HELSINKI UNIVERSITY OF TECHNOLOGY
DIGITAL SYSTEMS LABORATORY
Series **A:** Research Reports
No. **38**; August 1996

ISSN 0783–5396
ISBN 951–22–3212–X

TIME PROCESSES OF TIME PETRI NETS

Tuomas Aura
Digital Systems Laboratory
Department of Computer Science
Helsinki University of Technology
Otaniemi, FINLAND

Helsinki University of Technology
Department of Computer Science
Digital Systems Laboratory
Otaniemi, Otakaari 1
FIN–02150 ESPOO, FINLAND

HELSINKI UNIVERSITY OF TECHNOLOGY
DIGITAL SYSTEMS LABORATORY
Series **A:** Research Reports
No. **38**; August 1996

ISSN 0783–5396
ISBN 951–22–3212–X

Time processes of time Petri nets

Tuomas Aura

Abstract: The objective of this thesis is to give time Petri nets a partial order semantics, like the nonsequential processes of untimed net systems. A time process of a time Petri net is defined as a traditionally constructed causal process with a valid timing. This means that the events of the process are labeled with occurrence times which must satisfy specific validness criteria. An efficient algorithm for checking validness of known timings is presented. Interleavings of the time processes are defined as linearizations of the causal partial order of events where also the time order of events is preserved. The relationship between firing schedules of a time Petri net and the interleavings of the time processes of the net is shown to be bijective. Also, a sufficient condition is given for when the invalidity of timings for a process can be inferred from its initial subprocess. An alternative characterization for the validness of timings results in an algorithm for constructing the set of all valid timings for a process. The set of all valid timings is presented as sets of alternative linear constraints, which can be used in optimization problems. The techniques developed can be used to compute, for example, the maximum time separation of two events in a process. The existence of a valid timing for a given process can be decided in NP time.

Keywords: Time Petri nets, processes, timing analysis, partial order semantics, causality, net theory

Printing: TKK Monistamo; Otaniemi 1996

Helsinki University of Technology
Department of Computer Science
Digital Systems Laboratory
Otaniemi, Otakaari 1
FIN–02150 ESPOO, FINLAND

Phone: $\frac{90}{+358\ 9}$ 4511
Telex: 125 161 htkk fi
Telefax: +358 9 451 3369
E-mail: lab@saturn.hut.fi

Contents

1 Introduction **4**

1.1 Outline of the thesis 6

2 Time Petri nets **7**

2.1 Definition 7

2.2 Behavior 9

2.3 Boundedness 12

2.4 Zeno's paradox 13

3 Time processes **14**

3.1 Causal processes 14

3.2 Valid timings 17

3.3 Algorithm for verifying validness 21

3.4 Example 23

3.5 Invalid timings and process growth 24

4 Processes and firing schedules **26**

4.1 Interleavings 26

4.2 Relation between interleavings and firing schedules 28

5 All valid timings of a process **38**

5.1 Deciding events 38

5.2 Algorithm for all valid timings 42

5.3 More about invalid timings in growing processes 47

5.4 Example 49

5.5 Extended free choice time Petri nets 50

6 Conclusions **53**

List of symbols and notational conventions

$\wedge$	logical and
$\vee$	logical or
$\Rightarrow$	logical implication
$\exists$	existential quantifier
$\forall$	universal quantifier
$\cap$	set intersection
$\cup$	set union
$\setminus$	set difference
$\subseteq$	subset
$\|X\|$	set cardinality
$\mathbb{T}$	domain of time values
$\mathbb{I}$	time intervals
SI	static interval
${}^\bullet x$	preset of x
$x^\bullet$	postset of x
${}^{\bullet\bullet}e$	previous events
$e^{\bullet\bullet}$	next events
β	branching factor
Eft	earliest firing time
Lft	latest firing time
$Enabled(M)$	enabled transitions
$Fireable(S)$	fireable transitions
θ	firing delay
σ	a firing schedule
ρ	an interleaving
$FS(\rho)$	firing schedule of ρ
τ	timing function
$Min(CN)$	initial elements in CN
$Max(CN)$	final elements in CN
$\perp$	bottom element
$Cut(E)$	cut of E
$Earlier(e)$	earlier events of e
TOE	time of enabling

B	a set of conditions
CN	a causal net
E	a set of events
$EFCTPN$	an extended free choice time Petri net
F	a flow relation of a net
G	a flow relation of a process
I	a clock function
M	a marking
M_0	an initial marking
N	a net
NS	a net system
P	a set of places
S	a state
T	a set of transitions
TPN	a time Petri net
b	a condition
e	an event
p	a net homomorfism
s	a place
t	a transition

1 Introduction

Petri nets [25, 18] are a formalism for modelling and analyzing distributed and concurrent systems. They are characterized by fine grained control over concurrency and synchronization, and equal emphasis of state and actions. Petri nets describe the causal behavior of systems. This is natural, because one of the distinctive characteristics of distributed systems is the lack of global time. Therefore, causal relationships are a more reliable criterion for reasoning about the order of events than timing would be. In practical system, however, timing of events is often just as important as the causal order. Most distributed systems have nonideal features like timeouts and alarms. Furthermore, performance aspects force designers of distributed systems, such as communication protocols, to maintain synchrony between concurrent subsystems with local clocks. The difficulties in designing time dependent distributed systems are likely to increase the demand for formal methods with timing capabilities. Consequently, time extensions are being planned, for example, to the LOTOS specification language [19]. This development is sped by an ever increasing need to specify communication and media systems with critical timing properties, and by the fact that the theory and methods for causal systems have already reached reasonable maturity.

Many time related extensions of Petri net formalisms have been introduced to facilitate performance analysis [24, 17]. Most of them attach the timing information on top of the systems, without having any effect on the causal relations between events. Recently, more attention has been given to net classes where time constraints restrict the causal behavior of the system and limit its state space [8, 12, 33]. Time Petri nets [16] are perhaps the simplest formalism for modelling systems where time limits can force events to occur and keep others from happening. Nevertheless, we find them to be sufficiently powerful for theoretical consideration of time behavior of systems, their simple structure in this case being only an advantage.

In time Petri nets, there is an upper and a lower bound for the time an event can remain enabled without occuring after its preconditions are met. This can be used to model time limits in system specifications and imprecise timing like skew of local clocks, as well as asynchronous timer interrupts. The upper time bound of one potential event can limit the time when another conflicting event can occur, creating dependences not seen in causal systems. Furthermore, the timing limits can be used in a way giving the net class the expressive power of a universal computer [21]. This is a strong indication that the analysis on time Petri nets is more complicated than it is for untimed net classes.

Even though timing constraints reduce the number of reachable markings of the net they, instead, increase the cost of examining the state space. In addi-

tion to the causal state, clock information has to be carried along throughout the analysis. In reachability graphs of time Petri nets [2, 1], the states of the net contain a marking and intervals of possible firing times of all enabled transitions. The states are grouped into state classes where the possible firing times are presented with sets of inequalities. Complex rules are given for transforming the state classes in firings and for checking their equivalence. A variation of the state class method has been considered in [6], making the approach computationally somewhat more feasible. A more comprehensible, although not so optimized reachability graph has been proposed by [23]. Being much simpler than the state class approach, this is a good point of comparison for new and improved analysis methods. In it, a state contains the marking and the readings of the local clocks of enabled transitions. State changes are divided in two types: either time passes or a transition fires.

An alternative approach to inspecting the behavior of concurrent systems is to use the partial order semantics, like nonsequential processes [22, 13, 5] and branching processes [20, 9]. The research on branching processes has lately lead to efficient deadlock detection and model checking algorithms for net systems [15, 11]. It is rather obvious that the benefits of the partial order approach to the analysis of systems should be assessed also with regard to the analysis of timed systems. Processes have been defined and successfully utilized for some net classes with time [31, 14, 30], but in these cases the timing does not interfere with the causal order of events. We will show in this theses that nonsequential processes can be successfully used in presenting the behavior of time Petri nets. Although the occurrence times of events seem to put even causally unrelated events in a sequential order, the concurrent parts of the process develop independently within the specified time limits. In a causal system, the relative speed of concurrent events can be arbitrary and order results only from synchronization by shared events. In time constrained systems, the speeds are given some bounds, but the events are not forced to a fixed sequence. Instead, the order of events arises from synchronizing events, or from the time limits of enabled events representing potential interactions. Thus, the seemingly global dependences created by the timing derive from local causes.

In this thesis, we propose a notion of time process for time Petri nets, examine carefully the relation between the time processes and the firing schedules of the nets, and present algorithms for verifying validness of timings and constructing valid timings for a given process. The basic idea is that under time constraints, not all processes or timings of the events are possible. We find out which processes have a valid timing and which do not, and what kind of timings a process can have. With the developed techniques, questions like, what is the maximum time separation between two events in a given process, can be answered.

1.1 Outline of the thesis

Section 2 contains basic definitions. The time Petri net formalism and fundamental notions about the behavior of time Petri nets are defined. Emphasis is put on the details that differ from the conventional definitions in the literature. The class of systems is restricted to contact-free ones with divergent time.

The most important new concepts, valid timing and time process, are introduced in Section 3. The time processes of time Petri nets are constructed by labeling the events of processes of untimed net system with time values. Criteria is given for when the choice of time values should be considered a valid representation of system behavior. An efficient algorithm is presented for verifying the validness. The section ends with discussion on the persistence of invalidity in growing processes.

Discussion and proofs about the exact relationship between a time Petri net and its time processes can be found in Section 4. The relation between firing schedules of a time Petri net and interleavings of the time processes of the net is shown to be bijective.

In Section 5, the set of all valid timings of a process is characterized with linear inequalities. The discussion is based on the idea that the occurrence of every potential event has to be somehow decided by an actual event of the process. The results can be used to analyze properties of all valid timings, such as maximal time separation of events.

The last Section 6 summarizes the thesis and draws some conclusions.

2 Time Petri nets

Time Petri nets are a simple yet powerful formalism for modelling concurrent systems with time constraints. In time Petri nets, transitions are labeled with time intervals. There is an upper and a lower limit for the time a transition can be enabled before firing. This makes it possible to model imperfect timing where the exact durations of events are not known. Time limits in real systems are often specified in the same way, by giving worst case boundaries. Time Petri nets can be used to analyze absolute limits of timing and effects of the time constraints on the state space of the system. A wide variety of time-related phenomena can be observed in them.

