNRS & INRIA, Rennes, France, loic.helouet@inria.fr
[2] IIT Bombay, Mumbai, India, preerakcontractor@gmail.com

Abstract. This paper considers verification of timed models handling additional quantities progressing linearly such as distance of moving objects to a target. We introduce a variant of Petri nets called *trajectory nets* where some places are standard control places containing tokens, and other places contain a trajectory of an object. We give a semantics for this model, and propose an abstraction of sets of equivalent trajectories into symbolic domains. These domains cannot be represented by Difference Bound Matrices, but one can compute in polynomial time a symbolic representation of successor configurations. Furthermore domains are closed under this successor relation, and the set of domains of a trajectory net is finite. A consequence is that, when the control part of a trajectory net is bounded, reachability, coverability and verification of safety properties involving distances are PSPACE-Complete.

1 Introduction

Some properties of cyber-physical systems such as transport networks call for the verification of quantitative properties addressing time, but also continuous values such as distances. A typical example is safety of metro networks, where one wants to guarantee safety headways, or bound the number of trains in tunnels to guarantee safe evacuation of passengers in case of power failure. Models such as timed automata [3] or time Petri nets [19] only address time, and cannot be used to handle such problems. Models that can address both time and continuous values such as distances rapidly have the expressive power of hybrid automata [2], for which most problems become undecidable.

This paper introduces *trajectory nets*, a model tailored for the analysis of safety properties of systems involving both time and distances such as metro networks. Trajectory nets are a variant of time Petri nets, where some places are dedicated to control, and other places depict object movements with simplified representations called *trajectories*. Configurations assign an integral number of token to control places, and a trajectory, representing the remaining time and distance to the end of a trip to a subset of trajectory places. Dealing with trajectories allows to define properties that address both distances and time. Verification of a safety property of the form "At each instant, less than K trains are in a tunnel" amounts to a reachability question for sets of configurations depicting forbidden positions of objects.

As a first contribution of this paper, we define trajectory nets and give their semantics in terms of configurations, discrete events (end of progress of a trajectory, creation of a new one) and timed moves. As in many continuous models, the set of possible configurations is infinite. This comes on one hand from the unboundedness of the discrete contents of control places, and on the other hand from the continuous representation of trajectories. We show that in their full generality, trajectory nets can simulate a two-counters machine, and are hence Turing Powerful. As a consequence, safety properties relying on coverability or reachability of a configuration are undecidable.

As a second contribution of the paper, we show that the continuous part of configurations can be represented symbolically by sets of linear inequalities called *domains*. Abstracting time with regions and zones in timed automata [3] or domains in variants of Petri nets [5,17,14] is a standard approach. However, for trajectory nets, domains have to abstract away two types of continuous values : time and distance. They define sets of solutions that cannot be represented with the usual zones, and cannot be encoded by Difference Bound Matrices. Nevertheless, we show that we can compute in polynomial time a successor relation on domains, that domains are closed under this relation, and that the set of reachable domains of a given trajectory net is finite. A consequence is that, for trajectory nets with bounded control places, one can compute a sound and complete symbolic abstraction of the timed behaviour called a *state class graph*, and use it to verify properties of the original model. We then show that checking coverability, reachability and safety properties involving distances are PSPACE-Complete problems for bounded trajectory nets.

2 Preliminaries

In the rest of the paper, we will denote respectively by $\mathbb{R}, \mathbb{Q}, \mathbb{N}$ the sets of reals, rationals, and non-negative integers. We will denote by $\mathbb{R}^{\geq 0}, \mathbb{Q}^{\geq 0}$ the sets of positive reals and rationals, and by $\mathcal{I}_{\mathbb{Q}}$ the set of intervals of the form $[a, b]$ or $[a, \infty)$ where $a, b \in \mathbb{Q}$. Let $X = \{x_1, \cdots, x_n\}$ be a set of variables. A *linear constraint* over X with rational coefficients (or simply constraint for short) is an expression of the form $a_1 \cdot x_1 + a_2 \cdot x_2 + \cdots a_n \cdot x_n \leq b$, where b is a rational value and $a_i's$ are rational coefficients (which can have value 0). A constraint is *two-dimensional* if it has at most two variables with non-zero coefficients.

A *valuation* for a set of variables X is a map $\mu : X \to \mathbb{R}$. We will say that a valuation μ *satisfies* a linear constraint $C(X) ::= \sum a_i \cdot x_i \leq b$ iff replacing every x_i by its valuation $\mu(x_i)$ in $C(X)$ yields a tautology.

A *system* of linear constraints over a set of variables X is a set of linear constraints. It is two-dimensional iff all its constraints are two-dimensional, i.e. all its linear inequalities are of the form $a_i \cdot x_i \leq b_i$, or $a_i \cdot x_i - b_j \cdot x_j \leq c_{i,j}$. A valuation satisfies a system S iff it satisfies all linear constraints in S (i.e. systems are conjunctions of constraints over X). A valuation that satisfies S is called *a solution* for S. We will denote by $[\![S]\!]$ the set of solutions for S and say that S is *satisfiable* iff $[\![S]\!] \neq \emptyset$. Slightly abusing our definition, we will sometimes

adopt a compact notation and write $a \leq expr \leq b$ instead of a conjunction of constraints of the form $-expr \leq -a$ and $expr \leq b$.

A two-dimensional system S involving only inequalities of the form $a_i \leq x_i, x_i \leq b_i$, or $x_i - x_j \leq c_{i,j}$ is called a *zone*. It can be encoded by a Difference Bound Matrix (DBM for short) [10], that is a matrix DB_S indexed by $x_\perp, x_1, \cdots, x_n$, where variable $x_\perp$ is a dummy variable representing value 0. In a DBM DB_S, a cell $DB_S[x_i, x_j]$ holding value $b_{i,j}$ encodes inequality $x_i - x_j \leq b_{i,j}$ and a constraint of the form $x_i \leq b_i$ is represented by an entry $DB_S[x_i, x_\perp] = b_i$. Alternatively, zones and DBMs can be represented with *constraint graphs*, i.e. weighted graphs whose vertices are variables $x_\perp, x_1, \cdots, x_n$, and whose edges $(x_i, w_{i,j}, x_j)$ are weighted by $w_{i,j} = DB[x_i, x_j]$. DBMs and their graph representations allow for efficient polynomial algorithms to check satisfiability, compute canonical forms, intersections... (see [6] for a survey). For instance, checking non-emptiness of $[\![S]\!]$ amounts to verifying that the constraint graph for S does not contain negative cycles.

3 Trajectory nets

This section describes a new model called *trajectory nets* that can represent movements of objects in a one-dimensional space. This model can describe conveyors, metro networks,... and refers to positions of objects. We show in section 6 how to address safety properties that depend on simultaneous positions of objects. As explained in introduction, such properties are useful for metro operators to guarantee that passengers can be safely evacuated in a short amount of time in case of power failure. This means setting a limit on the number of trains that are positioned on a dangerous zone (tunnel, bridge...) at any instant, and verifying that this limit is never exceeded. Granting such properties means that one is able to refer to positions of trains.

Petri nets are a rather straightforward choice to model transport networks. Stations and track segments between stations can be represented with places, arrivals and departures of vehicles with transitions, and the structure of the network itself is depicted by the flow relation of the net. Addressing trips and dwells durations can be done via a time(d) extension of Petri nets. For instance, one can use time Petri nets [19], that assign a static interval $[\alpha_t^s, \beta_t^s]$ to every transition to represent possible durations of dwells in stations and of trips from one station to the next one. A standard semantics of Time Petri nets is to keep track at each instant of the time remaining before firing of each enabled transition. However, time(d) variants of Petri nets cannot handle real valued variables beyond timing information related to transitions or tokens, and are hence not expressive enough to address properties referring to positions of vehicles.

Consider, for instance, the pictures in Figure 1. These diagrams are standard representations of metro trajectories called *space-time diagrams*. They represent trips with functions mapping time elapsed with the remaining distance to the next station. The two diagrams on the left represent movements of two trains $train_1, train_2$ traveling respectively on a track from a station S_1 to a station S_2, and from a station S_5 to a station S_6. The track segments represented on these

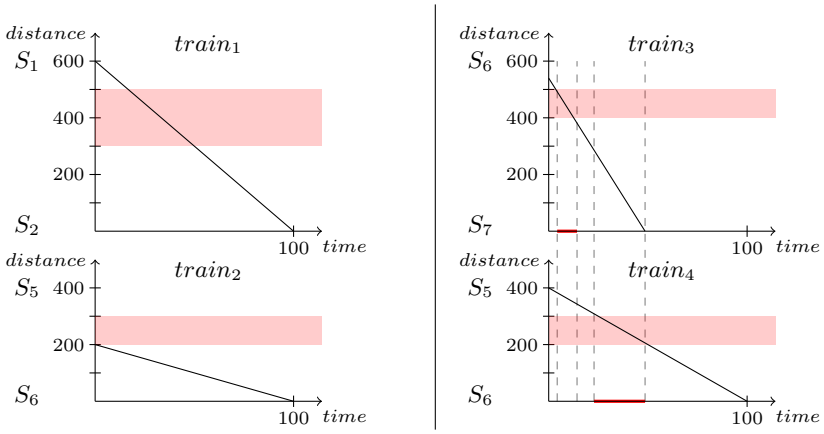

Fig. 1. Space time diagrams: trains positions are needed to address safety issues

diagrams contain dangerous zones, symbolized by red areas. The remaining trip duration for $train_1$ and $train_2$ is 100 in both diagrams, but one can easily notice that $train_1$ moves faster than $train_2$. From this situation, both trains cannot be simultaneously in their danger zone in the future, because $train_2$ already exited this area. So, knowing durations of trips does not suffice to distinguish two trajectories or detect dangers. Now consider the two space-time diagrams on the right of Figure 1. The two trains represented will enter and leave their danger zone. However, as $train_3$ is fast, it will leave its danger zone before $train_4$ enters it. These two examples illustrate the fact that addressing safety of moving objects requires to consider more information than a remaining trip duration for each object. A model such as Time Petri nets [19] is hence not precise enough. A solution is to work with hybrid models, that can handle variables representing evolution of objects positions. It is however well known that most problems are undecidable for hybrid automata [4]. In the rest of this section, we introduce a new model that can address time and distance, without reaching the expressive power of hybrid automata. This model contains transitions that represent objects arrivals or departures, standard places holding tokens, and a new type of place containing space-time diagrams representing forecast trajectories of objects.

Definition 1. *A* Trajectory net *is a tuple* $\mathcal{N} = (P, T, F, I, H)$ *where* $P = P_T \uplus P_C$ *is a set of places. We distinguish a set* P_T *of trajectory places, and a set* P_C *of control places.* $T = \{\sigma_1, \dots \sigma_{|T|}\}$ *is a set of transitions,* $F \subseteq P \times T \cup T \times P$ *is a flow relation. The function* $I : P_T \to \mathcal{I}_{\mathbb{Q}}$ *associates a rational interval* $[\alpha_p^s, \beta_p^s]$ *to every trajectory place* $p \in P_T$*, and the function* $H : P_T \to \mathbb{Q}$ *associates a rational distance to every trajectory place.*

The rational value $H(p) \in \mathbb{Q}^{\geq 0}$ is called the *initial distance* of p, and is the length of the physical space represented by p. The rational interval $I(p) = [\alpha_p^s, \beta_p^s]$ defines the range of possible durations of trips in that physical space. Trajectory places in P_T are holders for trajectories, and control places are standard places

containing tokens, used to allow or forbid firing of transitions. The flow relation of a trajectory net follows the usual terminology of Petri nets. A pair $(p, \sigma) \in F$ from a place $p \in P_C$ to a transition σ means that σ needs a token in place p to fire. Similarly a pair $(p, \sigma) \in F$ with $p \in P_T$ means that σ can fire only if place p contains a trajectory with remaining trip duration (resp. remaining distance to destination) equal to 0. On the other hand, a pair $(\sigma, p) \in F$ indicates that firing of transition σ will produce a fresh token (or a fresh trajectory) in place p. We denote by ${}^\bullet(\sigma) = \{p \in P \mid (p, \sigma) \in F\}$ the *preset* of σ, i.e. the set of places from which σ consumes a token or a trajectory when firing, and by $(\sigma)^\bullet = \{p \in P \mid (\sigma, p) \in F\}$ the *postset* of σ, i.e., the set of places where tokens or trajectories are added when firing σ. In the rest of the paper, to simplify semantics, we consider trajectory nets where $|{}^\bullet(\sigma) \cap P_T| \leq 1$ and $|(\sigma)^\bullet \cap P_T| \leq 1$. Slightly abusing notations, we will hence write $p = {}^\bullet(\sigma) \cap P_T$ instead of $\{p\} = {}^\bullet(\sigma) \cap P_T$, and similarly for $(\sigma)^\bullet \cap P_T$.

Figure 2 shows the basic elements of trajectory net: two trajectory places p_1, p_2 represented by large circles, two control places p_3, p_4, represented by small circles, a transition σ represented by a rectangle. The flow relation is represented as usually in Petri nets with arrows connecting places and transitions. On this example, we have ${}^\bullet(\sigma) = \{p_1, p_3\}$ and $(\sigma)^\bullet = \{p_2, p_4\}$. Place p_3 contains a token, and place p_1 a trajectory.

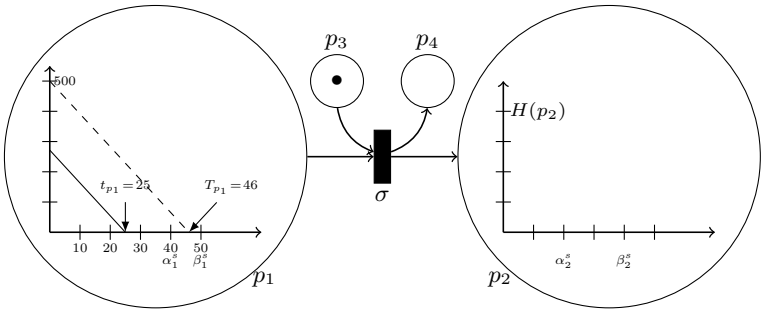

Fig. 2. Basic elements of a trajectory net: places, transitions, trajectories.

Definition 2. *Let $p \in P_T$ be a trajectory place, and $I(p) = [\alpha_p^s, \beta_p^s]$ be the interval depicting the possible duration of a movement in place p. A* trajectory *in p is a pair of real numbers $tr_p = (T_p, t_p)$, where $T_p \in [\alpha_p^s, \beta_p^s]$ denotes the initial duration of a movement in the physical space represented by place p and $t_p \leq T_p$ is the current remaining trip time in that place. A trajectory tr_p is* progressing *if $t_p > 0$ and is* blocked *otherwise.*

As the initial distance $H(p)$ is fixed for every trajectory place $p \in P_T$, choosing non deterministically an initial duration T_p of a trajectory in interval $[\alpha^s, \beta^s]$ amounts to choosing a speed for an object. A consequence is that T_p and t_p provide information on the speed v_p and remaining distance d_p of an object following trajectory (T_p, t_p). We have $v_p = \frac{H(p)}{T_p}$, and $d_p = t_p \cdot \frac{H(p)}{T_p}$. Given a trajectory

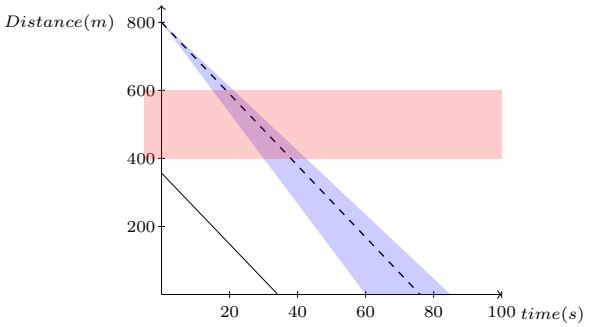

Fig. 3. A Trajectory (T_p, t_p) in a place p with $H(p) = 800m$ and $I(p) = [60, 85]$. We have $T_p = 76$ and $t_p = 34$. The blue cone represents all initial trajectories in p for possible initial values for $T_p \in I(p)$. The red area represents a danger zone (e.g. a tunnel) between distance 400 to 600, that needs to be considered for safety.

place $p \in P_T$, we denote by $\mathcal{T}r(p)$ the set of all trajectories that may appear in p, that is the set of all pairs $\mathcal{T}r(p) = \{(T_p, t_p) \mid T_p \in [\alpha_p^s, \beta_p^s] \wedge 0 \leq t_p < T_p\}$. We assume that represented objects have a constant speed $v = H(p)/T_p$ during their trip, which allows us to represent their trajectories as segments. The semantics of a trajectory net is defined in terms of configurations, that assign an integral number of tokens to control places in P_C, and a trajectory to a subset of places in P_T. We explicitly differentiate blocked and progressing trajectories.