There are other net formalisms for timed systems, all with their specific strengths and weaknesses. For practical modelling, time stream nets [8, 27] might be a better choice than time Petri nets. They allow the modeller to choose from several alternative synchronization policies for each transition. Time stream nets can be reduced to time Petri nets, which means that analysis methods for time Petri nets should be easily adaptable to them. The TB nets in [12] are an interesting generalization of different time net classes, giving the modeller great freedom in choosing the time and firing semantics of the net. The TB net formalism is significantly more expressive than any other time net classes in the literature. In performance analysis, timed Petri nets [24], where every transition has a fixed firing duration, are a popular formalism, along with various stochastic nets [17], in which the firing duration has a known statistical distribution. These net classes are optimized for efficient and straightforward analysis of average behavior. There are also numerous restrictions and variations of these net classes, most of which have been developed with some special modelling needs or analysis algorithms in mind.

We begin by giving an exact definition of time Petri nets in Section 2.1 and continue with fundamental notions about their behavior in Section 2.2. Assumptions about contact-freeness and divergence of time will be made in Sections 2.3 and 2.4.

2.1 Definition

The domain of time values $\mathbb{T}$ is either nonnegative real numbers, nonnegative rational numbers or natural numbers. We denote by $[\tau_1, \tau_2]$ the closed interval between two time values $\tau_1, \tau_2 \in \mathbb{T}$, and by $\mathbb{I}$ the set of such intervals. Infinity is allowed at the upper bound.

$$\mathbb{I} = \{[\tau_1, \tau_2] \mid \tau_1 \in \mathbb{T} \ \wedge \ \tau_2 \in \mathbb{T} \cup \{\infty\}\}\,. \tag{1}$$

An interval can be of zero length ($\tau_1 = \tau_2$), containing only a single time value.

Definition 1 (time Petri net) A *time Petri net* is a five-tuple $TPN = (P, T, F, SI, M_0)$, where

P is a set of *places*,
T is a set of *transitions*,
$P \cap T = \emptyset$,
$F \subseteq (P \times T) \cup (T \times P)$ is a *flow relation*,
$SI : T \longrightarrow \mathbb{I}$ is a function called *static interval*, and
$M_0 \subseteq P$ is the *initial marking* of the time Petri net.

The boundaries of the static interval are called *earliest firing time* Eft and *latest firing time* Lft. The *preset* of $x \in P \cup T$ is ${}^\bullet x = \{y \mid yFx\}$ and *postset* is $x^\bullet = \{y \mid xFy\}$. The members of the preset and the postset of a transition t are called *preplaces* and *postplaces* of t, respectively. A *marking* $M \subseteq P$ is a subset of the places.

It is assumed that the initial marking is finite, $|M_0| < \infty$, and that there is some constant *branching factor* $\beta < \infty$ such that $|{}^\bullet x| < \beta$ and $|x^\bullet| < \beta$ for all $x \in P \cup T$. Moreover, the presets and postsets of transitions must be nonempty, $|{}^\bullet t| > 0$ and $|t^\bullet| > 0$ for all $t \in T$. □

Time Petri nets were introduced in [16]. The above definition does not strictly follow any previous definition, but rather combines the properties that we find useful for this discussion. We will outline the differences below. Basically, what we have is the class of elementary net systems [29] enriched with the static intervals on transitions, and with some finiteness requirements.

There are no weights on the arcs, and the markings are represented by sets, limiting the capacity of all places to one. This differs from the normal definition of time Petri net but coincides with the 1-safeness assumption made by most analysis methods. The time Petri nets are of finite synchronization, meaning that presets and postsets are finite. There are no sinks or sources, that is, transitions with empty presets or postsets.

In the literature, earliest and latest firing times are often allowed to have rational values but no real values. Real numbers are not permitted because the state space of a finite net could then become infinite [2], and irrational values would be difficult to handle in computation. In practical models, there is no need for real numbers in any case, as the time bounds are usually worst case estimates of physical timing limits, or sometimes exact integers in digital systems. As noted in [23], the static intervals can as well be required to have integer boundaries instead of the more common rationals. In finite nets, rational intervals can be converted to integers by multiplying the boundary values by the least common multiple of their denominators. Rational numbers

are used either for ease of modelling or in order to blur the restriction to discrete division of time for interval boundaries. The choice of the domain for time values is not important for us. Therefore, we allow all reasonable alternatives.

We allow the boundaries of a static interval to equal zero, which means that the transition in question can fire without delay. Such firings and the corresponding events are called *immediate*. The class of systems that can be modelled with this extension is significantly larger than it would be if immediate events were not allowed. Furthermore, we will see that the complexity of timing analysis is not affected by the immediate events. However, we will disallow infinite sequences of immediate firings that consume no time or only a bounded amount of time in total.

In order to simplify the notation, we extend the domain of preset and postset functions from single elements to sets of elements so that the value is the union of the individual presets or postsets. When $X \subseteq P \cup T$,

$$ {}^\bullet X = \{y \in P \cup T \mid x \in X \wedge y \in {}^\bullet x\} \text{ and} \tag{2} $$

$$ X^\bullet = \{y \in P \cup T \mid x \in X \wedge y \in x^\bullet\}, \tag{3} $$

This makes it possible to use the the notations ${}^{\bullet\bullet}e$ and $e^{\bullet\bullet}$ for the sets of previous and next events in a causal order in the later sections.

2.2 Behavior

The fundamental concepts of net behavior like enabledness and firing will defined by adapting the definitions of [28] to our net class. They differ somewhat from the more usual ones in [2].

Definition 2 (enabled) A transition t of a time Petri net is *enabled* at marking M iff ${}^\bullet t \subseteq M$. The set of all enabled transitions at marking M is denoted by $Enabled(M)$. □

Note that being enabled does not necessarily mean that the transition can fire. It simply means that all preplaces of the transition are filled. Transitions have to be enabled for the length of their earliest firing time before they can fire, as will be specified in the firing condition.

In time Petri nets, a marking is not sufficient information to describe a complete state of the system. The state must also include timing information. This is given as a clock function that, for each enabled transition, gives the amount of time passed since its enabling. The definition of state, and the idea for the firing condition and firing rule between states are from [28].

Definition 3 (state) A *state* of a time Petri net $TPN = (P, T, F, SI, M_0)$ is a pair $S = (M, I)$, where

M is a marking of TPN, and

$I : Enabled(M) \longrightarrow \mathbb{T}$ is called the *clock function*.

The *initial state* of TPN is $S_0 = (M_0, I_0)$, where $I_0(t) = 0$ for all $t \in Enabled(M_0)$. □

The state of a time Petri net can change in two ways:.

1. Time passes but no transition fires: Only the clock function is adjusted.
2. A transition fires but no time passes: The firing changes the marking of the net. Clocks of disabled transitions are discarded and new clocks are started for newly enabled transitions.

The total time that can pass before the next transition fires is limited by the firing intervals of enabled transitions.

The division into two different kinds of state changes captures the basic idea of time Petri nets. For technical considerations, it is, however, preferable to have only one kind of state change. Therefore, the two changes are combined into a single step where time first passes and a transition then fires. Two or more state changes by passing of time can obviously be combined into one that takes the sum of the two times. Two simultaneous firings cannot as easily be merged into one. We can, however, add a zero length time passing step between any transitions firing without delay. This way, any sequence of time passing and transition firing state changes can be converted into a sequence of two stage state changes where, in each state change, zero length or longer time passes and then a single transition fires.

Finally, we give the firing rule and the firing condition of time nets. For the firing of a transition to be possible at a certain time, four things must be taken in account. First, the transition must be enabled at the current marking. Second, the postplaces of the transition must be empty, or belong to both preset and postset of the transition. Third, at least the earliest firing time of the transition must have passed since its enabling. Last, no more time must have passed since the last firing than that which the latest firing times of all of the presently enabled transitions allow.

Definition 4 (firing condition) A transition t may fire from state $S = (M, I)$ after delay $\theta \in \mathbb{T}$ iff

$$t \in Enabled(M), \tag{4}$$
$$(M \setminus {}^\bullet t) \cap t^\bullet = \emptyset, \tag{5}$$
$$Eft(t) \leq I(t) + \theta, \text{ and} \tag{6}$$
$$\forall t' \in Enabled(M) : I(t') + \theta \leq Lft(t'). \tag{7}$$

A transition satisfying Equations 4,6 and 7 but not Eqn. 5 is said to be *in contact*. The set of all transitions that may fire from state S is denoted by $Fireable(S)$ and the set of all transitions in contact is $Contact(S)$. □

From the definition we see directly that $Fireable((M, I)) \subseteq Enabled(M)$. Not all enabled transition can fire because of timing constraints. The second requirement (Eqn. 5) protects from the insertion of a second token into a place, thus causing 1-safeness of the system. It will be made obsolete by the assumption of contact-freeness below.

The new marking after a firing is calculated in the usual way by removing tokens from the preset places of the firing transition and adding tokens to the postset places. New clock values are calculated for the transitions that remain enabled. Newly enabled transitions are assigned a zero clock value. The clock function has no value for disabled transitions.

Definition 5 (firing rule) When transition t *fires after time* θ from state $S = (M, I)$, the new state $S' = (M'', I')$ is given as follows:

$$M' = M \setminus {}^\bullet t, \tag{8}$$
$$M'' = M' \cup t^\bullet, \text{ and} \tag{9}$$
$$I'(t) = \begin{cases} I(t) + \theta & \text{if } t \in Enabled(M'), \\ 0 & \text{if } t \in Enabled(M'') \setminus Enabled(M'), \\ \text{undefined} & \text{else.} \end{cases} \tag{10}$$

□

In time Petri nets, the firing sequence is enriched with timing information, and it is called firing schedule [2].