More formally, a *configuration* is a triple $C = (M, B, \mathcal{T})$, where $M : P_C \to \mathbb{N}$ is a marking of control places, $B \subseteq P_T$ is a subset of trajectory places containing blocked trajectories, and $\mathcal{T} : P_T \to \bigcup_{p \in P_T} \mathcal{T}r(p)$ associates a progressing trajectory to a subset of places in P_T. For consistency, we require that $\mathcal{T}(p) \in \mathcal{T}r(p)$. Given a marking, we will write $M \geq {}^\bullet(\sigma)$ iff $M(p) \geq 1$ for every $p \in {}^\bullet(\sigma) \cap P_C$. We will denote by $M - {}^\bullet(\sigma)$ the marking M' such that $M'(p) = M(p)$ for every place $p \notin {}^\bullet(\sigma) \cap P_C$ $M'(p) = M(p) - 1$ for every place $p \in {}^\bullet(\sigma) \cap P_C$. We will denote by $M + (\sigma)^\bullet$ the marking M' such that $M'(p) = M(p)$ for every place $p \notin (\sigma)^\bullet \cap P_C$ and $M'(p) = M(p) + 1$ for every place $p \in (\sigma)^\bullet \cap P_C$.

We represent a configuration $C = (M, B, \mathcal{T})$, with the following graphical convention: we draw $M(p)$ tokens (black dots) in each control place $p \in P_C$, and for a trajectory place $p \in P_T$ we represent trajectory $\mathcal{T}(p) = (T_p, t_p)$ by a dashed segment with coordinates $[(0, H(p)); (T_p, 0)]$ representing the initial trajectory of an object, and a thick segment with coordinates $[(d_p, 0); (0, t_p)]$ representing the remaining displacement. Figure 3, shows a possible trajectory in a place $p \in P_T$ with $H(p) = 800m$, and $I(p) = [60, 85]$. The trajectory represented is (T_p, t_p) with $T_p = 76s$ and $t_p = 34s$. The object moving in the physical space represented by p initially started a trip of duration $76s$ and of length $800m$, represented by the dashed segment in Figure. The speed of this object is $v_p = 800/76 = 10.52m/s$. As the remaining trip duration is $t_p = 34s$, one can easily compute the remaining distance to go for this object, namely $d_p = 357\ m$. The remaining trip is represented by the thick segment in the diagram. Similarly, in Figure 2, let $H(p_1) = 500m$, and $I(p_1) = [40, 50]$. Place

p_1 contains a trajectory (T_{p_1}, t_{p_1}) with $T_{p_1} = 46s$ and $t_{p_1} = 25s$. The remaining distance to go for this object is $d_{p_1} = 271,73m$, and its speed is $v_{p_1} = 10.87m/s$.

The semantics of a trajectory net is defined in terms of timed and discrete moves from a configuration to the next one. The system starts in an initial configuration $C_0 = (M_0, B_0, \mathcal{T}_0)$ such that $B_0 = \emptyset$, and for every p such that $\mathcal{T}_0$ is defined, $\mathcal{T}_0(p) = (T_p^0, t_p^0)$ with $T_p^0 = t_p^0 \in [\alpha_p^s, \beta_p^s]$. The main idea of the semantics is that a transition σ can fire if the control places in the preset ${}^\bullet(\sigma)$ allow firing of σ and the objects in the trajectory places of ${}^\bullet(\sigma)$ have reached their final destination. In other terms, all trajectories in the preset of a fired transition must be blocked.

Upon firing of a transition σ, control tokens in ${}^\bullet(\sigma) \cap P_C$ are consumed, blocked trajectories in ${}^\bullet(\sigma) \cap P_T$ are deleted, and new trajectories in $(\sigma)^\bullet \cap P_T$ and new tokens in control places of $(\sigma)^\bullet \cap P_C$ are created. We adopt an exclusive semantics w.r.t. trajectory places, i.e. a transition σ can fire only if the trajectory places in $(\sigma)^\bullet$ are empty. When modeling metro networks, this semantics is appropriate to represent a fixed block policy, where tracks are divided into exclusive blocks that can contain at most one train at any instant[3]. Upon firing, for each place $p \in (\sigma)^\bullet \cap P_T$ new trajectories are sampled: their initial duration T_p is a value chosen non deterministically in $[\alpha_p^s, \beta_p^s]$ and we set $\mathcal{T}(p) = (T_p, T_p)$. On the other hand, elapsing time allows for the progress of existing trajectories. However, we consider an urgent semantics, that allows elapsing $\delta > 0$ time units in a configuration C only if no discrete firing can occur in C.

Discrete moves

Discrete moves are either the blocking of a trajectory or the firing of a transition. Blocking a trajectory $tr_p = (T_p, t_p)$ in place p is possible only if $t_p = 0$, and consists in deleting tr_p, and adding p to the list of places containing a blocked trajectory. Formally, it is defined by the following operational semantics rule:

$$\frac{\begin{array}{c} p \in P_T \\ \mathcal{T}(p) = (x, 0) \text{ for some } x \in [\alpha_p^s, \beta_p^s] \\ \mathcal{T}'(p_i) = \begin{cases} \mathcal{T}'(p_i) \text{ if } p_i \neq p \\ \text{is undefined otherwise} \end{cases} \\ B' = B \cup \{p\} \end{array}}{C = (M, B, \mathcal{T}) \overset{block\ p}{\longrightarrow} C' = (M, B', \mathcal{T}')}$$

We will say that a transition σ is *firable* in a configuration $C = (M, B, \mathcal{T})$ iff $M \geq {}^\bullet(\sigma)$, the trajectory in ${}^\bullet(\sigma) \cap P_T$ is blocked, and the place depicting the physical space needed to perform action σ is free, that is, $\forall p \in (\sigma)^\bullet \cap P_T$, $p \notin B$ and $\mathcal{T}(p)$ is undefined. The effect of a firing σ is the consumption of all tokens in ${}^\bullet(\sigma) \cap P_C$, the production of a new token in each place of $(\sigma)^\bullet \cap P_C$, the deletion of the blocked trajectory in ${}^\bullet(\sigma) \cap B$, and the creation of a new trajectory $\mathcal{T}(p') = (T_{p'}, T_{p'})$ in place $p' = (\sigma)^\bullet \cap P_T$ with $T_{p'} \in [\alpha_{p'}^s, \beta_{p'}^s]$. We will write $M[\sigma\rangle M'$ when M' is the marking obtained after firing of σ from M,

[3] Though this fixed block semantics may seem very constrained, many metro networks in the world are operated this way.

i.e. when $M' = M - {}^\bullet(\sigma) + (\sigma)^\bullet$. Then, the firing of a transition σ is formally defined by the following operational semantics rule:

$$\frac{\begin{array}{c} M \geq {}^\bullet(\sigma) \wedge M[\sigma\rangle M' \\ \forall p \in (\sigma)^\bullet \cap P_T, p \notin B \wedge \mathcal{T}(p) \text{ is undefined} \\ {}^\bullet(\sigma) \cap P_T \subseteq B \ \wedge \ B' = B \setminus {}^\bullet(\sigma) \\ \mathcal{T}'(p) = \begin{cases} (T_p, T_p), \text{ with } T_p \in [\alpha_p^s, \beta_p^s] \text{ if } p = (\sigma)^\bullet \cap P_T \\ \mathcal{T}(p) \text{ otherwise} \end{cases} \end{array}}{C = (M, B, \mathcal{T}) \xrightarrow{\sigma} C' = (M', B', \mathcal{T}')}$$

When a new trajectory is added in a trajectory place, we necessarily have $d_p = H(p)$. As $H(p)$ is constant, choosing T_p is equivalent to choosing a speed v_p for a vehicle, and memorizing this initial choice and t_p allows to compute d_p after several timed moves.

Timed moves

The effect of time elapsing is to reduce the remaining trip time of progressing transitions. Elapsing δ time units is allowed if this duration does not exceed remaining trip time of any progressing trajectory. As in Time Petri nets [19], we adopt an *urgent semantics*, that is we forbid time progress if a discrete event can occur. Time progress of δ is hence forbidden if some transition is firable less than δ time units after the current date, or if a trajectory gets blocked. For a given description of trajectories $\mathcal{T}$, we denote by $\mathcal{T} + \delta$ the function that associates the pair $(\mathcal{T} + \delta)(p) = (T_p, t_p - \delta)$ with place p if $\mathcal{T}(p) = (T_p, t_p)$.

$$\frac{\begin{array}{c} 0 < \delta \leq \min\{t_p \mid \exists T_p, (T_p, t_p) \in \mathcal{T}(P_T)\} \\ \forall \sigma \in T, \sigma \text{ is not firable} \end{array}}{C = (M, B, \mathcal{T}) \xrightarrow{\delta} C' = (M, B, \mathcal{T} + \delta)}$$

Obviously, our semantics enjoys time additivity, i.e. if $C_1 \xrightarrow{\delta_1} C_2$ and $C_2 \xrightarrow{\delta_2} C_3$, then $C_1 \xrightarrow{\delta_1+\delta_2} C_3$. Notice also that timed and discrete moves are exclusive. It is hence natural to describe runs of a trajectory net as an alternation of timed and discrete moves. A *run* of a trajectory net from a configuration $C_0 = (M_0, B_0, \mathcal{T}_0)$ is a sequence of timed and discrete moves $\rho = (M_0, B_0, \mathcal{T}_0) \xrightarrow{\delta_0} (M_0, B_0, \mathcal{T}_0 + \delta_0) \xrightarrow{e_1} (M_1, B_1, \mathcal{T}_1) \cdots$, where each move of the form $(M_i, B_i, \mathcal{T}_i) \xrightarrow{\delta_i} (M_i, B_i, \mathcal{T}_i + \delta_i)$ is a legal timed move, and $(M_i, B_i, \mathcal{T}_i) \xrightarrow{e_i} (M_{i+1}, B_{i+1}, \mathcal{T}_{i+1})$ is a legal discrete move, that is a blocking of a trajectory, $(M_i, B_i, \mathcal{T}_i) \xrightarrow{block\ p} (M_{i+1}, B_{i+1}, \mathcal{T}_{i+1})$ or a firing of a transition $(M_i, B_i, \mathcal{T}_i) \xrightarrow{\sigma_i} (M_{i+1}, B_{i+1}, \mathcal{T}_{i+1})$.

We will write $(M, B, \mathcal{T}) \xrightarrow{*} (M', B', \mathcal{T}')$ if there exists a sequence of discrete and timed moves leading from $(M, B, \mathcal{T})$ to $(M', B', \mathcal{T}')$. Without loss of generality, we assume that a net starts in an initial configuration $C_0 = (M_0, B_0, \mathcal{T}_0)$ without blocked trajectories, i.e. such that $B_0 = \emptyset$. We also assume that the trajectories of all trains are at their beginning, i.e. for every $p \in P_T$ where $\mathcal{T}_0(p)$ is defined, we have $\mathcal{T}_0(p) = (T_p^0, T_p^0)$ for some $T_p^0 \in [\alpha_p^s, \beta_p^s]$, and that all trains still have some remaining time before being blocked, i.e. $T_p^0 > 0$. We will denote

by $Reach(C_0)$ the set of reachable configurations, i.e. $Reach(C_0) = \{(M, B, \mathcal{T}) \mid C_0 \xrightarrow{*} (M, B, \mathcal{T})\}$. We will say that a trajectory net is *bounded* iff, there exists an integer K such that for every configuration $(M, B, \mathcal{T})$ in $Reach(C_0)$, and for every place $p \in P_C$, $M(p) \leq K$.

We will address reachability problems, i.e. study whether a particular configuration $(M, B, \mathcal{T})$ is reachable. Now, asking whether a configuration $(M, B, \mathcal{T})$ is reachable refers to the exact position of objects, and is a too precise question. To cope with this problem, we transform contents of trajectory places into markings as follows: given a configuration $C = (M, B, \mathcal{T})$ we define a *complete marking* M_C that associates an integral number of tokens to each place in P, such that $M_C(p) = M(p)$ if $p \in P_C$, and $M_C(p) = 1$ if $p \in P_T$ and $\mathcal{T}(p)$ is defined or if $p \in B$. We then differentiate three decision problems:

- *exact reachability*: for a given configuration C, does $C \in Reach(C_0)$?
- *boolean reachability*: for a given configuration C, is there a configuration $C' \in Reach(C_0)$ such that $M_C = M_{C'}$?
- *coverability*: for a given configuration C, is there a configuration $C' \in Reach(C_0)$ such that $M_C \leq M_{C'}$?

Coverability can be used to check that at a given instant, two trains cannot occupy rail sections that require mutual exclusion. Note however that coverability is not fine enough to define safety properties involving distances. For instance, one cannot use coverability to check existence of configurations where two objects are simultaneously in a critical area (a red zone similar to those in the diagrams of Figure 1). As explained at the beginning of this Section, this is a standard property to ensure in metro networks to guarantee passengers safety. In Section 6, we will show that we can address such properties involving distances.

Reachability is decidable for timed automata, using a abstraction of clock values and building a *region automaton* [3]. Reachability and coverability are undecidable in general for time Petri nets [15]. Coverability is decidable for timed-arc Petri nets [1,13], but reachability remains undecidable [22]. For a weak interpretation of TPNs semantics, [21] shows that the set of reachable markings of a TPN $\mathcal{N}$ coincides whith the set of reachable markings of an untimed version of $\mathcal{N}$. It is well known that coverability [20] and reachability [18] are decidable for Petri nets, and hence also for TPNs with a weak semantics.

Inspiring from [21], we can easily transform a trajectory net $\mathcal{N} = (P = P_C \cup P_T, T, F, I, H)$ and its initial configuration $C_0 = (M_0, B_0, \mathcal{T}_0)$ into a standard untimed Petri net $\mathcal{U}(\mathcal{N}) = (P', T, F)$ where $P' = P_C \cup \{p^U \mid p \in P_T\}$ replaces every trajectory place by a standard place, with initial marking $M_0^{\mathcal{U}} = M_{C_0}$. However, in general $\mathcal{U}(\mathcal{N})$ allows more markings and more runs than $\mathcal{N}$. Consider the example of Figure 4, with configuration $C_0 = (M_0, B_0, \mathcal{T}_0)$ with $M_0(p_3) = 1, M_0(p_4) = M_0(p_5) = 0$, $\mathcal{T}_0(p_1) = (1, 1)$, $\mathcal{T}_0(p_2) = (2, 2)$, and its translation to a standard Petri net $\mathcal{U}(\mathcal{N})$ on the right of the Figure. One can see from this example that in the initial configuration C_0, transition σ_1 is firable, but transition σ_2 will never fire. On the other hand, in $\mathcal{U}(\mathcal{N})$, both transitions σ_1 and σ_2 can fire from M_{C_0}. This example shows that erasing time makes some new markings

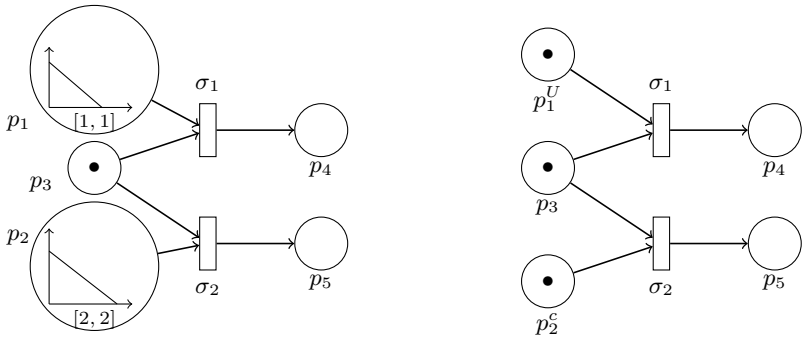

Fig. 4. A trajectory net and its untimed abstraction into a Petri net.

reachable. Hence we cannot rely on a simple untiming of a trajectory net to address reachability or coverability. We can further show that trajectory nets are powerful enough to model a two-counter machine, yielding undecidability of most problems.