Definition 6 (firing schedule) A *firing schedule* of a time Petri net is a finite or infinite sequence of pairs of transitions and time values

$$\sigma = (t_1, \theta_1), (t_2, \theta_2), (t_3, \theta_3), \ldots \tag{11}$$

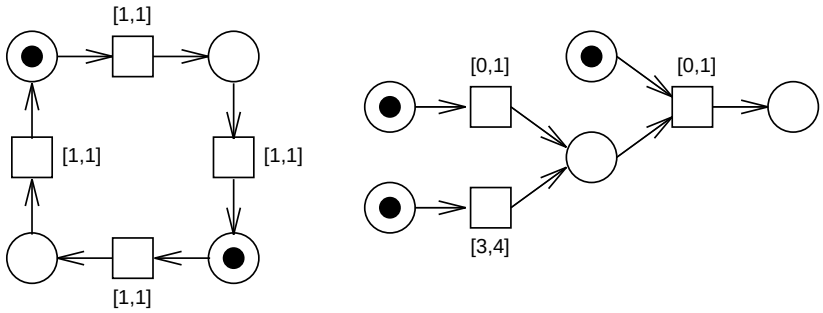

Figure 1: Contact-free time nets

where the t_i are transitions and $\theta_i \in \mathbb{T}$ are their *firing delays*. The firing schedule is *fireable* from the initial state S_0 if there exist states $S_1, S_2, S_3, \ldots$ such that the transition t_i may fire from state S_{i-1} (according to firing condition in Def. 4) and the firing leads to new state S_i (according to the firing rule in Def. 5) for $i = 1, 2, 3, \ldots$.

A state of a time Petri net is *reachable* if some fireable firing schedule leads from the initial state of the net to that state. A marking is reachable iff there is a reachable state with that marking. □

2.3 Boundedness

We want to exclude from our class of nets the ones where Eqn. 5, is needed to prevent a second token from being inserted to a place.

Definition 7 (contact-free) The time Petri net is *contact-free* iff in every reachable state S of the net, no transition is in contact, $Contact(S) = \emptyset$. □

A time Petri net can be contact-free even though the underlying elementary net system obtained by removing all timing information has contacts. On the other hand, contact-freeness of the elementary net system implies the contact-freeness of the time Petri net. The timing constraints can protect the net from reaching a contact state but they cannot create any new reachable contact states. In Fig. 1, there are two contact-free time nets whose corresponding elementary net systems have states with transitions in contact. In the contact-free time Petri nets, the latest allowed firing time for a transition forces the removal of the first token from a place before the second one is inserted to the place. Note also that tight loops do not create contacts.

Assumption 8 From now on, all time Petri nets will be assumed contact-free.

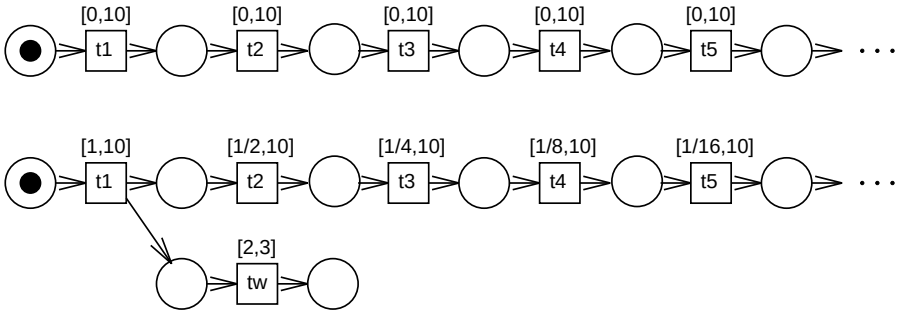

Figure 2: Two time Petri nets with nondivergent time

2.4 Zeno's paradox

Another anomaly that we need to exclude from the class of nets under consideration is the possibility of infinite firing schedules that consume no time. Moreover, in infinite nets there can be infinite firing schedules where transitions consume nonzero amounts of time, but the the total time is bounded by a finite constant.

Definition 9 (divergent time) The time Petri net TPN has *divergent time* iff for every infinite firing schedule $(t_1, \theta_1), (t_2, \theta_2), (t_3, \theta_3), \ldots$ of TPN, the series $\theta_1 + \theta_2 + \theta_3 + \cdots$ diverges. □

Figure 2 shows two time Petri nets that do not have divergent time. We will make some comments about the problems created by such degenerate timings in Section 4.2.

Assumption 10 From now on, we assume that all time Petri nets have divergent time unless otherwise stated.

3 Time processes

The firing schedule presentation of system behavior forces causally independent events in the system to a linear order. Since the whole purpose of Petri nets is to model the fine details of concurrency, a different presentation is needed, one that retains both causal dependence and concurrency. This is what processes are made for. We will now extend the notion of process to time constrained systems, modelled by time Petri nets.

Time processes of time Petri nets will be constructed by labeling traditionally defined causal processes with time values and giving validness criteria for them. Section 3.1 recalls some well-known definitions. In Section 3.2, a time process is defined as a causal process with a valid timing. An algorithm for deciding validness of a timing is presented in Section 3.3. Section 3.4 gives examples of valid and invalid timings. Section 3.5 discusses persistence of invalid timings in growing processes.

3.1 Causal processes

We recall the definitions of a causal net and homomorfism from the literature. The definition of homomorfism from time Petri nets to causal nets is a straightforward adaptation of the usual net homomorfism. A causal process of a time Petri net is defined as a causal net and a homomorfism between these nets. Timing information and constraints will be attached to the causal processes in Section 3.2.

Causal nets are used to depict causal histories of systems. The transitions of a causal net represent events in the system. An event changes the state of the system and it can only occur if certain conditions are fulfilled in the current state of the system. The preplaces of a transition represent the necessary preconditions for the event. Postplaces are the conditions that are made true by the event. The conditions can be thought of as logical or physical states that either are true or false (marked or unmarked). The flow relation of a causal net indicates causal relations between events. There are no loops in the net so that the one can talk about the past and the future of an event.

Definition 11 (causal net) A *causal net* $CN = (B, E, G)$ is a finitary, acyclic net, where

$$\forall b \in B : |b^\bullet| \leq 1 \wedge |{}^\bullet b| \leq 1.$$

Places B of a causal net are called *conditions* and transitions E are called *events*. Preplaces are called *preconditions* and postplaces *postconditions*. □

The term causal net is from [9]. Causal nets are called occurrence nets, for

example, in [13]. *Finitary* means that every $x \in B \cup E$ has only a finite number of G-predecessors.

If $CN = (B, E, G)$ is a causal net, we denote by $\leq$ the partial order G^* on $B \cup E$ and by $<$ the corresponding strict partial order. Event e is said to causally precede e' if $e \leq e'$. We write $Min(CN)$ for the $\leq$-minimal elements of a causal net and $Max(CN)$ for the $\leq$-maximal events, and call these *initial elements* and *final elements*, respectively.

$$Min(CN) = \{x \in B \cup E \mid \nexists y \in B \cup E : y < x\}, \tag{12}$$

$$Max(CN) = \{x \in B \cup E \mid \nexists y \in B \cup E : y > x\}. \tag{13}$$

In a nonempty net where G is finitary, there is always at least one minimal element. In a contact-free causal net, the minimal elements are places.

In a causal net, we identify the preset and postset of a condition with the unique events in them and use the notations ${}^\bullet b$ and $b^\bullet$ to mean a single event.

Defining processes, we give the conditions of causal nets the meaning that a place of a time Petri net is marked. Events represent firings of the transitions. The partial order of events gives the causal dependences between the firings. The initial conditions of the causal net precede all events in the causal partial order in the same way as the initial marking of a time Petri net is the initial cause of all transition firings in it.

The relation between time Petri nets and causal nets will be stated using a homomorfism, a mapping that preserves the local structure of the net. We define homomorfism as a mapping from a time Petri net to a causal net, because these are the net classes it will be used for. Nevertheless, this is the familiar net homomorfism of [32]. The formulation of the definition is similar to [9], but since we have not defined marking for a causal net, the minimal conditions will be used instead.

Definition 12 (homomorfism) Let $TPN = (P, T, F, SI, M_0)$ be a time Petri net and $CN = (B, E, G)$ a causal net. A mapping $p : B \cup E \longrightarrow P \cup T$ is a *homomorfism* if

$p(B) \subseteq P$,
$p(E) \subseteq T$,
$\forall e \in E$: the restriction of p to ${}^\bullet e$ is a bijection between ${}^\bullet e$ and ${}^\bullet p(e)$ and the restriction to $e^\bullet$ is a bijection between $e^\bullet$ and $p(e)^\bullet$, and the restriction of p to $Min(CN)$ is a bijection between $Min(CN)$ and M_0. □

In order to shorten the notation, we extend the domain of the homomorfism

p to sets of elements $X \subseteq B \cup E$, that is, $p(X) = \{p(x) \mid x \in X\}$. Especially for presets and postsets, ${}^\bullet p(e) = p({}^\bullet e)$ and $p(e)^\bullet = p(e^\bullet)$

We will frequently make use of an auxiliary function Cut [10]. In the context of a process, we write

$$Cut(E') = (E'^\bullet \cup Min(NS)) \setminus {}^\bullet E'. \tag{14}$$

The intuition of a cut is that it represents a state of the system. Thus, the function Cut should be used only on downward closed sets of events (sometimes called configurations). Otherwise, it does not make much sense.

The idea of a process in an untimed net system is that it is a possible "run" of the system in the same way as a firing sequence. But the process, unlike firing sequences, does not specify an interleaving of causally independent events. The entire causal history of the system is stored in the process, without additional information on the firing order of unrelated transitions. Nonsequential processes for Petri nets go as far back as [22]. Processes for untimed nets are defined in [13] as a mapping from a causal net to the original net. Different approaches to process construction are compared in [4]. Processes are sometimes called *concurrent runs* [26].

In time nets, the causal relationships are not the only way in which events influence each other. The time constraints of one event depend on the other events creating complex dependences between events and their timing. Also, occurrence times are an essential part of the history of system modelled by a time Petri net. We will define processes of time Petri nets, called time processes, by giving first an auxiliary definition of causal processes which have no timing information and whose structure is not constrained by the timing. In Section 3.2, the events of the causal process will be labeled with occurrence times and a validness criteria will be given to identify the labelings that correspond to actual runs of the system. Causal process is defined as a causal net and a homomorfism that relates it to the original net, in exactly the same way as processes of untimed nets have been defined, for example, in [9].