Theorem 1. *Reachability, boolean reachability and coverability are undecidable for trajectory nets.*

Proof (Sketch). We can simulate the behavior of an unbounded two-counter machine with an unbounded trajectory net. As in [21], zero tests are simulated by enabling simultaneously two transitions $\sigma_{i,Z}, \sigma_{i,NZ}$, and forcing $\sigma_{i,NZ}$ to fire urgently before $\sigma_{i,Z}$ if a place simulating the tested counter is not empty. The complete encoding is provided in Appendix B.□

A standard way to recover decidability of reachability and coverability in timed extensions of Petri nets is to restrict to *bounded* nets, and define a symbolic abstraction of timing information allowing a finite partition of the space of configurations. For time Petri nets, where time is measured by clocks attached to enabled transitions, [5] defines an abstraction called state classes, that are equivalence classes for sets of configurations with identical markings and equivalent constraints on the values of clocks. These constraints are called *domains*, and can be compared to zones or regions of timed automata [3]. In the next section, we consider state classes and domains for trajectory nets, and give a sound abstraction of continuous values appearing in configurations.

4 Domains

In this section we define domains for trajectory nets. Domains are a way to define symbolically the value of initial and remaining trajectory durations with positive real valued variables. For a given configuration $C = (M, B, \mathcal{T})$, we have two types of trajectories: blocked trajectories, and progressing ones. For blocked trajectories, the remaining running time is known (it is 0), and the initial time is of no use to define a position of an object: it is at distance 0 w.r.t the end

of the represented space. For progressing trajectories, i.e. trajectories in a place $p \in P_T$ for which $\mathcal{T}(p)$ is defined, we have $\mathcal{T}(p) = (T_p, t_p)$. As already mentioned in Section 3, we only need these two variables T_p and t_p to compute the speed or the current position of an object, which are important information when considering some safety properties.

Definition 3 (Domains). *Let $\mathcal{N}$ be a trajectory net, with set of trajectory places P_T. Let $P \subseteq P_T$ represent places with progressing trajectories. Then, a domain for $\mathcal{N}$ with progressing trajectories in P is a set of inequalities D over variables $V_D = \{T_i, t_i \mid i \in P\}$, of the form:*

$$\alpha_i^1 \leq T_i \leq \beta_i^1 \text{ for all } i \in P \tag{1}$$

$$\alpha_i^2 \leq t_i \leq \beta_i^2 \text{ for all } i \in P \tag{2}$$

$$t_i - t_j \leq \gamma_{ij}^3 \text{ for all } i \neq j \tag{3}$$

$$\alpha_i^4 \leq T_i - t_i \leq \beta_i^4 \text{ for all } i \in P \tag{4}$$

$$\alpha_{ij}^5 \leq T_i - t_i + t_j \leq \beta_{ij}^5 \text{ for all } i \neq j \tag{5}$$

$$-T_i + t_i + T_j - t_j \leq \gamma_{ij}^6 \text{ for all } i \neq j \tag{6}$$

*where each type of inequality $(1-6)$ appears **exactly** once in D for each $i \in P$ and for each pair of distinct places $i, j \in P$, and $\alpha_i^1, \alpha_i^2, \alpha_i^4, \alpha_i^5$. $\beta_i^1, \beta_i^2, \beta_i^4, \beta_i^5$ $\gamma_{ij}^3, \beta_{ij}^5, \gamma_{ij}^6$ are either constant values, $-\infty$, or $+\infty$.*

Domains in Definition 3 are systems of linear inequalities, but inequalities of type (5) involve expressions with three variables, and inequalities of type (6) use expressions with four variables. Consequently, our domains are not two-dimensional, and differ from the domains proposed by [5]. Further, they **cannot be encoded using DBMs**. We will however show in the rest of the paper that these domains can be efficiently manipulated in polynomial time. Following the definitions of Section 2, we will say that a valuation $\mu : V_D \to \mathbb{R}$ for V_D is a *solution* for D iff replacing variables T_i, t_i in V_D by their values $\mu(T_i), \mu(t_i)$ yields a tautology, and denote by $[\![D]\!]$ the set of all solutions for D. Slightly abusing our notation, we will write $\mathcal{T} \in [\![D]\!]$ when the valuation $\mu_{\mathcal{T}}$ that associates variables $\{T_i, t_i\}$ with their respective values in $\mathcal{T}$ is a solution of D. We will say that two domains D_1, D_2 are equivalent iff $[\![D_1]\!] = [\![D_2]\!]$. Even if two domains are equivalent, they may have different representations. Indeed, consider a single pair of variables $T_1 = t_1$ whose values lie in the interval $[3, 4]$. We can represent the constraint on T_1, t_1 as $D_1 = \{3 \leq T_1 \leq 12; 0 \leq T_1 - t_1 \leq 0; 0 \leq t_1 \leq 4\}$. However the domain $D_2 = \{0 \leq T_1 \leq 4; 0 \leq T_1 - t_1 \leq 0; 3 \leq t_1 \leq 20\}$ represents the same set of solutions. We can show that a *canonical form* for domains exists (Proposition 1) , and can be computed in polynomial time (Proposition 2).

Definition 4. *Let $D = \{a_i \leq expr_i \leq b_i\}$ be a domain of the form given in Definition 3. Then, the* canonical form *for D is a domain $D^* = \{a_i^* \leq expr_i \leq b_i^*\}$, where a_i^* is the smallest value taken by $expr_i$ in $[\![D]\!]$, and b_i^* is the largest value taken by $expr_i$ in $[\![D]\!]$.*

Proposition 1. *The canonical form of a domain D is unique and preserves $[\![D]\!]$.*

Proof (Sketch). A domain D is a set of inequalities of the form $a_i \leq expr_i$, and $expr_j \leq b_j$. A solution $\mu_S \in [\![D]\!]$ is a map that associates a real value with every variable t_i (resp. T_i). Let $\mu_S(expr_i)$ be the value obtained by replacing every variable t_i by $\mu_S(t_i)$ and T_i by $\mu_S(T_i)$ in $expr_i$. The values $a_i^* = \min_{\mu_S \in [\![D]\!]} \mu_S(expr_i)$ and $b_j^* = \max_{\mu_S \in [\![D]\!]} \mu_S(expr_j)$ are unique, so D^* is uniquely defined. Every expression of the form $a_i \leq expr_i \leq b_i$ can be equivalently rewritten as $a_i^* \leq expr_i \leq b_i^*$ because $a_i^* \leq \mu(expr_i) \leq b_i^*$ for every $\mu \in [\![D]\!]$, and we have $a_i \leq a_i^* \leq b_i^* \leq b_i$. A complete proof is given in appendix C. □

As all domains over a fixed set of progressing trajectories have the same types of inequalities, and differ only by the constants used, a direct consequence of Proposition 1 is that two domains D, D' are equivalent if and only if $D^* = D'^*$.

Proposition 2. *The canonical form for a domain D can be computed in PTIME.*

Proof. We perform the following linear transformation: $x_i = T_i - t_i$ and $y_i = -t_i$ to get a new set of inequalities:

$$\begin{array}{rcl} \alpha_i^1 \leq x_i - y_i \leq \beta_i^1 \\ -\beta_i^2 \leq \quad y_i \quad \leq -\alpha_i^2 \\ y_j - y_i \leq \gamma_{ij}^3 \end{array} \qquad \begin{array}{rcl} \alpha_i^4 \leq \quad x_i \quad \leq \beta_i^4 \\ \alpha_{ij}^5 \leq x_i - y_j \leq \beta_{ij}^5 \\ -x_i + x_j \leq \gamma_{ij}^6 \end{array}$$

Notice that our linear transformation is bijective. Hence, for any solution $\mu : \{T_i, t_i\} \to \mathbb{R}$ in the original domain, there exists a **unique** solution μ' such that $\mu'(x_i) = \mu(T_i) - \mu(t_i), \mu'(y_i) = -\mu(t_i)$ and for any solution $\mu' : \{x_i, y_i\} \to \mathbb{R}$ in the new domain, there exists a **unique** solution μ such that $\mu(T_i) = \mu'(x_i) - \mu'(y_i)$, and $\mu(t_i) = -\mu'(y_i)$. Hence there is a bijection between the two domains.

Observe that this new domain is of dimension 2. It can hence be encoded as a DBM or a constraint graph, and finding a canonical form for this new domain can be done by computing the shortest paths in the constraint graph. The cost of this calculus is in $O(n^3)$ for n variables. The optimal bounds obtained for the domain over variables $\{x_i, y_i\}$ are bounds for expressions of the form $x_i - y_i$, $x_i - y_j$, etc. that directly encode expressions of the original domain D (for instance $x_i - y_i = T_i$, and $x_i - y_j = T_i - t_i + t_j$). The sharp bounds obtained for the new domain can hence be immediately used as optimal bounds for D, except for the expression of the form $-(\beta_i^2)^* \leq y_i \leq -(\alpha_i^2)^*$, where we need a sign inversion to obtain $(\alpha_i^2)^* \leq t_i \leq (\beta_i^2)^*$.

For a domain D addressing properties of k trajectories, the linear transformation of D is in $O(k^2)$, as we have $3 \cdot k + 3 \cdot k^2$ inequalities D, and performing the transformation for each inequality takes constant time. Computing the canonical form of the new domain can be done in $O(k^3)$ time using the Floyd-Warshall algorithm, as the new domain has $2 \cdot k$ variables. Hence, the canonical form of D can be computed in $O(k^3)$. Note that the constants obtained in the canonical form are linear combinations of α_i^s and β_i^s with integer coefficients. □

Let $\mathcal{N}$ be a trajectory net, and let $P \subseteq P_T$ be the set of places containing progressing trajectories in initial configuration C_0. The initial domain D_0 for $\mathcal{N}$ and P is the set:

$$D_0 = \begin{cases} T_i^0 \leq T_i \leq T_i^0 & \text{for all } i \in P \\ T_i^0 \leq t_i \leq T_i^0 & \text{for all } i \in P \\ t_i - t_j \leq \infty & \text{for all } i \neq j \\ 0 \leq T_i - t_i \leq 0 & \text{for all } i \in P \\ -\infty \leq T_i - t_i + t_j \leq \infty & \text{for all } i \neq j \\ -T_i + t_i + T_j - t_j \leq \infty & \text{for all } i \neq j \end{cases}$$

Let μ_0 be the valuation such that $\mu_0(T_p) = \mu_0(t_p) = T_p^0$ for every place where $\mathcal{T}_0$ is defined. Obviously, $[\![D_0]\!] = \{\mu_0\}$. The initial domain D_0 meets the requirements of Definition 3. To show that the form of domain of Definition 3 is sufficient to represent all domains of a net, it remains to show that the effect of a transition firing, or of a trajectory blocking after some delay δ as proposed in the semantics of Section 3 can be encoded through algebraic operations (variable changes, unions of inequalities and projections) that preserve the types of inequalities considered in Definition 3.

4.1 Successors after firing a transition

Let D be a domain with set of progressing trajectories P. We want to compute the set of constraints on variables attached to progressing trajectories of the net after firing a transition σ. Let $p = {}^\bullet(\sigma) \cap P_T$ and $p' = (\sigma)^\bullet \cap P_T$. First, σ can fire only if p is an empty place (a trajectory in p was formerly blocked), and p' is also empty. According to our semantics, adding a trajectory in p' means sampling a new trip duration $T_{p'} \in [\alpha^s_{p'}, \beta^s_{p'}]$ and adding in p' a new progressing trajectory $(T_{p'}, T_{p'})$. The sampled value is totally independent from the values of variables in D, so the new set of constraint on progressing trajectories after firing of σ is the set:

$$\begin{aligned} SuccF(D,\sigma) = D \cup \{\alpha^s_{p'} \leq T_{p'} \leq \beta^s_{p'}\} \cup \{\alpha^s_{p'} \leq t_{p'} \leq \beta^s_{p'}\} \cup \{0 \leq T_{p'} - t_{p'} \leq 0\} \\ \cup \{t_{p'} - t_i \leq \infty \mid i \in P\} \cup \{t_i - t_{p'} \leq \infty \mid i \in P\} \\ \cup \{-\infty \leq T_{p'} - t_{p'} + t_i \leq \infty \mid i \in P\} \\ \cup \{-\infty \leq T_i - t_i + t_{p'} \leq \infty \mid i \in P\} \\ \cup \{-T_{p'} + t_{p'} + T_i + t_i \leq \infty \mid i \in P\} \\ \cup \{-T_i + t_i + T_{p'} + t_{p'} \leq \infty \mid i \in P\} \end{aligned}$$

One can immediately notice that if D is a domain, then so is $SuccF(D,\sigma)$.

4.2 Successors after blocking a trajectory

Blocking a progressing trajectory $tr_p = (T_p, t_p)$ from a configuration occurs after elapsing $\delta = t_p$ time units, and is allowed if δ is the minimal duration among all progressing trajectories. We hence have to consider transformations on a domain D occurring after a sequence of timed and discrete moves of the form

$C \xrightarrow{\delta} C' \xrightarrow{block\ p} C''$. Remark, from the semantics that $\delta = t_p$, so blocking of tr_p can occur only if $t_p = \min\{t_j \mid \exists p_j \in P_T, \mathcal{T}(p_j) = (T_j, t_j)\}$. This requirement can be easily translated into a new constraint: trajectory tr_p can be blocked from some configuration satisfying domain D iff $D^{p\leq *} ::= D \cup \{t_p \leq t_j \mid j \neq p \wedge (T_j, t_j)$ is a progressing trajectory$\}$ is satisfiable.

As a blocked trajectory does not constrain any more possible durations of other trajectories, the domain capturing the remaining constraints in configuration C'' is the projection on remaining variables once t_p time units have elapsed. To obtain this set of constraints, we proceed as follows:

- We make a variable change. Let t'_j denote a variable representing the new value of remaining travel time of trajectory j after elapsing t_p time units. Then we have $t'_j = t_j - t_p$. We hence replace every variable t_j by $t'_j + t_p$ in every inequality of $D^{p\leq *}$. Let us call D' this new domain.
- We eliminate variables T_p and t_p from domain D'. This elimination can be done in polynomial time using the well-known Fourier-Motzkin algorithm (see Appendix A and [9]).
- We replace every occurrence of a variable t'_j by an unprimed variable t_j to obtain a successor domain $SuccB(D, p)$, and we compute its canonical form.

Proposition 3. *Let D be a domain of a trajectory net $\mathcal{N}$, and let D' be a system of linear inequalities that is a successor of D via construction of $SuccB(D, p)$ or $SuccF(D, \sigma)$. Then D' is a domain of $\mathcal{N}$.*

Proof (Sketch). $SuccF(D, \sigma)$ trivially satisfies this property, as it only adds constraints of the form $\alpha_i^S \leq T_i \leq \beta_i^S$, $\alpha_i^S \leq t_i \leq \beta_i^S$, and $T_i = t_i$. The proof for $SuccB(D, p)$ is more involved, as it requires eliminating variables for the blocked trajectory. Yet, during elimination, some inequalities are unchanged because they refer to T_i, and not to t_i. For other inequalities, combining expressions of the form $expr_j \leq t_i$ and $t_i \leq expr_k$ to obtain a new expression $expr_j \leq expr_k$ during the elimination process either produces tautologies, or new expressions that are of the form of inequalities in Definition 3. A complete proof is given in Appendix D.□

5 Soundness, Completeness, Finiteness

Now that we have defined domains for trajectory nets, and shown that we can effectively compute a canonical representation for $SuccF(D, \sigma)$ the set of constraints that hold after firing a transition σ and $SuccB(D, p)$ the constraints that hold after blocking a trajectory in place p when starting from a domain D, we can define state classes.

Definition 5. *A* state class *of a trajectory net $\mathcal{N}$ is a triple $SC = (M, B, D)$, where M is a marking, B is a subset of trajectory places with blocked trajectories, and D is a domain of $\mathcal{N}$ in canonical form.*

We can define a symbolic transition relation among state classes as follows:

- $(M, B, D) \stackrel{Block\ p}{\longrightarrow_S} (M, B', D')$ if $B' = B \cup \{p\}$, and D' is the canonical representation of $SuccB(D, p)$, and
- $(M, B, D) \stackrel{\sigma}{\longrightarrow_S} (M', B', D')$ if $M[\sigma\rangle M'$, $B' = B \setminus {}^\bullet(\sigma)$, and D' is the canonical representation of $SuccF(D, \sigma)$.