Definition 13 (causal process) Let $TPN = (P, T, F, SI, M_0)$ be a time Petri net. A *causal process* of TPN is a pair (CN, p), where

CN is a causal net and
p is a homomorfism from CN to TPN. □

It is often important to know if any two conditions in a set map to the same place of the original net. When all conditions of correspond to different places,

that is,

$$\forall b, b' \in B' : p(b) = p(b') \Rightarrow b = b', \tag{15}$$

we say that B' *maps injectively* to places. In processes of 1-safe untimed net systems, sets $Cut(E')$ are guaranteed to map injectively to places for all downward closed sets E'. This is not the case with contact-free time Petri nets. To see why, consider causal processes of the contact-free time Petri net in Fig. 1(a). The processes consist of two causally unrelated chains. It is possible to choose one condition from each chain so that they map to the same place of the net. In a process of an untimed net system, this would mean that the net is not 1-safe.

3.2 Valid timings

So far, we have listed concepts from the literature with the aim of forming a coherent set of basic definitions. We will now start building a new theory on top of them.

The notion of process will be extended to time Petri nets. This is done by adding timing information to the causal processes of the net. Time values depicting occurrence times are attached to the events of the process. Validness criteria are defined for deciding if the process with the time values is a *time process* of the time Petri net. The idea is that the time constraints imposed by the earliest and latest firing times of events restrict the set of possible timings the events of the time process can have. Not all timings are possible and some causal processes may even have no valid timings at all, being thus impossible in the system. The earliest and latest firing times put restrictions on the occurrence times of events in processes in the same way as they restrict the firing times in firing schedules. Furthermore, timing constrains the structure of the processes in the same way as it limits the number of reachable markings of a time Petri net.

There are not many references to combinations of timing and partial order semantics in the Petri net literature. [31] defines processes of timed Petri nets by an algebra of labeled partial orders. An alternative approach to processes of timed Petri nets, along the lines of [4], can be found in [30]. [14] introduces processes for a class of free choice nets where delays with upper and lower bounds are associated with the places of the net. In all the cases, timing is added on top of the processes of the underlying untimed net in the sense that the structure of the processes is not restricted by the timing. Such processes are useful in performance analysis, but they do not allow time constraints to limit the state space of the system.

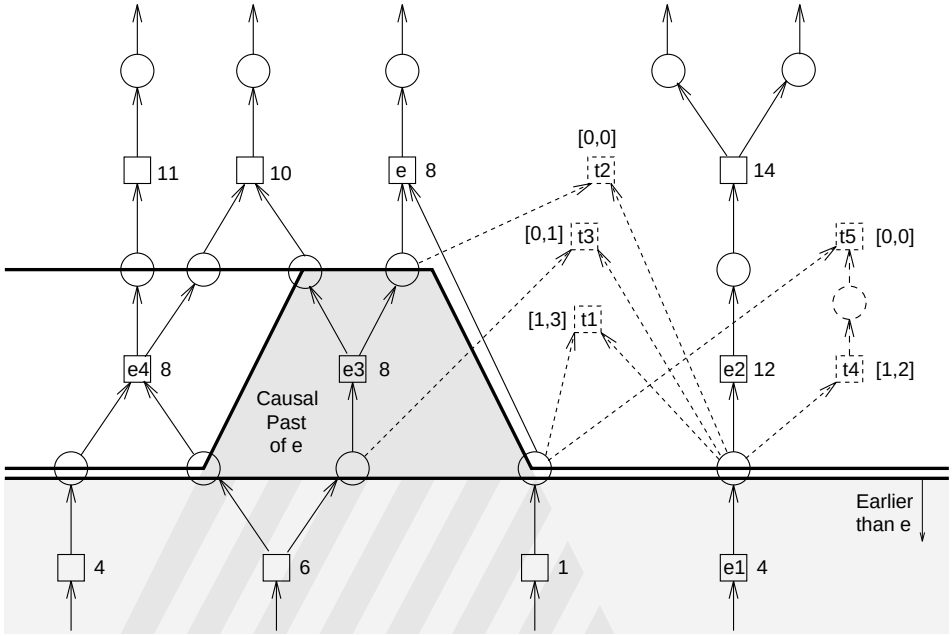

Figure 3: Events and occurrence times

In order to motivate the definition of valid timing, we demonstrate the dependences between between events of a time constrained system. Figure 3 shows a segment of a causal process of a time Petri net. Occurrence times are written next to the events. The dashed transitions are not part of the process, but they depict transitions enabled at certain times. Assume that the event e in the process is just about to occur. The darker grey area is the causal past of e, and the light grey area contains the events with an earlier occurrence time than e. The upper black line intuitively marks the two possible global states at which event e can occur depending on the interleaving of concurrent events. The lower black line marks the global state before the clock was advanced the last time. Let us denote the state of the system on this line by C_e. Below, we will argue that the possibility of event e and its occurrence time should be determined based on the state C_e rather than the causal past of e or the global state at which e occurs.

– We want to show how dependences arise between causally unrelated parts of the process. Look at transition t_1 enabled at the global state where e occurs. It became enabled at time 4 and has to fire before or at time $4 + 3 = 7$. Therefore, event e cannot occur at time 8. In this way, events e_1 and e_2 and their timing can affect e. The latest firing times of transitions that are not fired in the precess can either make it impossible for e to occur or just restrict its possible occurrence times. Thus, it is

not enough to look at the local state where e occurs, but one must also consider causally unrelated parts of the process in order to know if e can occur at the given time.

- This would indicate that we have to consider the global state where e occurs. However, the state is not uniquely known, as demonstrated by the branching upper black line in the figure.

- Even though t_2 is enabled at the global state at which e occurs, it cannot keep e from occurring. This is because one of the conditions enabling t_2 is not in C_e and, consequently, the latest firing time of t_2 cannot be earlier than the occurrence time of e. Thus, when assessing the possibility of event e, it is not necessary to consider all conditions in the global state where e occurs, but only those in C_e.

- Transition t_3 is enabled at C_e but not at any of the possible global states where e occurs. It can still stop e, because it does not allow e_3 to occur and e depends causally on e_3. In any case, it is just as important to know that event e_3 cannot occur at the given time than it is for e.

- Transition t_4 has to fire at time 6 or earlier. Unlike the other transitions, t_4 does not have causal relation to e through its preset. Therefore, one might think that t_4 cannot in any way affect e. But this is not true! t_4 does stop e because the next transition t_5 inevitably disables e. We might not know that there is a transition t_5, but to be on the safe side, we must require all transitions enabled at C_e to have latest possible firing times greater than or equal to the occurrence time of e. Otherwise, we cannot know that the occurrence time of e is possible.

- Events e, e_3 and e_4 have the same occurrence time and $C_e = C_{e_3} = C_{e_4}$. If we check that one of the events is not kept from firing by transitions enabled at C_e, it implies the same for all of them. Thus, all events with the same occurrence time can be processed together.

We will now give the definition of time process and continue with a related algorithm and examples in Sections 3.3 and 3.4. The entire Section 4 is devoted to showing that the validness criteria is solid in the sense that valid time processes have a particular relation to the firing schedules of the time Petri net.

Definition 14 (valid timing, time process) Let TPN be a time Petri net and (CN, p) its causal process, where $CN = (B, E, G)$. A *timing function* $\tau : E \longrightarrow \mathbb{T}$ is a function from events into time values. The values of τ are called *occurrence times* of the events. If B' is a set of conditions and transition

t is enabled at $p(B')$, the *time of enabling* for t in B' is defined as

$$TOE(B', t) = \max(\{\tau(^\bullet b) \mid b \in B' \setminus Min(CN) \wedge p(b) \in {}^\bullet t\} \cup \{0\}). \quad (16)$$

The set of *earlier events* for an event e is

$$Earlier(e) = \{e' \in E \mid \tau(e') < \tau(e)\}. \quad (17)$$

A timing function τ is a *valid timing* of the causal process iff

$$\forall e \in E : \tau(e) \geq TOE(^\bullet e, p(e)) + Eft(p(e)), \text{ and} \quad (18)$$

$$\forall e \in E : \forall t \in Enabled(p(C_e)) : \tau(e) \leq TOE(C_e, t) + Lft(t), \quad (19)$$

where $C_e = Cut(Earlier(e))$.

A *time process* of TPN is a triple (CN, p, τ) where (CN, p) is a causal process of TPN and τ is a valid timing of the causal process. □

The definition of valid timing has been derived from the firing condition of time Petri nets (Def. 4). The two criteria (Eqn. 18 and 19) impose the earliest and the latest firing times on the events (like Eqn. 6 and 7, respectively). The auxiliary function TOE gives the time when a transition becomes enabled at a set of conditions, i.e. the occurrence time of the last of the previous events. When there are no previous events (t is enabled at the initial marking), the time of enabling is naturally zero. The lower bounds of occurrence times (Eqn. 18) are easily checked as they depend only on the previous events. The upper bounds (Eqn. 19) are more complicated because they create dependences between causally unrelated parts of the process, as seen in the example above. The latest firing times of all transitions enabled at the set $C_e = Cut(Earlier(e))$ must be considered. The C_e has the same meaning as in the example.

The definition of valid timing does not give any direct way for constructing time processes. Instead, it is optimized for checking validness of known timing functions. Note that the set C_e is equal for all events e with the same value of the timing function,

$$\tau(e) = \tau(e') \Rightarrow C_e = Cut(Earlier(e)) = Cut(Earlier(e')) = C_{e'}. \quad (20)$$

Thus, Eqn. 19 has to be considered only once for each different occurrence time. Several events with the same occurrence time correspond to zero delays in firing schedules. For consecutive immediate firings, conformance to latest firing times needs to be checked only once. Consequently, firings with zero delay do not complicate the timing analysis with time processes. This is significant, because time constrained systems often have large parts that only

perform immediate events. Time Petri net models of such systems will have many transitions with static interval $[0, 0]$, and their processes will have large sets of events with equal timing values. When analyzing timing of the processes, we do not want the required effort to be increase if untimed parts of the system are described accurately in the model. The definition of validness has been formulated in this way to explicitly avoid unnecessary timing checks at parts of the process where time does not advance.

3.3 Algorithm for verifying validness

Figure 4 shows pseudo-code of an algorithm for deciding the validness of a timing function τ on a causal process (CN, p) of a time Petri net, where $CN = (B, E, G)$. The algorithm is based directly on the definition of valid timing. It checks the requirements of Equations 18 and 19 carefully avoiding extra work with events having the same occurrence time. In the following, we describe the algorithm and determine its complexity.

Checking that the validness criterion on the earliest firing times (Eqn. 18) holds is straightforward. It takes up to $\beta|E|$ steps, where β is the branching factor of the net.

In order to check that the criterion on latest firing times (Eqn. 19) holds, the algorithm enumerates all different sets $C_e = Cut(Earlier(e))$ with $e \in E$ and transitions enabled at them. The events are first sorted according to their occurrence times. The sorting takes up to $N \log N$ steps where $N = |\{\tau(e) \mid e \in E\}|$ is the number of different time values on the events.

A data structure CUT is needed for storing the sets C_e. It must be a set of conditions accessible in constant time by the corresponding places. This can be implemented as an array of size $|P|$, holding conditions and indexed by the places corresponding to the stored conditions.

The sets C_e are then constructed into the data structure CUT one after another by increasing values of $\tau(e)$. When a set C_e is known, the next set $C_{e'}$ is constructed by checking the conditions of C_e and postconditions of all events with occurrence time $\tau(e)$. The latter are easily obtained from the sorted set E. The conditions whose next events also have occurrence time $\tau(e)$ are discarded; the others belong to $C_{e'}$ and they are reinserted into CUT. In this way, the conditions of $C_{e'}$ are collected one by one. There are N different sets C_e to go through. In each set C_e, there are up to P conditions to be processed. This means inserting at most $N|P|$ conditions to CUT during the execution. The transitions enabled at the sets C_e are enumerated in the following way. Every time a new condition belonging to C_e is found, the algorithm checks if it enables any transition with the other conditions already in CUT. The

```
global variables CUT, NEW, time, limit
EV : array 1...|E| of events
CUT : set of conditions, accessed by corresponding places
NEW : set of conditions

(* Eft *)
for each e ∈ E do
      for each e' ∈ ••e do
            if τ(e) < τ(e') + Eft(p(e)) then return false
(* Lft *)
sort E by τ(e) and insert into EV[1...|E|]
i := 0, time := 0, CUT := ∅
repeat
      NEW := CUT, CUT := ∅, limit := ∞
      call add_new_to_cut
      repeat
            if i = 0 then NEW := Min(CN) else NEW := EV[i]•
            call add_new_to_cut
            i := i + 1
      until i = |E| + 1 or τ(EV[i]) > time
      if i ≤ |E| then time := τ(EV[i])
      (* Now CUT = Cut(Earlier(EV[i])) and Eqn. 19 is time ≤ limit,
         or i = |E| + 1 and we are done *)
until i = |E| + 1 or time > limit
return (i = |E| + 1)

procedure add_new_to_cut
      for each b ∈ NEW do
            if b• ≠ ∅ or τ(b•) > time then
                  for each t ∈ p(b)• do
                        if •b = ∅ then toe := 0 else toe := τ(•b)
                        for each s ∈ •t do
                              if s ≠ p(b) then
                                    find b' ∈ CUT such that s = p(b')
                                    if such b' exists then
                                          if •b' ≠ ∅ then toe := max(toe, τ(•b'))
                                    else
                                          toe := ∞
                                          break
                        limit := min(limit, toe + Lft(t))
                  insert b into CUT
```

Figure 4: Checking the validness of a timing

new condition is then inserted into CUT. The new condition has at most β transitions possibly enabled by it. Enabledness of each of them is checked by looking for the at most $\beta - 1$ other preplaces in CUT. This means less than β^2 constant time accesses to CUT for each new condition. Thus, during the entire execution, inserting conditions into CUT takes at most $\beta^2 N|P|$ steps.

Altogether, the sorting and the rest of the computation take up to $N \log N + \beta^2 N|P|$ steps.

Theorem 15 The validness of a timing function on a causal process can be checked in time $O(N \log N + \beta^2 N|P|)$, where N is the number of different values of the timing function in the process, β is the branching factor and $|P|$ the number of places in the net. □

Naturally, if the set of events is already sorted by the timing, validness can be decided in $O(\beta^2 N|P|)$ steps. For causal processes that have been generated by time Petri nets much smaller than the process, the sorting is the most expensive part of the computation. On the other hand, if the process is about the same size as the net, the $\beta^2 N|P|$ will dominate. When there are many events with equal time values, the algorithm will do significantly less work than when all events have unique occurrence times, because events with equal time are processed together.

3.4 Example

In Figure 5, there is a time Petri net and its time process. ($\mathtt{w} = \infty$.) The values of the timing function have been printed next to the events. The timing in Fig. 5(b) is valid. All different sets $Cut(Earlier(e))$ are shown in grey. They are only three such sets because two are equal: $\tau(e_2) = \tau(e_3)$ implies $Cut(Earlier(e_2)) = Cut(Earlier(e_3))$. The transitions enabled at the sets have been drawn with dashed lines.

On the contrary, the timings in Figure 6 are not valid. The sets and enabled transitions that conflict with Eqn. 19 are shown. In (a), transition t_2 should fire before e_3 and disable it. The timing is clearly not in accordance with the firing condition. Net (b) demonstrates a more surprising requirement for validness. In Fig. 5(b) we already saw that the firing times of the transitions are completely legal. Nevertheless, the timing is not valid. This is because the left side of the process has not been generated far enough to make conclusions about the possibility of the firing times. We will see in Section 5 that it is essential for all parts of the process to be complete up to same time value. Otherwise, validness cannot be determined.

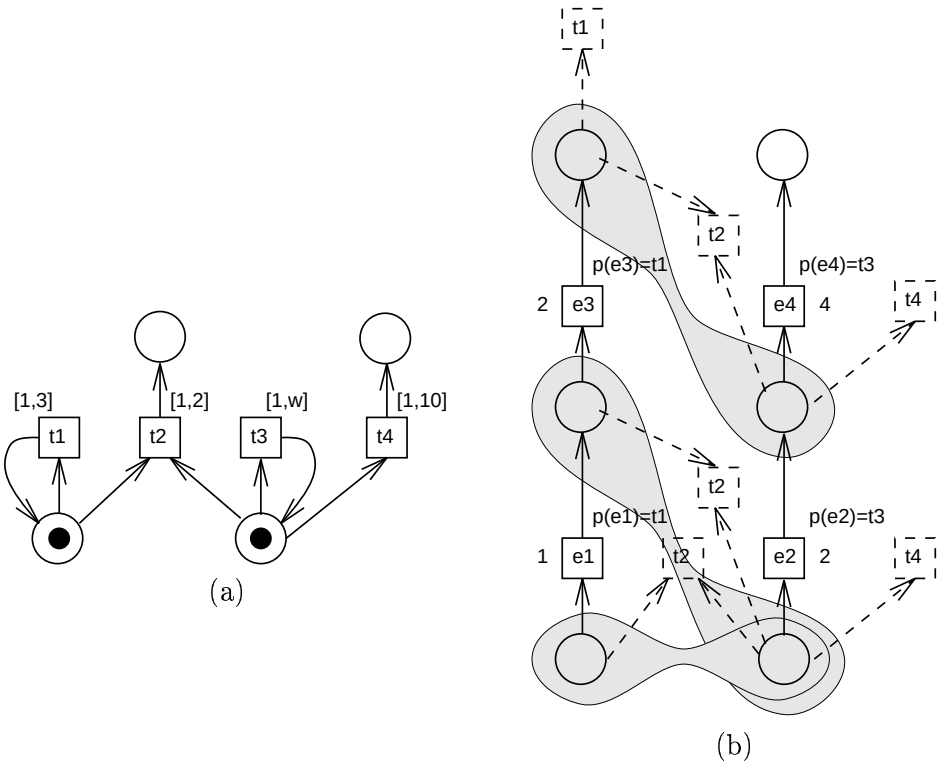

Figure 5: A time Petri net and a process with valid timing

3.5 Invalid timings and process growth

In the preceding example we saw that an invalid timing can become valid when the process grows. The question arises, when can one be certain that a timing cannot be made valid by generating the process further. To answer this, we must first define when a process is larger than another. The following definition is essentially same as in [13]; only the τ has been added.

Definition 16 (initial subprocess) Let (CN', p') and (CN, p) be causal processes of a time Petri net, where the causal nets are $CN = (B, E, G)$ and $CN' = (B', E', G')$. (CN', p') is an *initial subprocess* of (CN, p) iff $B' \cup E'$ is a downward closed (with respect to G) subset of $B \cup E$ and G' and p' are restrictions of G and p to $B' \cup E'$. □

The following theorem seems more complicated than it really is. It simply states that invalidity persists in growing processes if a new event cannot be added to a location where it would remedy the invalid situation. This idea will be restated in a more comprehensible form in Section 5. We delay the

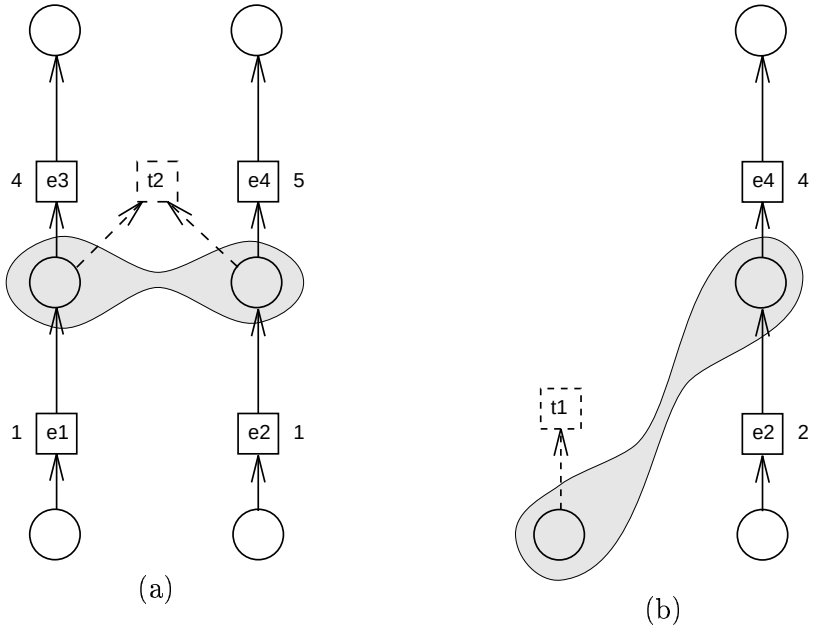

Figure 6: Two invalid timings

proof until then.