We will write $(M, B, D) \longrightarrow_S (M', B', D')$ if either $(M, B, D) \stackrel{Block\ p}{\longrightarrow_S} (M', B', D')$ or $(M, B, D) \stackrel{\sigma}{\longrightarrow_S} (M', B', D')$. We will denote by $Reach^S(M_0, B_0, D_0)$ the set of state classes that can be built inductively from the initial sate class D_0 by application of the symbolic transition relation $\longrightarrow_S$.

Definition 6. *The state class graph of a trajectory net $\mathcal{N}$ is the transition system $SC(\mathcal{N}) = \big(Reach^S(M_0, B_0, D_0), \longrightarrow_S, (M_0, B_0, D_0)\big)$.*

Notice that $SC(\mathcal{N})$ is defined even if $Reach^S(M_0, B_0, D_0)$ is not finite. We will say that a configuration $C = (M, B, \mathcal{T})$ *matches* with a state class $SC = (M', B', D)$ iff $M = M'$, $B = B'$ and $\mathcal{T} \in [\![D]\!]$.

Definition 7. *A* symbolic run *of $\mathcal{N}$ is a sequence of state classes of the form $\rho^S = (M_0, B_0, D_0) \stackrel{e_0}{\longrightarrow} (M_1, B_1, D_1) \stackrel{e_1}{\longrightarrow} \ldots$ such that for every index $i \geq 0$, $e_i \in \{Block\ p_i, \sigma_i\}$ and $(M_i, B_i, D_i) \stackrel{e_i}{\longrightarrow_S} (M_{i+1}, B_{i+1}, D_{i+1})$.*

Proposition 4 (Soundness). *Let $\rho^S = (M_0, B_0, D_0) \stackrel{e_0}{\longrightarrow} (M_1, B_1, D_1) \stackrel{e_1}{\longrightarrow} \ldots$ be a symbolic run of a trajectory net $\mathcal{N}$. Then, there exists a run $\rho = (M_0, B_0, \mathcal{T}_0) \stackrel{e_0}{\longrightarrow} (M_1, B_1, \mathcal{T}_1) \stackrel{e_1}{\longrightarrow} \ldots$ of $\mathcal{N}$ such that for every $i \geq 0$, $(M_i, B_i, \mathcal{T}_i)$ matches with (M_i, B_i, D_i).*

Proposition 5 (Completeness). *Let $\rho = (M_0, B_0, \mathcal{T}_0) \stackrel{e_0}{\longrightarrow} (M_1, B_1, \mathcal{T}_1) \stackrel{e_1}{\longrightarrow} \ldots$ be a run of a trajectory net $\mathcal{N}$. Then, there exists a symbolic run $\rho^S = (M_0, B_0, D_0) \stackrel{e_0}{\longrightarrow} (M_1, B_1, D_1) \stackrel{e_1}{\longrightarrow} \ldots$ of $\mathcal{N}$ such that for every $i \geq 0$, $(M_i, B_i, \mathcal{T}_i)$ matches with (M_i, B_i, D_i).*

The proofs for Propositions 4 and 5 are obtained by induction on the length of runs, and are detailed in Appendix E.

Theorem 2. *Let $\mathcal{N}$ be a bounded trajectory net, with initial configuration C_0. Then the set of canonical domains that can be computed inductively from D_0^* is finite.*

Proof. A domain is defined by a set of inequalities. The number of inequalities depend only on the number of progressing trajectories, and these inequalities involve constants. We can prove (see Lemma 1 in Appendix F) that constants appearing in canonical domains of a trajectory net are linear combinations of constants appearing in D_0 and in intervals $[\alpha_i^s, \beta_i^s]$. It was also proved (see Lemma 2 in Appendix, and [5]) that the number of linear combinations of a finite set of constants in a rational interval $[A, B]$ is finite. It remains to show that this type of bounding interval exists for the values of constants $\alpha_i^1, \beta_i^1, \alpha_i^2, \beta_i^2, \gamma_{ij}^3, \alpha_i^4, \beta_i^4, \alpha_{ij}^5, \beta_{ij}^5, \gamma_{ij}^6$ appearing in domains of trajectory nets. We

show in Lemma 3 in Appendix that this interval is $[-2 \cdot C_{max}, 2 \cdot C_{max}]$, where C_{max} is the maximal value appearing in an interval $[\alpha_i^s, \beta_i^s]$.

Then, combining Lemma 1, Lemma 2, and Lemma 3 we get that constants in domains can take a finite number of values. Hence only a finite number of inequalities appear in domains. Let us denote by I_0 this set of possible inequalities. The set of possible domains is finite, since each domain is a subset of I_0. □

Following Theorem 2, we can give an upper bound on the number of state classes in $SC(\mathcal{N})$. Let us first compute the size of I_0. Assuming that we consider only domains in canonical form, every constraint in I_0 is of the form $a \leq expr \leq b$, with $-2 \cdot C_{max} \leq a$ and $b \leq 2 \cdot C_{max}$. Assuming that all α_i^s and β_i^s are rational numbers with a common denominator d, there exists at most $4 \cdot C_{max}.d$ possibilities for values of a and b in expressions. Similarly, expression $expr$ are of the form given in Definition 3, and there are hence $3 \cdot (|P_T| + |P_T|^2)$ expressions. The size of I_0 is hence $12 \cdot C_{max} \cdot d \cdot (|P_T| + |P_T|^2)$, and each domain is a subset of inequalities from I_0. This gives an upper bound on the number of domains, which is in $O(2^{|I_0|})$. In a state class, component B is a subset of trajectory places, and hence there are at most $2^{|P_T|}$ possible values for B. Last, for a K bounded trajectory net, the number of possible markings for control places is in $O(|P_C|^{K+1})$ [12]. The number of state classes is hence in $O(2^{|I_0|+|P_T|} \cdot |P_C|^{K+1})$.

6 Reachability, Coverability, Safety

An important property of the state class graph is that all solutions for domains that are reachable in $SC(\mathcal{N})$ are also reachable in $\mathcal{N}$. This immediately gives an algorithm to check coverability or reachability properties.

Theorem 3. *Given a state class (M_n, B_n, D_n) reachable from initial state class (M_0, B_0, D_0), and a solution $\mathcal{T}_n \in [\![D_n]\!]$, there exists a run in the original trajectory net that ends in configuration $(M_n, B_n, \mathcal{T}_n)$.*

Proof (Sketch). The proof is similar to the proof for soundness (Proposition 4), i.e. uses an induction on the length of paths in the state class graph. (See details in Appendix E).□

Theorem 4. *Reachability, boolean reachability, and boolean coverability are decidable in PSPACE for bounded nets*

Proof. These problems can be solved by a non-deterministic exploration of the state class graph. Let us first consider reachability of a given configuration $(M, B, \mathcal{T})$. Assume that a state class (M, B, D) such that $\mathcal{T} \in [\![D]\!]$ is reachable in $SC(\mathcal{N})$. Then, according to Theorem 3, there exists a run of $\mathcal{N}$ reaching $(M, B, \mathcal{T})$. Consider now the boolean reachability and coverability problems. Let $sc=(M, B, D)$ be a reachable state class. One can notice that the boolean marking M_C is identical for every configuration C matching sc. We hence denote this marking by M_{sc}, and compute it by assigning a token to a place $p_i \in P_T$ iff T_i, t_i

are variables used in D or if $p_i \in B$. Then, deciding reachability (resp. coverability) of a marking M consists in finding a state class sc such that $M_{sc} = M$ (resp. $M_{sc} \geq M$).

As shown in Section 5, the size of the state class graph is exponential w.r.t. the number of places and w.r.t. the value on constants appearing in intervals attached to trajectory places. Encoding a state class can be done in $\log(|SC(\mathcal{N})|)$ hence in polynomial space, and reachability questions can be addressed in nlogspace w.r.t. the size of the graph. At each step of an exploration reaching a particular state class $SC_i = (M_i, B_i, D_i)$, checking $M = M_i$, $B = B_i$, $M = M_{SC_i}$ or $M \leq M_{SC_i}$ can be done in linear time w.r.t the number of places, and checking $\mathcal{T} \in [\![D_i]\!]$ can be done in linear time w.r.t the number of inequalities in D by replacing every variable by its value in each inequality. As the number of inequalities in canonical domains is quadratic w.r.t. the number of trajectory places, checking $\mathcal{T} \in [\![D_i]\!]$ can be done in PTIME. Hence, reachability boolean reachability, and coverability can be checked in NPSPACE, which is equivalent to PSPACE by Savitch's theorem [23].□

Remark 1. Notice that a trajectory net without trajectory places is a Petri net (transitions can fire as soon as their preset is filled, after any delay). Hence, reachability and coverability in trajectory nets are at least as hard as reachability and coverability in 1-safe Petri nets. These problems are known to be PSPACE-Complete [7,11].

Corollary 1. *Reachability, boolean reachability, and boolean coverability for bounded nets are PSPACE-Complete.*

Proof. PSPACE membership is proved by Theorem 4. As highlighted in Remark 1, a reachability or a coverability problem for 1-safe nets is also a reachability/coverability problem for trajectory nets (with empty set of trajectory places). As these two problems are PSPACE-Complete [7,11], we get the result.□

6.1 Extending coverability to safety properties

Reachability is often a too precise question and one is usually interested in properties that address ranges of values for positions of objects. Let us get back to the case study introduced in Section 3, namely avoidance of simultaneous dangerous situations in a metro network. Metro networks can be easily represented by trajectory nets: trajectory places represent track portions between two stations, or a finer partition of a physical network into track portions called *blocks*, transitions symbolize departures, arrivals etc. Obviously, metro networks have very strict safety requirements that must be guaranteed by physical equipments such as signals and brakes. Safety issues also appear at the operational level. At any instant, evacuation of passengers should be feasible with the lowest risks, and for that reason, operators want to avoid situations where more than K trains are in tunnels or on bridges.

Addressing such safety properties is neither a reachability nor a coverability question: it requires that *at any instant*, all trajectories of trains avoid a *set of*

unsafe positions. Let $p \in P_T$ be a trajectory place, of length $H(p)$, and assume that the track portion represented by p contains a tunnel. We can easily define two values d_p^s and d_p^e defining respectively the position of the entry and exit of the tunnel in that track (for instance, in Figure 3, we have $d_p^s = 600$ and $d_p^e = 400$). Let $\mathcal{T}p = (T_p, t_p)$. The function gives us the initial duration T_p of a trip from a station to the next one, the remaining time t_p before the end of this trip, but we can also compute the position d_p of the considered train on the track. We have assumed that all objects moving in a trajectory net have a constant speed during the whole duration of a trajectory, sampled when the object enters a place. A consequence is that $\frac{H(p)}{T_p} = \frac{d_p}{t_p}$ at any instant. Hence $d_p = \frac{H(p) \cdot t_p}{T_p}$, and a train in place p is in a tunnel iff the following property is satisfied:

$$Tunnel(p) ::= d_p^e \leq \frac{H(p) \cdot t_p}{T_p} \leq d_p^s$$

Now assume that we want to avoid a situation where trains in a set of places $X = \{p_1, \ldots, p_k\} \subseteq P_T$ are in tunnels at the same instant. It means that we have to avoid any configuration that satisfies the property
$Unsafe(X) ::= \bigcup_{p_i \in X} Tunnel(p_i)$.

The domains computed in the state class graph $SC(\mathcal{N})$ are symbolic representations of configurations reached immediately after discrete moves. It can be the case that, after each discrete move, all trains are located before a tunnel in their respective track segment. Verifying safety of a train network does not amount to verifying that $[\![D \cup Unsafe(X)]\!] = \emptyset$ for sets of places X containing tunnels and for every reachable domain D. We need to consider how configurations depicted in D evolve when elapsing time, i.e., build symbolic representations of configurations reached an arbitrary duration after a discrete move. Hence, we introduce time closure $S_\downarrow = (M, B, D_\downarrow)$ of a state class $S = (M, B, D)$.

Definition 8 (Time Closure). *Let $S = (M, B, D)$ be any state class having a set of places with progressing trajectories $P_{pr} \subseteq P_T$. We introduce variables δ (to represent the timed move) and t'_i for all $i \in P_{pr}$ (to represent time remaining in trajectories after a timed move of duration δ). The* time closure *of S is a 3-tuple $S_\downarrow = (M, B, D_\downarrow)$ with $D_\downarrow$ defined as:*

Case I: No transition is firable from the given marking M and set of blocked trajectories B, and hence timed moves are allowed by the semantics of trajectory nets. We have $D_\downarrow = D \cup \{0 \leq \delta \leq t_i \mid i \in P_{pr}\} \cup \{t'_i = t_i - \delta \mid i \in P_{pr}\}$

Case II: There exists a firable transition for the given marking M and set of blocked trajectories B, and hence timed moves are not allowed. We hence have $D_\downarrow = D \cup \{\delta = 0\} \cup \{t'_i = t_i \mid i \in P_{pr}\}$.

The time closure of a state class $S = (M, B, D)$ is a symbolic representation of possible configurations reachable after timed moves of arbitrary duration δ, including the configurations in domain D (i.e., when $\delta = 0$). As explained above, a property of interest for metro networks is that no more than K trains are in a

tunnel at any given instant. We denote by $Unsafe'(X)$ the set of inequalities obtained by replacing, for every place $p \in X$ variable t_p by t'_p in $Unsafe(X)$. A state class $S = (M, B, D)$ is *safe* for places $p_1, \dots p_K$ iff $[\![D_\downarrow \cup Unsafe'(p_1, \dots p_K)]\!] = \emptyset$. Verifying that a state class is safe for every subset of places of size K amounts to asking that $[\![D_\downarrow \cup Unsafe'(X)]\!] = \emptyset$ for every subset $X \subseteq P_T$ of places of size K containing tunnels. Notice that the set of all X's can be enumerated in $\log(|P_T|)$ space.

Remark 2. Non-emptiness of $D_\downarrow \cap Unsafe'(X)$ implies existence of a configuration violating the safety property, because all configurations in a state class D are reachable, and hence all configurations in $D_\downarrow$ too. Hence, checking safety for all state classes of $SC(\mathcal{N})$ guarantees that the trajectory net does not violate the safety property. This gives us a method to check safety of a metro network modeled with a trajectory net using its state class graph.

Let $X = \{p_1, \dots, p_k\}$. The constraint $Unsafe'(X)$ can be rewritten as $\left\{ d^e_{p_i} \leq \frac{H(p_i) \cdot t'_i}{T_i} \leq d^s_{p_i} \mid p_i \in X \right\}$ and as every T_i is a positive value, simplified to get $\left\{ d^e_{p_i} \cdot T_i \leq H(p_i) \cdot t'_i \leq d^e_{p_i} \cdot T_i \mid p_i \in X \right\}$. One can immediately observe that this set contains only linear inequalities of dimension 2 involving T_i and t'_i. Let us now consider $D_\downarrow$, defined for a set of places P of progressing trajectories. It obtained by replacing t_i by $t'_i + \delta$. It is hence a set of constraints of the form:

$$\begin{aligned}
\alpha^1_i \leq T_i \leq \beta^1_i &\text{ for all } p_i \in P \\
\alpha^2_i \leq t'_i + \delta \leq \beta^2_i &\text{ for all } p_i \in P \\
t'_i - t'_j \leq \gamma^3_{ij} &\text{ for all } p_i, p_j \in P \text{ with } i \neq j \\
\alpha^4_i \leq T_i - t'_i + \delta \leq \beta^4_i &\text{ for all } p_i \in P \\
\alpha^5_{ij} \leq T_i - t'_i + t'_j \leq \beta^5_{ij} &\text{ for all } p_i, p_j \in P \text{ with } i \neq j \\
-T_i + t'_i + T_j - t'_j \leq \gamma^6_{ij} &\text{ for all } p_i, p_j \in P \text{ with } i \neq j
\end{aligned}$$

Remark 3. Checking emptiness of $D_\downarrow \uplus \left\{ d^e_{p_i} \cdot T_i \leq H(p_i) \cdot t'_i \leq d^s_{p_i} \cdot T_i \mid p_i \in X \right\}$ can be done by elimination of variables one after another, and stopping as soon as an inequality is unsatisfiable, or when all variables are eliminated. We can use a variable change as in Proposition 2, and get an equivalent system of dimension 2. Hence, checking satisfiability of $D_\downarrow \uplus Unsafe(p_1, \dots, p_k)$ can be done in PTIME.