Theorem 17 Let a causal process (CN, p) with timing function τ have an event e and a transition t that violate the inequality of Eqn. 19. Moreover, assume $|b^\bullet| = 1$ for all $b \in Cut(Earlier(e)) \cap p^{-1}({}^\bullet t)$. If (CN, p) is an initial subprocess of another process $(\hat{CN}, \hat{p})$, then $(\hat{CN}, \hat{p})$ does not have any valid timing such that τ is a restriction of that timing function to the events of $\hat{CN}$. □

The process in Figure 6(a) fulfills the prerequisites of the theorem. Event e_3 and transition t_2 violate Eqn. 19 and all conditions enabling t_2 are consumed by some event. Thus, no time process having the process as an initial subprocess can have the same timing of events $e_1, \ldots, e_4$ as in the figure. On the other hand, Theorem 17 does not apply to the process of Fig. 6(b). As we have seen in Fig. 5(b), the timing is valid in a larger process.

4 Processes and firing schedules

In this section, we will look closely at the relation between time processes and firing schedules. A time process is a partial order that can be sorted into several different linear orders. The linear orders that respect the occurrence times of the events will be called interleavings of the time process. We will see that in a time Petri net, there is one-to-one correspondence between interleavings of time processes and firing schedules that are fireable from the initial state of the net.

Interleavings of a time process and a function from interleavings to firing schedules are introduced in Section 4.1. In Section 4.2, these are shown to be an essential link between the world of time processes and that of firing schedules.

4.1 Interleavings

We want to define interleaving of a time process as a linearization of the partial order of events where the events are ordered also by their occurrence times. In order for this to make sense, we have to prove that the causal and time order never conflict with each other.

Lemma 18 Let (CN, p, τ) be a time process of a time Petri net, where $CN = (B, E, G)$. Let also $e, e' \in E$ be two events of the process. Then,

$$e \leq e' \;\Rightarrow\; \tau(e) \leq \tau(e') \tag{21}$$

□

Proof Recall that $e \leq e'$ means $e\, G^*\, e'$, i.e. $e = e_1\, G\, b_1\, G\, e_2\, G\, b_2\, G \ldots G\, e_n = e'$ for some $e_i \in E;\ i = 1, \ldots, n$ and $b_i \in B;\ i = 1, \ldots, n-1$. From the definition of valid timing we get $e_i < e_{i+1} \;\Rightarrow\; \tau(e_{i+1}) \geq \tau(e_i) + Eft(p(e_{i+1}))$. Since the lower boundaries of the static intervals are nonnegative, the sequence $\tau(e_1), \ldots, \tau(e_n)$ is nondecreasing, and $\tau(e) \leq \tau(e')$. □

It is now possible to give the definition of an interleaving of a time process. A function from interleavings to firing schedules is will also be defined. Interleavings and the function will link time processes to firing schedules.

Definition 19 (interleaving) An *interleaving* of a time process is a finite or infinite sequence $\rho = e_1, e_2, e_3, \ldots$ consisting of the events of the process, such that every event is in the sequence exactly once, and both causal and time order are preserved,

$$(e_i < e_j \;\vee\; \tau(e_i) < \tau(e_j)) \;\Rightarrow\; i < j \text{ for all } i, j. \tag{22}$$

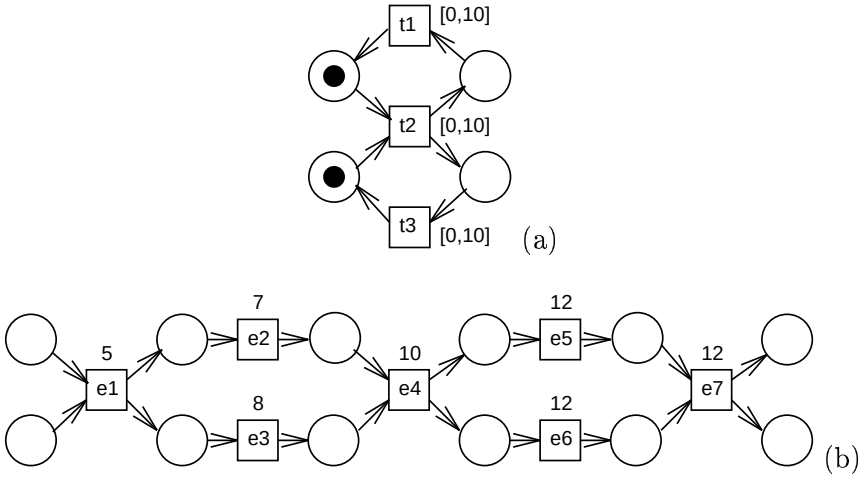

Figure 7: A process with two interleavings

The function FS maps interleavings to firing schedules of the net.

$$FS(\rho) = (p(e_1), \tau(e_1) - 0),\ (p(e_2), \tau(e_2) - \tau(e_1)),\ \ldots \tag{23}$$

□

To summarize, we have now three different order relations on the set of events: causal order ($e < e'$), time order ($\tau(e) < \tau(e')$) and a linear order in an interleaving ($e_i < e_{i+1}$). The orders do not conflict with each other. Causal and time order both imply order in interleavings.

In Figure 7, there is a time Petri net (a) and it process (b). The values of the timing function τ have been marked next to the events. The process has two interleavings,

$$\rho = e_1, e_2, e_3, e_4, e_5, e_6, e_7 \text{ and}$$
$$\rho' = e_1, e_2, e_3, e_4, e_6, e_5, e_7.$$

The order of e_2 and e_3 is fixed because $\tau(e_2) < \tau(e_3)$, but e_5 and e_6 can be interleaved arbitrarily because they have the same occurrence time and no causal relationship. The number of interleavings of a time process can be much smaller than it would be for the causal process, because the occurrence times determine the linear order. The firing schedules corresponding to the above interleavings are

$$FS(\rho) = (t_2, 5), (t_1, 2), (t_3, 1), (t_2, 2), (t_1, 0), (t_3, 0), (t_2, 0) \text{ and}$$
$$FS(\rho') = (t_2, 5), (t_1, 2), (t_3, 1), (t_2, 2), (t_3, 0), (t_1, 0), (t_2, 0).$$

4.2 Relation between interleavings and firing schedules

In the rest of this section, we will prove that the function FS is a bijective mapping from the interleavings of the processes of a time Petri net to the firing schedules fireable from the initial state of the net. It will be shown first that the firing schedule $FS(\rho)$ for an interleaving ρ is fireable. The proof of that theorem is then further utilized in showing that every process has at least one interleaving. After this sidestep, the mapping FS is shown to be surjective. That is, every fireable firing schedule is the image of some interleaving. The proof is done by constructing a time process from the firing schedule. The construction is unique up to renaming of process elements, meaning that the relation is bijective.

Throughout the rest of this section, having first fixed an interleaving $\rho = e_1, e_2, e_3, \ldots$, we write ρ_k for its prefix of length k. For each $k = 0, 1, 2, \ldots$, we write E_k for the set of events in ρ_k, $E_k = \{e_i \in E \mid i \leq k\}$. Specifically, $E_0 = \emptyset$.

Before stating the main theorems of this section, we need to prove some technical details.

Lemma 20 Let ρ be an interleaving of a time process of a time Petri, e_{k+1} its kth element, t a transition of the net, and E_k as defined above. If $Cut(E_l)$ maps injectively to places for all $0 \leq l \leq k$, the following formulas hold for E_k.

$$p(e_{k+1}) \in Enabled(Cut(E_k)). \tag{24}$$

$$TOE({}^\bullet e_{k+1}, p(e_{k+1})) = TOE(Cut(E_k), p(e_{k+1})) \tag{25}$$

$$TOE(Cut(Earlier(e_{k+1})), t) \leq TOE(Cut(E_k), t). \tag{26}$$

□

Proof Let $b \in {}^\bullet e_{k+1}$. Either $b \in Min(CN)$ or ${}^\bullet b < e_{k+1}$. In the latter case, ${}^\bullet b$ must be before e_{k+1} in the interleaving ρ, ${}^\bullet b \in E_k$. Thus, $b \in E_k{}^\bullet \cup Min(CN)$. On the other hand, $b^\bullet = e_{k+1} \notin E_k$, and $b \notin {}^\bullet E_k$. Putting these together, we get ${}^\bullet e_{k+1} \subseteq (E_k{}^\bullet \cup Min(CN)) \setminus {}^\bullet E_k = Cut(E_k)$. It follows that $p({}^\bullet e_{k+1}) \subseteq p(Cut(E_k))$. Since the homomorfism p is an isomorfism on the presets, ${}^\bullet(p(e_{k+1})) \subseteq p(Cut(E_k))$, i.e. $p(e_{k+1})$ is enabled at $p(Cut(E_k))$. Thus, Eqn. 24 holds.

$Cut(E_k)$ maps injectively to places. Let $b \in Cut(E_k)$, and $p(b) \in {}^\bullet p(e_{k+1})$. Assume that $b \notin {}^\bullet e_{k+1}$. This means that there is $b' \in Cut(E_k)$ such that $p(b') = p(b)$ but $b' \neq b$, which contradicts with the injective property. Thus, the assumption is wrong and b must be in ${}^\bullet e_{k+1}$. This and the definition of

TOE imply $TOE(Cut(E_k), p(e_{k+1})) \leq TOE({}^\bullet e_{k+1}, p(e_{k+1}))$. On the other hand, it is obvious from Eqn. 24 and the definition of TOE that $TOE({}^\bullet e_{k+1}, p(e_{k+1})) \leq TOE(Cut(E_k), p(e_{k+1}))$. Thus, we have Equation 25.

$Earlier(e_{k+1}) = E_l \subseteq E_k$ for some $0 \leq l \leq k$. For that l, $\tau(e) = \tau(e_{k+1}) \geq TOE(Cut(E_l), t)$ for all the events $e \in E_k \setminus E_l = \{e_{l+1}, \ldots, e_k\}$. $Cut(E_l)$ and $Cut(E_k)$ map injectively to places. Let $t \in Enabled(Cut(Earlier(e_{k+1}))) \cap Enabled(Cut(E_k))$. There has to be a subset $B' \subseteq Cut(Earlier(e_{k+1}))$ such that $p(B') = {}^\bullet t$, and a similar subset B'' in $Cut(E_k)$. The injective mapping to places guarantees that B' and B'' are unique. Consider first the case where $B' = B''$. Then, $TOE(Cut(E_l), t) = TOE(Cut(E_k), t)$. On the other hand, if $B' \neq B''$, there is a condition $b \in B'' \setminus B'$ for which ${}^\bullet b \in E_k \setminus E_l$. Then, it must be the case that $TOE(Cut(E_l), t) \leq TOE(Cut(E_k), t) = \tau({}^\bullet b)$. In both cases, Eqn. 26 holds. □

We now have the prerequisites to prove that interleavings of a time process are mapped to firing schedules that are fireable from the initial state of the net. This is the first important property of FS.

Theorem 21 If ρ is an interleaving of a time process of a time Petri net, then the firing schedule $FS(\rho)$ is fireable from the initial state of the net. The markings in the intermediate states of the firing schedule are $M_k = p(Cut(E_k))$ for $k = 1, 2, 3, \ldots$. □

Proof Let $TPN = (P, T, F, SI, M_0)$ be a time Petri net, (CN, p, τ) its time process and $\rho = e_1, e_2, e_3 \ldots$ an interleaving of the time process. The proof is done by induction on the prefixes of the interleaving.

Our goal is to prove that the firing schedule $FS(\rho_k)$ is fireable from the initial state S_0 of TPN. Throughout the induction, we need to carry three additional conditions, which relate the marking, contact-freeness and timing in the firing schedule to the respective properties of the time process. First, the marking after firing k first transitions of the firing schedule is given by

$$M_k = p(Cut(E_k)). \tag{27}$$

Second, the set of conditions $Cut(E_l)$ maps injectively to places for all $0 \leq l \leq k$ in the sense of Eqn. 15. Even though these two conditions obviously hold in processes of contact-free net systems, we cannot assume them without explicit proof in contact-free time Petri nets, whose underlying net systems are not necessarily contact-free. Last, the value of the clock function for all $t \in Enabled(M_k)$ is

$$I_k(t) = \tau(e_k) - TOE(Cut(E_k), t) \leq Lft(t). \tag{28}$$

if $k > 0$. Let $\tau(e_0)$ denote 0 in the rest of the proof, and let $\tau(e_k)$ have its standard meaning when $k > 0$. (This is just a syntactic trick to shorten the

notation. On need not worry what $\tau(e_0)$ actually means.) Then, the above equation holds for all $k \geq 0$.

Basis step: The empty firing schedule is a prefix of ρ. The initial marking of the net satisfies Eqn. 27 since $p(Cut(E_0)) = p(Cut(\emptyset)) = p(Min(CN)) = M_0$. The initial marking is a set of places and p is a bijection between it and $Min(CN)$. Therefore, no two events in $Cut(E_0)$ map onto the same place, as required by the injective property. The initial value of the clock function for all $t \in Enabled(M_0)$ is $I_0(t) = 0 < Lft(t)$, and it is undefined for other transitions.

Induction step: Assume that the firing schedule $FS(\rho_k)$ is fireable from S_0, that Equations 27 and 28 hold for the index k, and that $Cut(E_l)$ maps injectively to places for all $0 \leq l \leq k$.

The first goal to show that the firing schedule $FS(e_{k+1})$ is fireable from S_0. We begin by noting that $p(e_{k+1})$ is enabled at the marking M_k. Since $M_k = p(Cut(E_k))$, this follows directly from Eqn. 24.

It is not sufficient that $p(e_{k+1})$ is enabled, but it must also be fireable. Def. 4 requires that

$$Eft(p(e_{k+1})) \leq I_k(p(e_{k+1})) + \tau(e_{k+1}) - \tau(e_k) \text{ and} \tag{29}$$

$$\forall t \in Enabled(M_k) : I_k(t) + \tau(e_{k+1}) - \tau(e_k) \leq Lft(t). \tag{30}$$

We first examine the earliest firing time. Rearranging Eqn. 29, substituting the value of $I_k(p(e_{k+1}))$ from Eqn. 28 and applying Eqn. 25, we get a new inequality that is equivalent with Eqn. 29.

$$\begin{aligned} \tau(e_{k+1}) &\geq \tau(e_k) - I_k(p(e_{k+1})) + Eft(p(e_{k+1})) \\ &= TOE(Cut(E_k), p(e_{k+1})) + Eft(p(e_{k+1})) \\ &= TOE({}^\bullet e_{k+1}, p(e_{k+1})) + Eft(p(e_{k+1})). \end{aligned} \tag{31}$$

This inequality is immediate from the definition of valid timing (Eqn. 18). Thus, the criterion on the earliest firing time (Eqn. 29) is satisfied.

The criterion on the latest firing time (Eqn. 30) requires a little more work. Let $t \in Enabled(p(Cut(E_k)))$. We have to show that $I_k(t) + \tau(e_{k+1}) - \tau(e_k) \leq Lft(t)$. Substituting the value of $I_k(t)$, this becomes $\tau(e_{k+1}) \leq TOE(Cut(Earlier(e_{k+1}))) + Lft(t)$. But the definition of valid timing (Eqn. 19), and Lemma 20 give

$$\begin{aligned} \tau(e_{k+1}) &\leq TOE(Cut(Earlier(e_{k+1})), t) + Lft(t) \\ &\leq TOE(Cut(E_k), t) + Lft(t). \end{aligned} \tag{32}$$

Thus, the criterion on the latest firing time is also satisfied. Not only is the transition $p(e_{k+1})$ enabled after $FS(\rho_k)$, but it is also a fireable. Consequently, the firing schedule $FS(\rho_{k+1})$ is fireable from S_0.

Next, we need to show that the marking after $FS(\rho_{k+1})$ is correctly obtained from Eqn. 27 and that $Cut(E_{k+1})$ is maps injectively to places.

$$\begin{aligned} Cut(E_{k+1}) &= ({E_{k+1}}^\bullet \cup Min(NS)) \setminus {}^\bullet E_{k+1} \\ &= ({E_k}^\bullet \cup {e_{k+1}}^\bullet \cup Min(NS)) \setminus ({}^\bullet E_k \cup {}^\bullet e_{k+1}) \\ &= (((E_k{}^\bullet \cup Min(NS)) \setminus {}^\bullet E_k) \setminus {}^\bullet e_{k+1}) \cup {e_{k+1}}^\bullet \\ &= (Cut(E_k) \setminus {}^\bullet e_{k+1}) \cup {e_{k+1}}^\bullet. \end{aligned} \tag{33}$$

The restriction of p to a preset or a postset is a bijection, and because of the injective mapping of $Cut(E_k)$, the restriction of p to $Cut(E_k)$ is also a bijection. Hence, we have

$$M_{k+1} = p(Cut(E_{k+1})) = (p(Cut(E_k)) \setminus {}^\bullet p(e_{k+1})) \cup p(e_{k+1})^\bullet. \tag{34}$$

This coincides with the firing rule. Thus, the M_{k+1} given by Eqn. 27 equals the marking after $FS(\rho_{k+1})$.

Similarly, the bijections can be used to derive the following formula. When the net is contact-free, the right hand side equals the empty set (Def. 7).

$$p(Cut(E_k) \setminus {}^\bullet e_{k+1}) \cap p({e_{k+1}}^\bullet) = (M_k \setminus {}^\bullet p(e_{k+1})) \cap p(e_{k+1})^\bullet = \emptyset. \tag{35}$$

$Cut(E_k) \setminus {}^\bullet e_{k+1}$ maps injectively to places, because $Cut(E_k)$ does. The above equation shows that $p(Cut(E_k) \setminus {}^\bullet e_{k+1})$ is disjoint with $p({e_{k+1}}^\bullet)$. Thus, $Cut(E_{k+1}) = (Cut(E_k) \setminus {}^\bullet e_{k+1}) \cup {e_{k+1}}^\bullet$ maps injectively to places.

It remains to be shown that the timing function I after the $(k+1)$th firing is the I_{k+1} given by Eqn. 28. For those transitions t_{old} that remain enabled in the firing of $p(e_{k+1})$,

$$TOE(Cut(E_{k+1}), t) = TOE(Cut(E_k), t). \tag{36}$$

To see this, change $Earlier(e_{k+1})$ into E_{k+1} in the proof of Eqn. 26 of Lemma 20. When we know that t remains enabled, it is always the case that $B' = B''$ for the sets in the proof. Thus the equality in Eqn. 36. In the $(k+1)$th firing, the timing function becomes

$$\begin{aligned} I_{k+1}(t_{old}) &= \tau(e_{k+1}) - TOE(Cut(E_{k+1}), t) \\ &= \tau(e_{k+1}) - TOE(Cut(E_k), t) \\ &= I_k(t_{old}) + \tau(e_{k+1}) - \tau(e_k). \end{aligned} \tag{37}$$

The second equality above holds because of Eqn. 36, and the third is obtained simply by substituting the value of $I_k(t_{old})$ from Eqn. 28. From Eqn. 32 we see immediately that $I_{k+1}(t_{old}) \leq Lft(t_{old})$.

On the other hand, if t is newly enabled by the $(k+1)$th firing,

$$TOE(Cut(E_{k+1}), t) = \tau(e_{k+1}). \tag{38}$$

The above equation holds, because when t is newly enabled, there is necessarily a $b \in e_{k+1}{}^{\bullet} \subseteq Cut(E_{k+1})$ such that $p(b) \in {}^{\bullet}t$. As e_{k+1} has the highest value of τ in E_{k+1}, this implies $TOE(Cut(E_{k+1}), t) = \tau(e_{k+1})$, Eqn. 38. The timing function after the $(k+1)$th firing is

$$\begin{aligned} I_{k+1}(t_{new}) &= \tau(e_{k+1}) - TOE(Cut(E_{k+1}), t) && (39)\\ &= \tau(e_{k+1}) - \tau(e_{k+1}) = 0 \quad \leq Lft(t_{new}), && (40) \end{aligned}$$

where the second equality comes from Eqn. 38. Consequently, the value of I_{k+1} is in accordance with the firing rule.