Proposition 6. *Checking a safety property for bounded trajectory nets is PSPACE-complete.*

Proof. $\mathcal{N}$ violates a safety property of the form "no more than K trains in a tunnel" iff there exists a reachable configuration $C = (M, B, \mathcal{T})$ such that $\mathcal{T} \models Unsafe(X)$ for some subset of places $X = \{p_1, \dots p_K\}$ with tunnels. According to remark 2, this holds only if there exists a reachable state class $S = (M, B, D)$ such that $[\![D_\downarrow \cup Unsafe'(X)]\!] \neq \emptyset$

We know from the decision procedures for reachability and coverability that exploration of all state classes can be done in PSPACE. Then, for each state class

$S = (M, B, D)$ reached, we need to compute $D_\downarrow$ enumerate all subsets X of K places containing tunnels and with a progressing trajectory, and check emptiness of $D_\downarrow \cup Unsafe(p_1, \ldots, p_K)$. Enumeration of subsets of places of size K can be done in $\log(P_T)$ space. Then, following remark 3, we know that (un)satisfiability of $D_\downarrow \cup Unsafe(p_1, \ldots p_k)$ can be verified in PSPACE. The hardness comes from a reduction of the coverability problems for 1-safe nets: if C is a set of places to cover in a net $\mathcal{N}$, on can design a trajectory net $\mathcal{N}_T$ with one additional trajectory places $p_{T,c}$ per place in C and such that ${}^\bullet(p_{T,c}) = {}^\bullet(c)$, $(p_{T,c})^\bullet = (c)^\bullet$ and set zones to avoid as the whole length of these places. Reaching a configuration with objects in $X = \{p_{T,c} \mid c \in C\}$ in $\mathcal{N}_T$ amounts to covering C in $\mathcal{N}$.□

7 Conclusion

We have considered an extension of Petri nets enhanced with time and linear functions depicting trajectories of moving objects. Most problems for this model are undecidable in general. However, as soon as the control part is bounded, the behaviour of the model can be abstracted to a finite state class graph, and coverability, reachability, and safety properties addressing distance issues can be decided in PSPACE. Finiteness of the state class graph of trajectory nets comes from bounds on the values of variables, and from the particular structure of domains, that are conjunctions of linear inequalities with at most 4 variables, and coefficients in $\{1, -1\}$. This structure is preserved by projection, and hence by the successor relation among state classes. Preliminary work shows that domains for trajectory nets are a form of regular polyhedra, that can be encoded by Totally Unimodular Matrices (TUM). This needs to be formally proved, but it would explain why our domains have interesting closure and algorithmic properties. An interesting research direction is to consider extensions of trajectory nets that preserve this nice and efficient domain structure.

As future work, several extensions of the model can be considered. We have defined a restricted model, mainly tailored to represent metro networks. The first restriction is that transitions have at most one trajectory place in their preset and in their postset. This last restriction was used to simplify notations and reading, and can be easily relaxed: one can imagine firing rules consuming several blocked trajectories, and similarly producing several new trajectories in the postset of a transition. The techniques shown in this paper easily adapt. It should also be possible to consider trajectories in 2D or 3D spaces.

The second restriction imposes that trajectories are simple linear functions, i.e. the speed of an object is sampled once, and remains constant until a trajectory gets blocked. One could easily improve trajectories using piecewise linear function depicting behavior of objects moving at varying speeds (this is the model proposed in [16], Chapter 5). This would be useful to describe acceleration and braking phases. We conjecture that this extension still allows to work with TUMs, as long as trajectory places contain at most one trajectory.

More involved improvements would be to consider trajectories specified by polynomials, or to allow more than one trajectory per place. The single trajectory restriction makes sense when modeling metro networks for instance, as

many cities implement a fixed block traffic management policy, where tracks are partitioned in exclusive blocks that must be occupied by a single train at any instant. However, this type of management is progressively replaced by moving block policies, that allow several trains in a track segment provided they maintain safety distances. Relaxing the single trajectory restriction is also needed to model other situations such as road traffic. First experiments seem to show that a definition of domains for these two extensions require polynomial inequalities, i.e. expressions of the form $P(X) \leq c$, where $P(X)$ is a multivariate polynomial. It is known that such domains are closed under projection [24]. However, variable elimination has a doubly exponential complexity [8]. Further, we conjecture that finiteness of domains does not hold any more in this new setting.

References

1. P.A. Abdulla and A. Nylén. Timed Petri nets and BQOs. In *ICATPN*, LNCS 2075, p.53-70, 2001.
2. R. Alur, C. Courcoubetis, N. Halbwachs, T.A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *TCS*, 138(1):3–34, 1995.
3. R. Alur and D.L. Dill. A theory of timed automata. *TCS*, 126(2):183–235, 1994.
4. Rajeev Alur, Costas Courcoubetis, Nicolas Halbwachs, Thomas A. Henzinger, Pei-Hsin Ho, Xavier Nicollin, Alfredo Olivero, Joseph Sifakis, and Sergio Yovine. The algorithmic analysis of hybrid systems. *Theor. Comput. Sci.*, 138(1):3–34, 1995.
5. B. Berthomieu and M. Diaz. Modeling and verification of time dependent systems using time Petri nets. *IEEE Trans. Software Eng.*, 17(3):259–273, 1991.
6. V. Chandru. Variable elimination in linear constraints. *Comput. J.*, 36(5):463–472, 1993.
7. A. Cheng, J. Esparza, and J. Palsberg. Complexity results for 1-safe nets. *TCS*, 147(1&2):117–136, 1995.
8. G. E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decompostion. In *Automata Theory and Formal Languages*, pages 134–183, 1975.
9. G. Dantzig and B. C. Eaves. Fourier-Motzkin Elimination and Its Dual. *J. Comb. Theory, Ser. A*, 14(3):288–297, 1973.
10. D.L. Dill. Timing assumptions and verification of finite-state concurrent systems. In *Automatic Verification Methods for Finite State Systems*, volume 407 of *LNCS*, pages 197–212, 1989.
11. J. Esparza. Decidability and complexity of Petri net problems - an introduction. In *Proc. of Petri Nets*, volume 1491 of *LNCS*, pages 374–428, 1998.
12. J. Esparza. Petri nets lecture notes. Technical report, 2019.
13. D. de Frutos-Escrig, V.V. Ruiz, and O. Marroquín Alonso. Decidability of properties of timed-arc Petri nets. In M. Nielsen and D. Simpson, editors, *Application and Theory of Petri Nets 2000, 21st International Conference, ICATPN 2000, Aarhus, Denmark, June 26-30, 2000, Proceeding*, volume 1825 of *Lecture Notes in Computer Science*, pages 187–206. Springer, 2000.
14. L. Hélouët and P. Agrawal. Waiting nets. In *Proc. of PETRI NETS 2022*, volume 13288 of *LNCS*, pages 67–89, 2022.
15. N.D. Jones, L. H. Landweber, and Y. E. Lien. Complexity of some problems in Petri nets. *Theor. Comput. Sci.*, 4(3):277–299, 1977.

16. Karim Kecir. *Performance Evaluation of Urban Rail Traffic Management Techniques. (Évaluation de Performances pour les Techniques de Régulation du Trafic Ferroviaire Urbain)*. PhD thesis, University of Rennes 1, France, 2019.
17. D. Lime and O.H. Roux. Model checking of time Petri nets using the state class timed automaton. *Discret. Event Dyn. Syst.*, 16(2):179–205, 2006.
18. E.W. Mayr. An algorithm for the general Petri net reachability problem. In *Proceedings of the 13th Annual ACM Symposium on Theory of Computing, May 11-13, 1981, Milwaukee, Wisconsin, USA*, pages 238–246. ACM, 1981.
19. P. M. Merlin. *A Study of the Recoverability of Computing Systems*. PhD thesis, University of California, Irvine, CA, USA, 1974.
20. C. Rackoff. The covering and boundedness problems for vector addition systems. *Theor. Comput. Sci.*, 6:223–231, 1978.
21. P.-A. Reynier and A. Sangnier. Weak time Petri nets strike back! In *Proc. of CONCUR 2009*, volume 5710 of *LNCS*, pages 557–571, 2009.
22. V. V. Ruiz, F. C. Gomez, and D. de Frutos-Escrig. On non-decidability of reachability for timed-arc Petri nets. In *PNPM*, pages 188–. IEEE Computer Society, 1999.
23. Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970.
24. A. Tarski. A decision method for elementary algebra and geometry. Technical report, RAND Corporation, 1957.

A Fourier-Motzkin elimination

Fourier-Motzkin Elimination [9] is a method to eliminate a set of variables $V \subseteq X$ from a system of linear inequalities over X. Elimination produces another system of linear inequalities over $X \setminus V$, such that both systems have the same solutions over the remaining variables. Elimination can be done by removing one variable from V after another.

Let $X = \{x_1, \ldots, x_r\}$ be a set of variables, and w.l.o.g., let us assume that x_r is the variable to eliminate in m inequalities. All inequalities are of the form

$$c_{i,1} \cdot x_1 + c_{i,2} \cdot x_2 + \cdots + c_{i,r} \cdot x_r \leq d_i$$

where c_j's and d_i are rational values, or equivalently $c_{i,r} \cdot x_r \leq d_i - (c_{i,1} \cdot x_1 + c_{i,2} \cdot x_2 + \cdots + c_{i,r-1} \cdot x_{r-1})$

If $c_{i,r}$ is a negative coefficient, the inequality can be rewritten as $x_r \geq b_i - (a_{i,1} \cdot x_1 + a_{i,2} \cdot x_2 + \ldots a_{i,r-1} \cdot x_{r-1})$, and if $c_{i,r}$ is positive, the inequality rewrites as $x_r \leq b_i - (a_{i,1} \cdot x_1 + a_{i,2} \cdot x_2 + \ldots a_{i,r-1} \cdot x_{r-1})$, where $b_i = \frac{d_i}{c_{i,r}}$ and $a_{i,k} = \frac{c_{i,k}}{c_{i,r}}$ for every $k \in \{1, \cdots, r-1\}$.

We can partition our set of inequalities as follows.

- inequalities of the form $x_r \geq b_i - \sum_{k=1}^{r-1} a_{i,k} \cdot x_k$; denote these by $x_r \geq A_j(x_1, \ldots, x_{r-1})$ (or simply $x_r \geq A_j$ for short), for j ranging from 1 to n_A where n_A is the number of such inequalities;
- inequalities of the form $x_r \leq b_i - \sum_{k=1}^{r-1} a_{i,k} \cdot x_k$; denote these by $x_r \leq B_j(x_1, \ldots, x_{r-1})$ (or simply $x_r \leq B_j$ for short), for j ranging from 1 to n_B where n_B is the number of such inequalities;

– inequalities $\phi_1, \dots \phi_{m-(n_A+n_B)}$ in which x_r plays no role.

The original system is thus equivalent to:
$\max(A_1, \dots, A_{n_A}) \leq x_r \leq \min(B_1, \dots, B_{n_B}) \wedge \bigwedge_{i \in 1..m-(n_A+n_B)} \phi_i$.

One can find a value for x_r in a system of the form $a \leq x \leq b$ iff $a \leq b$. Hence, the above formula is equivalent to:
$\max(A_1, \dots, A_{n_A}) \leq \min(B_1, \dots, B_{n_B}) \wedge \bigwedge_{i \in 1..m-(n_A+n_B)} \phi_i$

Now, this inequality can be rewritten as system of $n_A \times n_B + m - (n_A + n_B)$ inequalities $\{A_i \leq B_j \mid i \in 1..n_A, j \in 1..n_B\} \cup \{\phi_i \mid i \in 1..m - (n_A + n_B)\}$, that does not contain x_r and is satisfiable iff the original system is satisfiable.

Remark 4. The Fourier-Motzkin elimination preserves finiteness and satisfiability of a system of constraints. In general, the number of inequalities can grow in a quadratic way at each variable elimination. It the case of trajectory nets, where domains are in canonical form, they always contain less than $2 \cdot |T|^2 + 2 \cdot |T|$ inequalities, and then elimination produces a system of at most $2 \cdot |T|^2 + 2 \cdot |T|$ inequalities once useless inequalities have been removed.

B Undecidability

Theorem 1 Reachability, boolean reachability and coverability are undecidable for trajectory nets

Proof. We can simulate the behavior of an unbounded two counters machine with an unbounded trajectory net.

A two-counter machine is specified with two counters C_1, C_2 that remember positive integral values, and a list of instructions $Inst_1, Inst_2, \dots Inst_m$. These instructions are of the form:

– $Inst_i : Inc(C_n)$, which effect is to increment counter C_n, and then move to instruction $i + 1$
– $Inst_i$: **If** $C_n > 0$ **then** $Dec(C_n)$ **else** $Inst_j$, that tests the value of counter C_n, decrements it an moves to the next instruction if $C_n > 0$, or moves to instruction $Inst_j$ otherwise.
– $Halt$, that stops the computation of the counter machine.

A configuration of a counter machine is a triple (i, c_1, c_2) representing the current instruction to execute, and the values of counters c_1, c_2. It is well known that counter machines are sufficient to encode Turing Machines, and hence that the question of whether a machine starting at instruction $Inst_1$ with counter values $c_1 = 0, c_2 = 0$ eventually halts is undecidable.

Consider an arbitrary counter machine $\mathcal{M}$ with counters C_1, C_2, and a list of instructions $Inst_1, Inst_2, \dots Inst_m$. We build a trajectory net $\mathcal{N}_\mathcal{M} = (P, T, I)$ that contains:

- Two control places p_{c1}, p_{c2} (one per counter), and one trajectory place pt_{inst_i} with $I(pt_{inst_i}) = [2, 2]$ for each instruction $Inst_i$. One control place p_i per instruction $Inst_i$, and an additional place p'_i if instruction i is a decrement instruction.
- one transition $\sigma_{i,inc(n)}$ for each instruction of the form $Inst_i = Inc(C_n)$, such that ${}^\bullet(\sigma_{i,inc(n)}) = \{pt_{inst_i}, p_i\}$, $(\sigma_{i,inc(n)})^\bullet = \{pt_{inst_{i+1}}, p_{c_n}, p_{i+1}\}$.
- three transitions $\sigma_{i,Z}, \sigma_{i,NZ}, \bar{\sigma}_{i,NZ}$ for each decrement instruction of the form $Inst_i$: **If** $C_n > 0$ **then** $Dec(C_n)$ **else** $Inst_j$ with:
  - ${}^\bullet(\sigma_{i,Z}) = \{pt_{inst_i}, p_i\}$, and $(\sigma_{i,Z})^\bullet = \{pt_{inst_j}, p_j\}$,
  - ${}^\bullet(\sigma_{i,NZ}) = \{pt_{inst_i}, p_i, p_{c_n}\}$, and $(\sigma_{i,NZ})^\bullet = \{p'_i\}$,
  - ${}^\bullet(\bar{\sigma}_{i,NZ}) = \{pt_{inst\,i}, p'_i\}$, and $(\bar{\sigma}_{i,NZ})^\bullet = \{pt_{inst\,i+1}, p_{i+1}\}$.

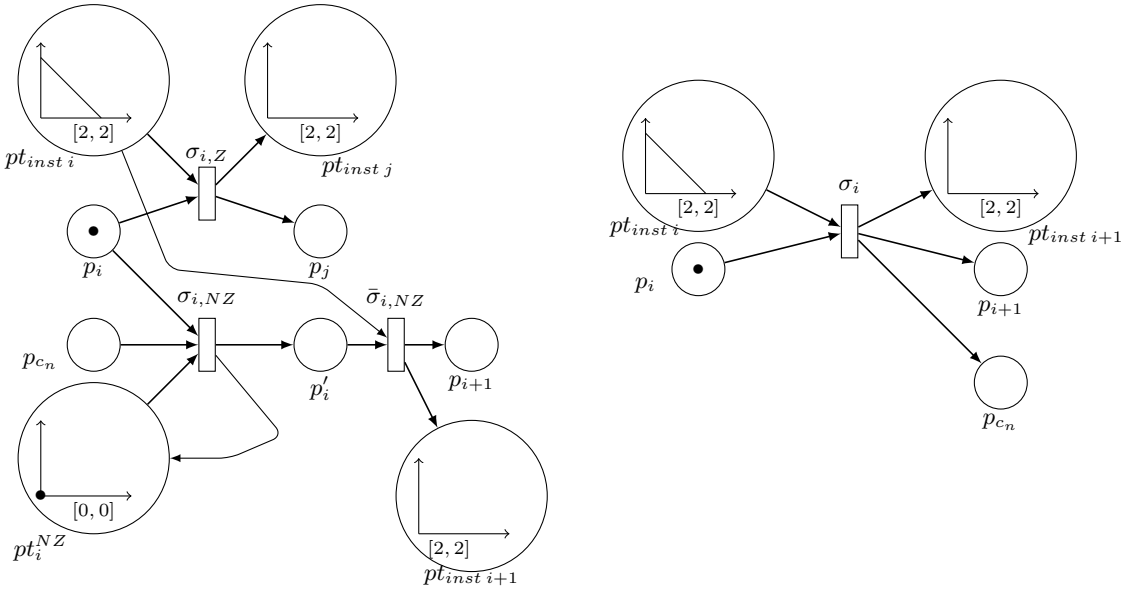

Fig. 5. Encoding instructions of a counter Machine with a trajectory net.

The trajectory net assembled this way encodes steps of the counter machine as follows: At every instant, to simulate execution of instruction i, all tokens are located in places p_i, p'_i (for a single i) and the value of counter C_n is stored in place p_{c_n}. A configuration $C = (M, B, \mathcal{T})$ with p_i marked and $\mathcal{T}(pt_{inst\,i}$ where $Inst_i$ is an increment of counter C_n encodes a state of the machine reaching instruction i. From this configuration, time can elapse up to the maximal amount of time allowed by $\mathcal{T}(p_{inst\,i})$, then trajectory tr_i is blocked, and transition σ_i fires. As a consequence, place $pt_{inst\,i+1}$ is filled with a new trajectory, $M(p_{c_n})$ is incremented, and place p_{i+1} receives a token. The new contents of $pt_{inst\,i+1}$, p_{i+1}, and p_{c_n} simulate the next configuration of the counter machine after execution

of instruction $Inst_i$. Similarly, let us consider a configuration $C = (M, B, \mathcal{T})$ with p_i marked and $\mathcal{T}(pt_{inst\,i})$ and where $Inst_i$ is a decrement instruction of counter C_n. From this configuration, two distinct scenarios can occur. If p_{c_n} is empty, then σ_i^{NZ} cannot fire, and hence the maximal amount of time allowed by $\mathcal{T}(p_{inst\,i})$ elapses , then trajectory tr_i is blocked, and transition σ_i^{Z} fires. As a consequence, place $pt_{inst\,j}$ is filled with a new trajectory, and place p_j receives a token. The new contents of $pt_{inst\,j}$, p_j, and p_{c_n} simulate the next configuration of the counter machine after execution of instruction $Inst_i$ when counter C_n is equal to 0. Conversely, assume that p_{c_n} holds m tokens. Then the firing of σ_i^{NZ} is urgent, and hence the marking of p_{c_n} is decremented, and a token is moves from p_i to p'_i. A new trajectory of duration 0 is put in place pt_i^{NZ} for use at the next occurrence of instruction $Inst_i$. The last step of simulation of the decrement instruction is to block and consume the contents of $p_{inst\,i}$ via transition $\bar{\sigma}_i^{NZ}$, and produce a new trajectory in place $p_{inst\,i+1}$ and a token in place p_{i+1}. One can easily see that for every sequence of configurations of a counter machine $\mathcal{M}$, there exists a run of $\mathcal{N}_{\mathcal{M}}$ such that the sequences of indexes of marked instruction places and the markings of p_{c_1}, p_{c_2} is exactly the run $\mathcal{M}$. Conversely, for every run of $\mathcal{N}_{\mathcal{M}}$, the sequences of indexes of marked instruction places and the markings of p_{c_1}, p_{c_2} coincide with a run of machine $\mathcal{M}$. As a consequence, if $Inst_m$ is a halting instruction, then one cannot decide in general whether some marking with $M(p_m) \geq 1$ is reachable. Coverability is hence undecidable. The result easily extends to reachability questions, as one can add to $\mathcal{N}_{\mathcal{M}}$ a gadget that consumes the contents of counter places and of $pt_{inst\,m}$, and ask whether marking M_{halt} such that $M_{halt}(p_m) = 1$ and all other places are empty is reachable.

C Canonical Domains

Proposition 1 The canonical form of a domain D is unique and preserves $[\![D]\!]$.

Proof. A domain D is a set of inequalities of the form $a_i \leq expr_i$, and $expr_j \leq b_j$. A solution $\mu_S \in [\![D]\!]$ is a map that associates a real value with every variable t_i (resp. T_i). Let $\mu(expr_i)$ be the value obtained by replacing every variable t_i by $\mu_S(t_i)$ and T_i by $\mu_S(T_i)$ in $expr_i$. Then the values $a_i^* = \min_{\mu_S \in [\![D]\!]} \mu_S(expr_i)$ and $b_j^* = \max_{\mu_S \in [\![D]\!]} \mu_S(expr_j)$ are unique. Hence D^* is uniquely defined.

Let us now show that the canonical form preserves the set of solutions $[\![D]\!]$. We can remark that an expression of the form $a_i \leq expr_i$ can be equivalently rewritten as $-expr_i \leq -a_i$. We can hence safely assume that domains are defined by sets of inequalities of the form $\{expr_j \leq b_j\}$. Let us now show that replacing an inequality $expr \leq \alpha$ by $expr \leq \alpha^*$ in D does not affect the solution set. Let D' denote the new system of inequalities obtained after this replacement. Suppose that the set of solutions $[\![D']\!]$ differs from $[\![D]\!]$. We can have two cases:

- There exists a feasible solution $\mu_S \in [\![D']\!]$ but $\mu_S \notin [\![D]\!]$. Let $v = \mu_S(expr)$. Since μ_S satisfies D', we have $v \leq \alpha^* \leq \alpha$, and hence μ_S also satisfies

$expr \leq \alpha$, and hence we also have $\mu_S \in [\![D]\!]$ (all other inequalities are unmodified, and hence cannot be violated in D). Contradiction.

– There exists a feasible solution $\mu_S \in [\![D]\!]$ but $\mu_S \notin [\![D']\!]$. Again, the only inequality which can be violated is $expr \leq \alpha^*$. Let $v = \mu_S(expr)$. As μ_S is a solution of D, we necessarily have $\alpha^* < v \leq \alpha$. However, we have chosen α^* to be optimal, i.e. $\alpha^* = \max_{\mu_S \in [\![D]\!]} \mu_S(expr)$. Contradiction.

Hence, we can say $[\![D']\!] = [\![D]\!]$. As D^* can be built by successively replacing every a_i by a_i^* and every b_j by b_j^*, we can say that $[\![D]\!] = [\![D^*]\!]$. □

D Closure of Domains

Proposition 3 Let D be a domain of a trajectory net $\mathcal{N}$, and let D' be a system of linear inequalities that is a successor of D via construction of $SuccB(D, p)$ or $SuccF(D, t)$. Then D' is a domain of $\mathcal{N}$.

Proof. The initial domain conforms to Definition 3. When a new trajectory is added in a place, then the variables T_i and t_i meet the constraint $\alpha_i^s \leq T_i \leq$ and $\alpha_i^s \leq T_i \leq$. So in the original domain reached before firing a transition has constants that are linear combinations of α_i^s and β_i^s, then so has the newly computed domin in $SuccF(D, t_i)$ for every transition t_i.

It now remains to show that the elimination of variables preserves the property. Assume that we want to eliminate variable t_i in a domain $D \cup \{t_i \leq t_j \mid j \neq i\}$, and obtain a new domain over variables $\{T_j, t'_j\}$, where t'_j is the remaining travel time for a trajectory. We have $t'_j = t_j - t_i$ so we do a variabke change of the form $t_j - > t'_j + t_i$ in domain D before starting elimination

D, with the constraint that $t_i \leq t_j$ can be written as

$$
\begin{array}{rll}
 & \alpha_i^2 \leq t_i \leq \beta_i^2 & \\
 & 0 \leq t_j - t_i & \text{for all } j \\
 & t_i - t_j \leq \gamma_{ij}^3 & \text{for all } i \neq j \\
 & t_j - t_i \leq \gamma_{ji}^3 & \text{for all } i \neq j \\
D_{\setminus\{t_i\}} \cup & \alpha_i^4 \leq T_i - t_i \leq \beta_i^4 & \\
 & \alpha_{ij}^5 \leq T_i - t_i + t_j \leq \beta_{ij}^5 & \text{for all } i \neq j \\
 & \alpha_{ji}^5 \leq T_j - t_j + t_i \leq \beta_{ji}^5 & \text{for all } i \neq j \\
 & -T_i + t_i + T_j - t_j \leq \gamma_{ij}^6 & \text{for all } i \neq j \\
 & -T_j + t_j + T_i - t_i \leq \gamma_{ji}^6 & \text{for all } i \neq j
\end{array}
$$

where $D_{\setminus\{t_i\}}$ is the set of all inequalities of D that do not contain t_i. With the variable change, we get

$$
\begin{array}{ll}
 & \alpha^2_i \leq t_i \leq \beta^2_i & \\
 & 0 \leq (t'_j + t_i) - t_i & \text{for all } j \\
 & t_i - (t'_j + t_i) \leq \gamma^3_{ij} & \text{for all } i \neq j \\
 & t'_j + t_i - t_i \leq \gamma^3_{ji} & \text{for all } i \neq j \\
D' \cup & \alpha^4_i \leq T_i - t_i \leq \beta^4_i & \\
 & \alpha^5_{ij} \leq T_i - t_i + t'_j + t_i \leq \beta^5_{ij} & \text{for all } i \neq j \\
 & \alpha^5_{ji} \leq T_j - (t'_j + t_i) + t_i \leq \beta^5_{ji} & \text{for all } i \neq j \\
 & -T_i + t_i + T_j - (t'_j + t_i) \leq \gamma^6_{ij} & \text{for all } i \neq j \\
 & -T_j + t'_j + t_i + T_i - t_i \leq \gamma^6_{ji} & \text{for all } i \neq j
\end{array}
$$

where D' is the domain obtained by replacing every t_j by $t'_j + t_i$

This system can be rewritten as

$$
\begin{array}{ll}
 & \alpha^2_i \leq t_i \leq \beta^2_i & \\
 & 0 \leq t'_j & \text{for all } j \neq i \\
 & -t'_j \leq \gamma^3_{ij} & \text{for all } i \neq j \\
 & t'_j \leq \gamma^3_{ji} & \text{for all } i \neq j \\
D' \cup & \alpha^4_i \leq T_i - t_i \leq \beta^4_i & \\
 & \alpha^5_{ij} \leq T_i + t'_j \leq \beta^5_{ij} & \text{for all } i \neq j \\
 & \alpha^5_{ji} \leq T_j - t'_j \leq \beta^5_{ji} & \text{for all } i \neq j \\
 & -T_i + T_j - t'_j \leq \gamma^6_{ij} & \text{for all } i \neq j \\
 & -T_j + t'_j + T_i \leq \gamma^6_{ji} & \text{for all } i \neq j
\end{array}
$$

We hence obtain a domain of the form

$$
\begin{array}{ll}
\alpha^2_k \leq t'_k + t_i \leq \beta^2_k & \text{for all } k \neq i \\
t'_k - t'_j \leq \gamma^3_{kj} & \text{for all } k \neq j \neq i \\
\alpha^4_k \leq T_k - t'_k - t_i \leq \beta^4_k & \text{for all } k \neq i \\
\alpha^5_{kj} \leq T_k - t'_k + t'_j \leq \beta^5_{kj} & \text{for all } k \neq j \neq i \\
-T_k + t'_k + T_j - t'_j \leq \gamma^6_{kj} & \text{for all } k \neq j \neq i
\end{array}
$$

$$
\begin{array}{ll}
\alpha^2_i \leq t_i \leq \beta^2_i & \\
0 \leq t'_j & \text{for all } j \neq i \\
-t'_j \leq \gamma^3_{ij} & \text{for all } i \neq j \\
t'_j \leq \gamma^3_{ji} & \text{for all } i \neq j \\
\alpha^4_i \leq T_i - t_i \leq \beta^4_i & \\
\alpha^5_{ij} \leq T_i + t'_j \leq \beta^5_{ij} & \text{for all } i \neq j \\
\alpha^5_{ji} \leq T_j - t'_j \leq \beta^5_{ji} & \text{for all } i \neq j \\
-T_i + T_j - t'_j \leq \gamma^6_{ij} & \text{for all } i \neq j \\
-T_j + t'_j + T_i \leq \gamma^6_{ji} & \text{for all } i \neq j
\end{array}
$$

We can hence isolate inequalitities that do not refer to T_i nor t_i in a set D'', Identidy a set D_{T_i} of inequalities that refer to T_i, but not to t_i, and a set D_{t_i} that contain all lines with t_i. We have:

$$D'' = D' \cup \begin{cases} t'_k - t'_j \leq \gamma^3_{kj} & \text{for all } k \neq j \neq i \\ \alpha^5_{kj} \leq T_k - t'_k + t'_j \leq \beta^5_{kj} & \text{for all } k \neq j \neq i \\ -T_k + t'_k + T_j - t'_j \leq \gamma^6_{kj} & \text{for all } k \neq j \neq i \\ \\ 0 \leq t'_j & \text{for all } j \neq i \\ -t'_j \leq \gamma^3_{ij} & \text{for all } i \neq j \\ t'_j \leq \gamma^3_{ji} & \text{for all } i \neq j \\ \alpha^5_{ji} \leq T_j - t'_j \leq \beta^5_{ji} & \text{for all } i \neq j \end{cases}$$

then

$$D_{T_i} = \begin{cases} \alpha^5_{ij} \leq T_i + t'_j \leq \beta^5_{ij} & \text{for all } i \neq j \\ -T_i + T_j - t'_j \leq \gamma^6_{ij} & \text{for all } i \neq j \\ -T_j + t'_j + T_i \leq \gamma^6_{ji} & \text{for all } i \neq j \end{cases}$$

and

$$D_{t_i} = \begin{cases} \alpha^2_k \leq t'_k + t_i \leq \beta^2_k & \text{for all } k \neq i \\ \alpha^4_k \leq T_k - t'_k - t_i \leq \beta^4_k & \text{for all } k \neq i \\ \alpha^2_i \leq t_i \leq \beta^2_i \\ \alpha^4_i \leq T_i - t_i \leq \beta^4_i \end{cases}$$

Eliminating t_i then consists in eliminating t_i from D_{t_i}

$$D_{t_i} = \begin{cases} \alpha^2_k - t'_k \leq t_i \leq \beta^2_k - t'_k & \text{for all } k \neq i \\ \alpha^4_k - T_k + t'_k \leq -t_i \leq \beta^4_k - T_k + t'_k & \text{for all } k \neq i \\ \alpha^2_i \leq t_i \leq \beta^2_i \\ \alpha^4_i - T_i \leq -t_i \leq \beta^4_i - T_i \end{cases}$$

which can be rewritten as

$$D_{t_i} = \begin{cases} \alpha^2_k - t'_k \leq t_i \leq \beta^2_k - t'_k & \text{for all } k \neq i \\ T_k - t'_k - \beta^4_k \leq t_i \leq T_k - t'_k - \alpha^4_k & \text{for all } k \neq i \\ \alpha^2_i \leq t_i \leq \beta^2_i \\ T_i - \beta^4_i \leq t_i \leq T_i - \alpha^4_i \end{cases}$$

Eliminating t_i, we get:

$$
D_{\bar{t_i}} = \begin{array}{ll}
\alpha_k^2 - t'_k \leq \beta_k^2 - t'_k & \text{for all } k \neq i \\
\alpha_k^2 - t'_k \leq T_k - t'_k - \alpha_k^4 & \text{for all } k \neq i \\
\alpha_k^2 - t'_k \leq \beta_i^2 & \\
\alpha_k^2 - t'_k \leq T_i - \alpha_i^4 & \\
T_k - t'_k - \beta_k^4 \leq \beta_k^2 - t'_k & \text{for all } k \neq i \\
T_k - t'_k - \beta_k^4 \leq T_k - t'_k - \alpha_k^4 & \text{for all } k \neq i \\
T_k - t'_k - \beta_k^4 \leq \beta_i^2 & \\
T_k - t'_k - \beta_k^4 \leq T_i - \alpha_i^4 & \\
\alpha_i^2 \leq \beta_k^2 - t'_k & \text{for all } k \neq i \\
\alpha_i^2 \leq T_k - t'_k - \alpha_k^4 & \text{for all } k \neq i \\
\alpha_i^2 \leq \beta_i^2 & \\
\alpha_i^2 \leq T_i - \alpha_i^4 & \\
T_i - \beta_i^4 \leq \beta_k^2 - t'_k & \text{for all } k \neq i \\
T_i - \beta_i^4 \leq T_k - t'_k - \alpha_k^4 & \text{for all } k \neq i \\
T_i - \beta_i^4 \leq \beta_i^2 & \\
T_i - \beta_i^4 \leq T_i - \alpha_i^4 &
\end{array}
$$

One can notice that in $D_{\bar{t_i}}$, all inequalities that do not contain T_i are either tautologies, or inequalities of the form given in Defn. 3

Let $D''' = D'' \cup X_{\bar{T_i}} \cup D_{T_i} \cup X_{T_i}$, where $X_{\bar{T_i}}$ is the set of inequalities that do not contain T_i in $D_{\bar{t_i}}$ and X_{T_i} is the set of inequalities that do contain T_i in $D_{\bar{t_i}}$. Then, eliminating T_i from D''' boils down to eliminating T_i from $D_{T_i} \cup X_{T_i}$

$$
D_{T_i} \cup X_{T_i} = \begin{cases}
\alpha_{ij}^5 \leq T_i + t'_j \leq \beta_{ij}^5 & \text{for all } i \neq j \\
-T_i + T_j - t'_j \leq \gamma_{ij}^6 & \text{for all } i \neq j \\
-T_j + t'_j + T_i \leq \gamma_{ji}^6 & \text{for all } i \neq j \\
\alpha_k^2 - t'_k \leq T_i - \alpha_i^4 & \\
T_k - t'_k - \beta_k^4 \leq T_i - \alpha_i^4 & \\
\alpha_i^2 \leq T_i - \alpha_i^4 & \\
T_i - \beta_i^4 \leq \beta_k^2 - t'_k & \text{for all } k \neq i \\
T_i - \beta_i^4 \leq T_k - t'_k - \alpha_k^4 & \text{for all } k \neq i \\
T_i - \beta_i^4 \leq \beta_i^2 &
\end{cases}
$$

This can be rewritten as:

$$
D_{T_i} \cup X_{T_i} = \begin{cases}
\alpha_{ij}^5 - t'_j \leq T_i \leq \beta_{ij}^5 - t'_j & \text{for all } i \neq j \\
T_j - t'_j - \gamma_{ij}^6 \leq T_i & \text{for all } i \neq j \\
T_i \leq \gamma_{ji}^6 + T_j - t'_j & \text{for all } i \neq j \\
\alpha_k^2 + \alpha_i^4 - t'_k \leq T_i & \\
T_k - t'_k - \beta_k^4 + \alpha_i^4 \leq T_i & \\
\alpha_i^2 + \alpha_i^4 \leq T_i & \\
T_i \leq \beta_k^2 + \beta_i^4 - t'_k & \text{for all } k \neq i \\
T_i \leq T_k - t'_k + \beta_i^4 - \alpha_k^4 & \text{for all } k \neq i \\
T_i \leq \beta_i^2 + \beta_i^4 &
\end{cases}
$$

Again, producing an expression $A_m \leq B_n$ from a pair of inequalities $A_m \leq T_i$ and $T_i \leq B_n$ in $D_{T_i} \cup X_{T_i}$ either produces a tautology, or an inequality of one of the forms in Defn. 3. Hence, domains are closed under computation of a successor for all symbolic moves.

E Soundness and completeness of state class graphs abstraction

Proposition 4 Let $\rho^S = (M_0, B_0, D_0) \xrightarrow{e_0} (M_1, B_1, D_1) \xrightarrow{e_1} \dots$ be a symbolic run of a trajectory $\mathcal{N}$. Then, there exists a run $\rho = (M_0, B_0, \mathcal{T}_0) \xrightarrow{e_0} (M_1, B_1, \mathcal{T}_1) \xrightarrow{e_1} \dots$ of $\mathcal{N}$ such that for every $i \geq 0$, $(M_i, B_i, \mathcal{T}_i)$ matches with (M_i, B_i, D_i).

Proof. To show that for any **finite** symbolic run $(M_0, B_0, D_0) \to \dots \to (M_n, B_n, D_n)$ (where the domain of each state class has a feasible solution), there exists a corresponding run of the trajectory net $(M_0, B_0, \mathcal{T}_0) \to \dots \to (M_n, B_n, \mathcal{T}_n)$, we will proceed by backward induction on the length of runs. Assume a run ending in state class (M_n, B_n, D_n).

First for the base case, i.e. sequences of length 1 ending in state class (M_n, B_n, D_n), we can show that there always exists a map $\mathcal{T}_n$ such that $\mathcal{T}_n \in [\![D_n]\!]$ because D_n is satisfiable. So $(M_n, B_n, \mathcal{T}_n)$ matches (M_n, B_n, D_n). It then remains to show that, if we can build a matching run up to length j, i.e. find a run $(M_j, B_j, \mathcal{T}_j) \xrightarrow{e_j} \dots \xrightarrow{e_{n-1}} (M_n, B_n, \mathcal{T}_n)$ such that $(M_k, B_k, \mathcal{T}_k)$ matches (M_k, B_k, D_k) for $k \in j \dots n$, then we can find a predecessor configuration for $(M_j, B_j, \mathcal{T}_j)$ and extend this run. We have two types of symbolic transitions:

1. For a transition $(M_i, B_i, D_i) \xrightarrow{\sigma}_S (M_j, B_j, D_j)$ of the state class graph, let us show that there exists a corresponding transition in the run of trajectory net $(M_i, B_i, \mathcal{T}_i) \xrightarrow{\sigma} (M_j, B_j, \mathcal{T}_j)$ with $\mathcal{T}_i \in [\![D_i]\!]$. Let $p(\sigma)^\bullet \cap P_T$. The map depicting trajectories $\mathcal{T}_i$ can be constructed simply by dropping the variables t_p, T_p of the forward trajectory from $\mathcal{T}_j$. The trajectories in other places are unaffected by the firing of σ. By induction hypothesis, we know that $\mathcal{T}_j \in [\![D_j]\!]$. Since the inequalities involving variables of unaffected trajectories are the same in D_i and D_j, we have $\mathcal{T}_i \in [\![D_i]\!]$.
2. For a transition $(M_i, B_i, D_i) \xrightarrow{\text{block } p}_S (M_j, B_j, D_j)$, let us show that there exists a corresponding pair of moves in the trajectory net $(M_i, \mathcal{T}_i) \xrightarrow{\delta=t_p} (M_i, \mathcal{T}_i'') \xrightarrow{\text{block } p} (M_j, \mathcal{T}_j)$ with $\mathcal{T}_i \in [\![D_i]\!]$. The trajectories $\mathcal{T}_i$ can be constructed from $\mathcal{T}_j$ as follows:
   - By the correctness of Fourier-Motzin Elimination, there exists t_p, T_p such that $\mathcal{T}_j \cup \{t_p, T_p\}$ is a solution to the domain D_i'' of the transformed variables. Also, $\mathcal{T}_i'' = \mathcal{T}_j \cup \{t_p'' = 0, T_p'' = T_p\}$ is the required trajectories for the configuration.
   - We can fix consistent values for t_p and T_p, and then, perform the inverse transformation $T_i = T_i''$ and $t_i = t_i'' + t_p$ to obtain $\mathcal{T}_i \in [\![D_i]\!]$.

   Then, $\mathcal{T}_i \in [\![D_i]\!]$ by construction of $\mathcal{T}_i$ and $(M_i, \mathcal{T}_i) \xrightarrow{\delta=t_f} (M_i, B_i, \mathcal{T}_i'') \xrightarrow{\text{block } p} (M_j, B_j, \mathcal{T}_j)$ is the corresponding transition extending the run of $\mathcal{N}$.

Proposition 5 Let $\rho = (M_0, B_0, \mathcal{T}_0) \xrightarrow{e_0} (M_1, B_1, \mathcal{T}_1) \xrightarrow{e_1} \dots$ be a run of a trajectory net $\mathcal{N}$. Then, there exists a symbolic run $\rho^S = (M_0, B_0, D_0) \xrightarrow{e_0}$

$(M_1, B_1, D_1) \xrightarrow{e_1} \dots$ of $\mathcal{N}$ such that for every $i \geq 0$, $(M_i, B_i, \mathcal{T}_i)$ matches with (M_i, B_i, D_i).

Proof. We proceed by induction on the length of runs to show that symbolic runs have a counterpart run in the concrete semantics of $\mathcal{N}$. Let us start with the base case, i.e. a run of size 1. The initial configuration $(M_0, B_0, \mathcal{T}_0)$ matches the state class (M_0, B_0, D_0) because values $T_p^0 = t_p^0$ for progressing trajectories are sampled from the respective time intervals $[\alpha_p^S, \beta_p^S]$ associated to trajectory places containing a progressing trajectory.

Let us consider that the property holds up to index n, that is, for the run $(M_0, B_0, \mathcal{T}_0) \xrightarrow{\delta_0} (M_0, B_0, \mathcal{T}_0 + \delta_0) \xrightarrow{e_0} (M_1, B_1, C_1) \dots (M_n, B_n, \mathcal{T}_n)$, there exists a sequence $(M_0, B_0, D_0) \xrightarrow{e_0} (M_1, B_1, D_1) \dots (M_n, B_n, D_n)$ such that $(M_i, B_i, \mathcal{T}_i)$ matches (M_i, B_i, D_i) for every $i \leq n$.

We know from the induction hypothesis that $\mathcal{T}_n \in [\![D_n]\!]$. Let us now consider possible successors of $(M_n, B_n, \mathcal{T}_n)$. We have two possible types of moves:

1. A move of the form $(M_n, B_n, \mathcal{T}_n) \xrightarrow{\sigma} (M_j, B_j, \mathcal{T}_j)$. By the semantics of model, the effect of firing of a transition σ on trajectories is the removal of a blocked trajectory and of tokens in the preset of σ and the creation of tokens in the postset of σ and of a new trajectory in $(\sigma)^\bullet \cap P_T)$. The rest of trajectories are unaffected by the firing. By construction of successor of state classes, we know that markings and blocked transitions follow the same rules in the semantics and in the symbolic moves. Hence, if $(M_n, B_n, D_n) \xrightarrow{\sigma}_S (M_{n+1}, B_{n+1}, D_{n+1})$, we necessarily have $M_{n+1} = M_j, B_{n+1} = B_j$. We also know that D_{n+1} is uniquely defined from D_n and σ (it is the canonical form of $SuccF(D_n, \sigma)$. It hence remains to show that $\mathcal{T}_j \in [\![D_{n+1}]\!]$. We know that D_n is satisfiable, because $\mathcal{T}_n \in [\![D_n]\!]$. Let p be the place receiving the new trajectory ofter firing of σ. We have $\mathcal{T}_j(p_k) = \mathcal{T}_n(p_k)$ for every place $p_k \neq p$, and $\mathcal{T}_j(p) = (T_p, T_p)$ with $T_p \in [\alpha_p^s, \beta_p^s]$. To compute D_{n+1}, we add the inequalities defined in $SuccF(D_n, \sigma)$ (see Section 4.1). The fresh variables satisfy the inequalities $\alpha_p^s \leq T_p \leq \beta_p^s$, $\alpha_p^s \leq t_p \leq \beta_p^s$ and $0 \leq T_p - t_p \leq 0$. The other inequalities in D_{n+1} involve only variables of unaffected trajectories and are unchanged from D_n. As $\mathcal{T}_n \in [\![D_n]\!]$, we have that $\mathcal{T}_j$ satisfies all inequalities in D_{n+1} too. So the induction step preserves the property for transitions firings.
2. A pair of moves $(M_n, B_n, \mathcal{T}_n) \xrightarrow{\delta = t_p} (M_n, B_n, \mathcal{T}_n'') \xrightarrow{\text{block p}} (M_j, B_j, \mathcal{T}_j)$. According to the semantics, we have $\mathcal{T}(p) = (T_p, t_p)$ and $t_p = \min(\{t_i\})$. Let $(M_n, B_n, D_n) \xrightarrow{block\ p}_S (M_{n+1}, B_{n+1}, D_{n+1})$, where D_{n+1} is the canonical form of $SuccB(D, p)$. The configuration $(M_j, B_j, \mathcal{T}_j)$ must match the state class $(M_{n+1}, B_{n+1}, D_{n+1})$. We have $M_{n+1} = M_n = M_j$, and $B_{n+1} = B_j = B_n \cup \{p\}$. It hence remains to show that $\mathcal{T}_j \in [\![D_{n+1}]\!]$. Performing the timed move $\delta = t_p$ from $(M_n, B_n, \mathcal{T}_n)$ consists in replacing trajectories of the form (T_i, t_i) by trajectories (T_i'', t_i''), to obtain configuration $(M_n, B_n, \mathcal{T}_n'')$. Let us now denote by D_n'' the domain obtained after replacing variables t_i by $t_i'' + t_p$. We have $\mathcal{T}_n'' \in [\![D_n'']\!]$. Note that this variable change is exactly the

first step performed when computing $SuccB(D, p)$. The rest of the calculus is the elimination of variables T_p and t_p using the Fourier-Motzkin projection. By correctness of Fourier-Motzkin elimination, we know that for any solution μ'' of D_i'', the projection of μ'' on remaining variables is a solution of D_{n+1}. According to the semantics of trajectory nets, $\mathcal{T}_j$ is the projection of $\mathcal{T}_n''$ on variables $\{T_k, t_k \mid k \neq p \wedge \mathcal{T}_n''(k)\text{is defined}\}$. Hence, $\mathcal{T}_j \in [\![D_{n+1}]\!]$, and this induction step preserves matching.

Theorem 3 Given a state class (M_n, B_n, D_n) reachable from initial state class (M_0, B_0, D_0), and a solution $\mathcal{T}_n \in [\![D_n]\!]$, there exists a run in the original trajectory net that ends in configuration $(M_n, B_n, \mathcal{T}_n)$.

Proof. The proof is similar to the proof for soundness (4). Let us assume that a run in the state class graph is $\rho^S = (M_0, B_0, D_0) \to \cdots \to (M_i, B_i, D_i) \to (M_j, B_j, D_j) \to \cdots \to (M_n, B_n, D_n)$. We can inductively construct a run ρ of the trajectory net that matches ρ^S. Let us assume that we have a partial run $(M_j, B_j, \mathcal{T}_j) \to \cdots \to (M_n, B_n, \mathcal{T}_n)$ with the induction hypothesis that $\mathcal{T}_j \in [\![D_j]\!]$. We can construct $(M_i, B_i, \mathcal{T}_i)$ such that $\mathcal{T}_i \in [\![D_i]\!]$. We can have two cases:

- $(M_i, B_i, D_i) \xrightarrow{\sigma}_S (M_j, B_j, D_j)$. The effect of firing σ on markings and blocked transitions is determinisitic, and is the same in the symbolic moves and in the semantics. Hence, we have $M_i = M_j + {}^\bullet(\sigma) - (transition)^\bullet$, $B_i = B_j$ and it remains to find a map $\mathcal{T}_i$ such that $\mathcal{T}_i \in [\![D_i]\!]$. Let t_p, T_p represent the variables associated with the trajectory created in place $p = (\sigma)^\bullet \cap P_T$ when firing σ. In this case, $\mathcal{T}_i$ is the projection of $\mathcal{T}_j$ on all its variables except $\{t_p, T_p\}$. Clearly, $\mathcal{T}_i$ satisfies the induction hypothesis $\mathcal{T}_i \in [\![D_i]\!]$. This is because, $D_j = SuccF(D_i, \sigma)$. Hence, when building D_j from D_i, we just add inequalities involving variables t_p, T_p and the remaining inequalities of D_j are unaffected. Hence the projection of $\mathcal{T}_j$ gives a map $\mathcal{T}_i \in [\![D_i]\!]$, and the discrete move $(M_i, B_i, \mathcal{T}_i) \xrightarrow{\sigma} (M_j, B_j, \mathcal{T}_j)$ is a valid move in the semantics of trajectory net.
- $(M_i, B_i, D_i) \overset{\text{block p}}{\longrightarrow}_S (M_j, B_j, D_j)$. By correctness of Fourier-Motzkin elimination, we know there exists t_p, T_p such that $\mathcal{T}_i'' = \mathcal{T}_j \cup \{p \to (T_p, t_p)\}$ is a solution of domain D_i'' obtained after transformation of variables in D_i. Then, for every fixed pair of values t_p, T_p, $\mathcal{T}_i = \{t_i, T_i | t_i = t_j - t_p, T_i = T_j\} \cup \{t_p, T_p\}$ is the set of trajectories satisfying the induction hypothesis $\mathcal{T}_i \in [\![D_i]\!]$. This is because:
  - By construction of successor of domains, we know after performing the inverse transformation $t_i = t_i'' - t_p$ and $T_i = T_i''$ on $\mathcal{T}_i''$ gives a solution a solution $\mathcal{T}_i$ which satisfies the inequalities $t_p \leq t_i, \forall i \neq p$, in addition to all the inequalities of D_i. Hence $\mathcal{T}_i \in [\![D_i]\!]$.
  - Also $(M_i, B_i, \mathcal{T}_i) \overset{\delta = t_p}{\longrightarrow} (M_i, B_i, \mathcal{T}_i'')$ is a valid time move since $t_p = \min\{t_i\}$.

  Hence, $(M_i, B_i, \mathcal{T}_i) \overset{\delta = t_p}{\longrightarrow} (M_i, B_i, \mathcal{T}_i'') \overset{\text{block p}}{\longrightarrow} (M_j, B_j, \mathcal{T}_j)$ are valid moves accoridng to the semantics of trajectory net and the induction hypothesis is satisfied.

By induction, we can hence construct a valid run of $\mathcal{N}$ which ends in a configuration $(M_n, B_n, \mathcal{T}_n)$ matching state class (M_n, B_n, D_n). When considering the last step $(M_0, B_0, D_0) \xrightarrow{e_0} (M_1, B_1, D_1)$, the induction step is still valid, but the matching run must start from $C_0 = (M_0, B_0, \mathcal{T}_0)$. So, assuming that $\mathcal{T}_0(p_i) = (T_i, T_i)$ with $T_i > 0$ for every non-empty place in P_T, e_0 is the blocking of a trajectory in some place p, and the choice of t_p, T_p is restricted to $T_p = t_p = \mathcal{T}_0(p)$.

F Finiteness of Domains

Lemma 1. *Let $\mathcal{N}$ be a trajectory net, and D be a canonical domain computed inductively from the initial domain D_0 of $\mathcal{N}$. The constants appearing in D are linear combinations of α_i^s, β_i^s and T_i^0 with integer coefficients.*

Proof. We can reuse the analysis of expressions generated by variable elimination from a domain D in the proof of Prop. 3 to show that expressions of the form $expr_j + bj \leq expr_k + b_k$ can be equivalently rewritten in an expression of the form $expr_j - expr_k \leq b_k - bj$ and are hence linear combinations of constants appearing in D. As constants in D_0 are T_i^0's, and new constants introduced by successors are α_i^s and β_i^s, we can conclude.

Lemma 2 ([5], Lemma 4). *Let A, B be constants, and $q_1, \ldots, q_n$ be rational constants. Then there is only a bounded number of linear combinations of $q_1, \ldots q_n$, with integer coefficients between A and B.*

Lemma 3. *Let $\mathcal{N}$ be a trajectory net, and D be a domain computed inductively from the initial domain D_0 of $\mathcal{N}$. Let C_{max} be the maximal value appearing in an interval $[\alpha_i^s, \beta_i^s]$. The constants appearing in D are in interval $[-2{\cdot}C_{max}, 2{\cdot}C_{max}]$*

Proof. Consider the general form of domains given in Defn. 3. All inequalities are of the form $\alpha \leq \ \text{expr} \ \leq \beta$, with $expr = \sum_{i \in S_1} A_i x_i + \sum_{i \in S_2} B_i x_i$ where x_i are variables with bounds $\alpha_i \leq x_i \leq \beta_i$ and $A_i > 0$ and $B_i < 0$. We can prove using Lemmas 4,5,6 below) that if bounds for all x_i's exist, then we can derive a bound for $expr$. For trajectory nets, we immediately have bounds $[\alpha_i^s, \beta_i^s]$ for variable T_i and $[0, \beta_i^s]$ for variable t_i.

Lemma 4. *Given an inequality $\alpha \leq \ expr \ \leq \beta$, with $expr = \sum_{i \in S_1} A_i x_i + \sum_{i \in S_2} B_i x_i$ where x_i are variables with bounds $\alpha_i \leq x_i \leq \beta_i$ and $A_i > 0$ and $B_i < 0$, an equivalent inequality is:*

$$\max(\alpha, \sum_{i \in S_1} A_i \alpha_i + \sum_{i \in S_2} B_i \beta_i) \leq \ expr \ \leq \min(\beta, \sum_{i \in S_1} A_i \beta_i + \sum_{i \in S_2} B_i \alpha_i)$$

Proof. Note that the minimum and maximum possible values that $expr$ can take, based on the bounds on x_i are $\sum_{i \in S_1} A_i \alpha_i + \sum_{i \in S_2} B_i \beta_i$ and $\sum_{i \in S_1} A_i \beta_i + \sum_{i \in S_2} B_i \alpha_i$ respectively. The Lemma 4 follows directly from this.

Lemma 5. *For the same settings as Lemma 4 if* $\alpha > \sum_{i\in S_1} A_i\beta_i + \sum_{i\in S_2} B_i\alpha_i$ *or* $\beta < \sum_{i\in S_1} A_i\alpha_i + \sum_{i\in S_2} B_i\beta_i$*, the inequality has no solution*

Proof. The proof again follows from the bound on expr based on bounds on x_i

Lemma 6. *For the inequality in Lemma 4, if it has a solution, then there exists an equivalent inequality* $\alpha' \leq \ expr \ \leq \beta'$ *with:*

$$\sum_{i\in S_1} A_i\alpha_i + \sum_{i\in S_2} B_i\beta_i \leq \alpha' \leq \sum_{i\in S_1} A_i\beta_i + \sum_{i\in S_2} B_i\alpha_i$$
$$\sum_{i\in S_1} A_i\alpha_i + \sum_{i\in S_2} B_i\beta_i \leq \beta' \leq \sum_{i\in S_1} A_i\beta_i + \sum_{i\in S_2} B_i\alpha_i$$

Proof.
- The lower bound on α' comes from Lemma 4 with $\alpha' = \max(\alpha, \sum_{i\in S_1} A_i\alpha_i + \sum_{i\in S_2} B_i\beta_i)$
- The upper bound on α' comes from Lemma 5 since the inequality has a solution
- The lower bound on β' comes from Lemma 5 since the inequality has a solution
- The upper bound on β' comes from Lemma 4 with $\beta' = \min(\beta, \sum_{i\in S_1} A_i\beta_i + \sum_{i\in S_2} B_i\alpha_i)$

In the setting of trajectory nets, we immediately have bounds for variables t_i and T_i: T_i is sampled from interval $[\alpha_i^s, \beta_i^s]$ and since time remaining time t_i for a trajectory has an original value T_i and then decreases we have $t_i \in [0, \beta_i^s]$. Now using Lemma 6, we have the following bounds on the constants for an equivalent system of inequalities to have a solution:

$$\alpha_i^s \leq \alpha_i^1 \leq \beta_i^s \tag{7}$$
$$\alpha_i^s \leq \beta_i^1 \leq \beta_i^s \tag{8}$$
$$0 \leq \alpha_i^2 \leq \beta_i^s \tag{9}$$
$$0 \leq \beta_i^2 \leq \beta_i^s \tag{10}$$
$$-\beta_j^s \leq \gamma_{ij}^3 \leq \beta_i^s \tag{11}$$
$$\alpha_i^s - \beta_i^s \leq \alpha_i^4 \leq \beta_i^s \tag{12}$$
$$\alpha_i^s - \beta_i^s \leq \beta_i^4 \leq \beta_i^s \tag{13}$$
$$\alpha_i^s - \beta_i^s \leq \alpha_{ij}^5 \leq \beta_i^s + \beta_j^s \tag{14}$$
$$\alpha_i^s - \beta_i^s \leq \beta_{ij}^5 \leq \beta_i^s + \beta_j^s \tag{15}$$
$$-\beta_i^s + \alpha_j^s - \beta_j^s \leq \gamma_{ij}^6 \leq -\alpha_i^s + \beta_i^s + \beta_j^s \tag{16}$$

G Expressiveness of Trajectory nets

As Trajectory nets are a timed variant of Petri nets, it is natural to ask how they compare with other timed variants of nets. As trajectory nets have a notion

of urgency, the closest mode in of course Time Petri nets [19] (TPNs for short). We compare these two formalisms in terms of generated timed languages. For a given run $\rho = C_0 \xrightarrow{\delta_0} C'_0 \xrightarrow{\sigma_0} \cdots$ of a trajectory net (resp. of a TPN), the timed word w_ρ associated with ρ is a sequence $w_\rho = (\sigma_0, d_0)(\sigma_1, d_1)\ldots$ where $\forall i \in \mathbb{N}, d_i = \sum_{j \leq i} \delta_j$. The timed language of a trajectory net $\mathcal{N}$ is the set of timed words associated with a run starting from C_0, i.e. $\mathcal{L}(\mathcal{N}) = \{w_\rho \mid \rho = C_0 \xrightarrow{\delta_0} C'_0 \xrightarrow{\sigma_0} \cdots\}$

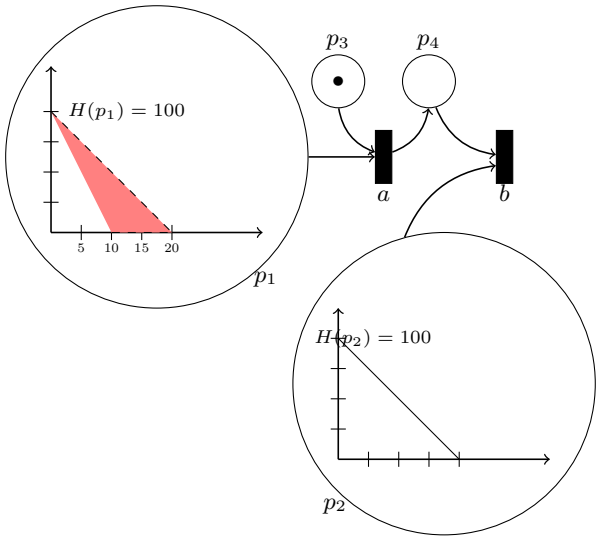

Fig. 6. A trajectory net without TPN counterpart.

Unbounded TPNs and unbounded trajectory nets are Turing Powerful. If silent transitions are allowed, then a trajectory net can simulate a Petri net and conversely. We hence consider nets without silent transitions and with injective labelings. Let us first compare bounded TPNs and bounded trajectory nets. Consider the example of Figure 6. Let $H(p_1) = H(p_2) = 100$, $I(p_1) = [10; 20]$, $I(p_2) = [20, 20]$

In this net, transition a can fire at a date d_a, and transition b can fire at date d_b such that $d_b \geq \max(20, d_a)$. In particular, it means that b must occur after a, and that the timed language of this net is composed timed words of the form $\mathcal{L}_1 = (a, x).(b, 20)$ where $10 \leq x \leq 20$. Assume that this language can be encoded with a TPN with two transition a and b. Firing b after a calls for the existence of a place p with ${}^\bullet(p) = a$ and $(t)^\bullet = b$. The timed interval attached to a is necessarily $10, 20$. Let $[\alpha, \beta]$ be the time interval attached to b. The language of the TPN is of the form $\mathcal{L}_2 = (a, x).(b, y)$, with $x \in [10, 20]$ and $x + \alpha \leq y \leq x + \beta$. We can show that there exists no pair of values for α, β such

that $\mathcal{L}_1 = \mathcal{L}_2$, because choosing value $x = 20$ enforces $\alpha = \beta = 0$, which is not a suitable time interval if one chooses $x < 20$.

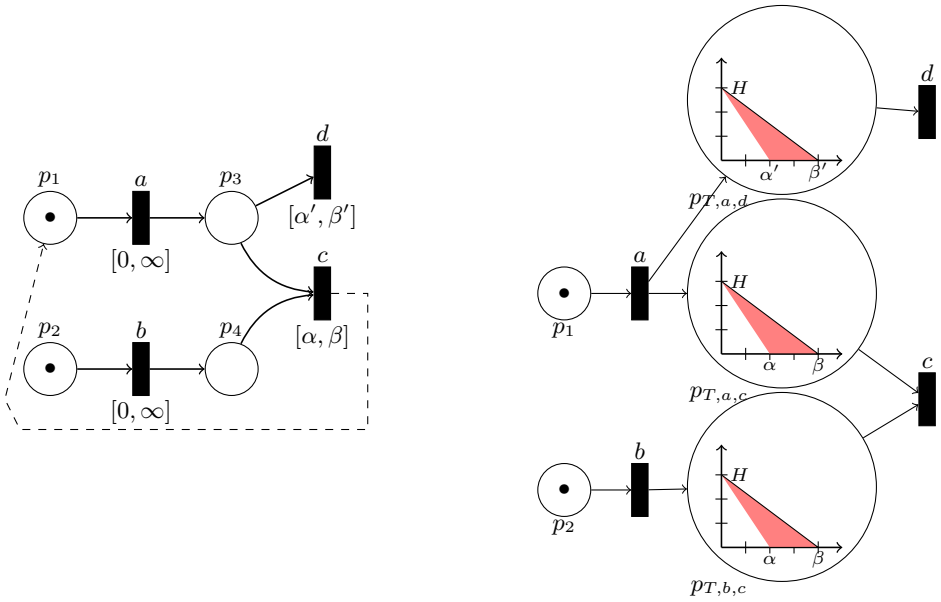

Fig. 7. A TPN without trajectory net counterpart.

Now consider the TPN on the left of Figure 7 (without the dashed flow relation). The timed language of this TPN is

$$\begin{aligned}\mathcal{L}_3 = \; & \{(a,d_a).(b,d_b).(c,d_c) \mid d_a,d_b,d_c \in \mathbb{R}^+, d_b+\alpha \le d_c \le d_b+\beta \wedge d_c \le d_a+\beta'\} \\ & \cup\{(b,d_b).(a,d_a).(c,d_c) \mid d_a,d_b,d_c \in \mathbb{R}^+, d_a+\alpha \le d_c \le d_a+\beta \wedge d_c \le d_a+\beta'\} \\ & \cup\{(a,d_a).(b,d_b).(d,d_d) \mid d_a,d_b,d_d \in \mathbb{R}^+, d_a+\alpha' \le d_d \le d_a+\beta' \wedge d_d \le d_b+\beta\} \\ & \cup\{(b,d_b).(a,d_a).(d,d_d) \mid d_a,d_b,d_d \in \mathbb{R}^+, d_a+\alpha' \le d_d \le d_a+\beta' \wedge d_d \le d_b+\beta\} \\ & \cup\{(a,d_a).(d,d_d).(b,d_b) \mid d_a,d_b,d_d \in \mathbb{R}^+, d_a+\alpha' \le d_d \le d_a+\beta' \wedge d_d \le d_b\}\end{aligned}$$

In this net, transition c fires between α and β time units after the firing of the last event between a and b. Timing constraints in trajectory nets can only be enforced by time intervals associated with trajectory places. Hence, if one wants to satisfy the tiling constraints $I(c) = [\alpha, \beta]$ in the net of Figure 7, a trajectory net must have two trajectory places $p_{T,a,c} \in (a)^\bullet \cap {}^\bullet(c)$ and $p_{T,b,c} \in (b)^\bullet \cap {}^\bullet(c)$ with $I(p_{T,a,c}) = I(p_{T,b,c}) = [\alpha, \beta]$. Similarly, to enforce a firing of d between α' and β' time units after the firing of a, a trajectory net must have a trajectory places $p_{T,a,d} \in (a)^\bullet \cap {}^\bullet(c)$ with $I(p_{T,b}) = [\alpha, \beta]$. Now consider the extension of the net of Figure **??** with the additional dashed flow between c and p_1. This TPN allows words of the form $w = (a, d_a).(b, d_b).(c, d_c).(a, d'_a).(d, d_d)$ with $d_a, d_b, d_c, d'_a, d_d \in$

$\mathbb{R}^+, d_b + \alpha \leq d_c \leq d_b + \beta \wedge d_c \leq d_a + \beta' \wedge d'_a > d_c \wedge d'_a + \alpha' \leq d_d \leq d'_a + \beta'$. Assume a trajectory net with places $p_{T,a,c}$), $p_{T,b,c}$ $p_{T,a,d}$ as described above. Then a trajectory net of this form cannot ply word w, because place $p_{T,a,d}$ still contains a trajectory when the second occurrence of a must fire.