We have shown that if the induction hypothesis holds for index k, it also holds for $k+1$. By induction we now know that it is true for any index. Especially, for any prefix ρ_i of an interleaving, the firing schedule $FS(\rho_i)$ is fireable from the initial state of the net. This suffices to show that any interleaving of finite or infinite length is mapped onto a fireable firing schedule. The marking after the kth firing in the schedule is $M_k = p(Cut(E_k))$. This concludes the proof of Theorem 21. □

We know now that there is a fireable firing schedule corresponding to every interleaving. Before showing more properties of the function FS, we sidestep into the relation between processes and interleavings. It is intuitively clear that a process should be completely characterized by its interleavings, but since an interleaving has only an enumerable amount of events, some processes might be too large to interleave. This is why we have limited the branching factor of the net. Furthermore, we have limited the discussion to systems with divergent time. Otherwise, not all processes would have interleavings. This is demonstrated by the time Petri net in Figure 2(b). The net has a process isomorfic with itself, pictured with occurrence times of events in Fig. 8. The time values leave only one possible interleaving, $e_1, e_2, e_3, \ldots$. But this interleaving does not have the event e_0 (with $\tau(e_0) = 2$) in it, because there are infinitely many events with time less than 2. The assumption of divergent time saves us from situations like this, as will be shown in the following.

Theorem 22 Let TPN be a time Petri net with divergent time. Every time process of TPN has an interleaving. □

Proof Let (CN, p, τ) be a process of a time Petri net, where the causal net is $CN = (B, E, G)$. The minimal events of CN are $Min(CN)^{\bullet}$. This set has at most $|Min(CN)| \cdot \beta = |M_0| \cdot \beta < \infty$ members. Thus, the set of minimal events in CN is finite. If one minimal event e_{min} is removed from E, the

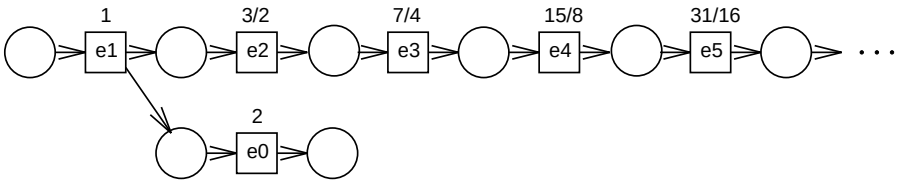

Figure 8: A process with nondivergent timing

minimal events of the resulting partial order either were already minimal or are in the finite set $e_{min}{}^{\bullet\bullet}$. Thus, after removal of e_{min}, the set of minimal events is still finite. By induction, the set of minimal events remain finite no matter how many minimal events are removed one after another.

Since the set of $\leq$-minimal events E_{min} is finite, there is always at least one event e_{min} in E_{min} that has the smallest time value $\tau(e_{min})$. A sequence of events $\rho = e_1, e_2, \ldots$ can be constructed by selecting for e_i a $\leq$-minimal event that has a minimal time value, removing it from the partial order, and proceeding with the selection of event e_{i+1} in the same way.

The constructed sequence ρ has the property that for any i, all events of E that causally or in time precede e_i belong to $E_i = \{e_1, \ldots, e_i\}$. This property is exactly Eqn. 22 in the definition of interleaving. The only thing that remains to show is that all events E are in the sequence ρ. Obviously, if $|E| < \infty$, every event of E will eventually be chosen for some e_k. For an infinite process, the proof is done by contradiction.

Assume that an event $e \in E$ is not in the sequence ρ. e has some time value $\tau(e)$. The sequence ρ satisfies Eqn. 22 which was the only property of interleaving that was actually used in the proof of Lemma 20 and, thereafter, was needed for Theorem 21. Hence, also the firing sequence $FS(\rho)$ is fireable from S_0. If the time Petri net has divergent time, the series

$$(\tau(e_1) - 0) \; + \; (\tau(e_2) - \tau(e_1)) \; + \; (\tau(e_3) - \tau(e_2)) \; + \; \ldots \qquad (41)$$

is divergent. This means that $\tau(e_i) \to \infty$ when $i \to \infty$. For some e_i in ρ, the time $\tau(e_i)$ will go over $\tau(e)$. Thus, the assumption is wrong, every event of E is in ρ, and ρ is an interleaving. □

The firing schedules in the range of the function FS are known to be fireable from the initial state of the time Petri net in question. We still have to show that the function is a bijection. The next theorem will show that it is surjective.

Theorem 23 Given a firing schedule σ of a time Petri net TPN, it is possible to construct a process of TPN with interleaving ρ, such that $\sigma = FS(\rho)$. □

Proof Let $TPN = (P, T, F, SI, M_0)$ be a time Petri net and let (t_1, θ_1), $(t_2, \theta_2), (t_3, \theta_3), \ldots$ be a firing schedule. We construct from the schedule a time process (CN, p, τ), where $CN = (B, E, G)$, and prove step by step that it, indeed, is a time process. After that, it will be easy to see that the process has the desired interleaving.

$$E = \{e_1, \ldots, e_n\}, \tag{42}$$

$$p(e_i) = t_i \text{ for all } i = 1, \ldots, n, \tag{43}$$

$$\tau(e_i) = \Sigma_{k=1}^{i} \theta_k, \ i = 1, \ldots, n. \tag{44}$$

$$B = \{b_{0,s} \mid s \in M_0\} \cup \{b_{i,s} \mid i \geq 1 \wedge s \in t_i^\bullet\}, \tag{45}$$

$$p(b_{i,s}) = s \text{ for all } i = 1, \ldots, n. \tag{46}$$

$$\begin{aligned} G = &\{(e_i, b_{i,s}) \mid i \geq 1 \wedge s \in t_i^\bullet\} \\ &\cup \{(b_{j,s}, e_i) \mid s \in {}^\bullet t_i \wedge j = \max\{k \mid b_{k,s} \in B \wedge k < i\}\}. \end{aligned} \tag{47}$$

In order to show that (CN, p, τ) is a time process of TPN, we have to prove that CN is a causal net, (CN, p) is a process, and τ is its valid timing. Clearly, CN is a net. It is acyclic because $e_i \, G \, b_{i,s} \, G \, e_j$ implies $i < j$, and inductively, $e_i < e_j$ implies $i < j$. It is also finitary, because for every e_j there are at most $j - 1$ events $e_k < e_j$. The preset of a condition is ${}^\bullet b_{i,s} = \{e_i\}$ for $i \geq 1$ and ${}^\bullet b_{0,s} = \emptyset$. Thus, $|{}^\bullet b| \leq 1$ for all $b \in B$.

To see that CN is a causal net, we still have to verify that $|b_{j,s}^\bullet| \leq 1$ for all conditions in B. Let $b_{j,s} \in B$. For the above chosen set B of conditions, $b_{j,s} \in B$ means $(j = 0 \wedge s \in M_0) \vee (j \geq 1 \wedge s \in {}^\bullet t_j)$. Assume $e_i \in b_{j,s}^\bullet$ and $e_{i'} \in b_{j,s}^\bullet$, where $i' < i$. In the chosen flow relation G, $e_i \in b_{j,s}^\bullet$ means

$$s \in {}^\bullet t_i \ \wedge \ b_{j,s} \in B \ \wedge \ j < i \ \wedge \ \nexists k : (j < k < i \ \wedge \ b_{k,s} \in B), \tag{48}$$

which equals

$$\begin{aligned} s \in {}^\bullet t_i \ \wedge \ ((j = 0 \ \wedge \ s \in M_0) \ \vee \ (j \geq 1 \ \wedge \ s \in {}^\bullet t_j)) \ \wedge \\ j < i \ \wedge \ \nexists k : (j < k < i \ \wedge \ s \in t_k^\bullet). \end{aligned} \tag{49}$$

The same holds for i'. $s \in {}^\bullet t_{i'} \wedge j < i' < i$. Together, these imply

$$s \in {}^\bullet t_{i'} \ \wedge \ s \in {}^\bullet t_i \ \wedge \ i' < i \ \wedge \ \nexists k : (i' < k < i \ \wedge \ s \in t_k^\bullet). \tag{50}$$

But such a firing schedule is not possible: $t_{i'}$ cannot consume token s if none has been produced after t_i. Therefore, i must equal i' and $|b_{j,s}^\bullet| \leq 1$ for all $b_{j,s} \in B$. Hence, CN is a causal net.

Next, we prove that p is a homomorfism. Clearly, $p(B) \subseteq P$ and $p(E) \subseteq T$. The restriction of p to $e_i^\bullet$ is a bijection between $e_i^\bullet$ and $p(e_i)^\bullet$ because

$$e_i^\bullet = \{b_{i,s} \mid (e_i, b_{i,s}) \in G\} = \{b_{i,s} \mid s \in t_i^\bullet\}. \tag{51}$$

Consider then p on the preset of an event,

$$^\bullet e_i = \{b_{j,s} \mid s \in {}^\bullet t_i \wedge j = \max\{k \mid b_{k,s} \in B \wedge k < i\}\}. \tag{52}$$

Let $s \in {}^\bullet p(e_i) = {}^\bullet t_i$. In the firing schedule, there is the firing (t_i, θ_i). This is possible only if $s \in M_0$ or $s \in t_j{}^\bullet$ for some $1 \leq j < i$, or otherwise t_i could not become enabled. Therefore, $s = p(b_{j,s})$ for some $0 \leq j < i$. Consequently, there is a condition $b_{j,s}$ in ${}^\bullet e_i$ for each $s \in {}^\bullet t_i$ and the restriction of p to ${}^\bullet e_i$ is surjective. Since the j in $b_{j,s} \in {}^\bullet e_i$ is a maximum, it is unique, and there is exactly one $b_{j,s}$ in ${}^\bullet e_i$ for each $s \in {}^\bullet t_i$, and the restriction is also injective. Thus, the restriction of p to ${}^\bullet e_i$ is a bijection between ${}^\bullet e_i$ and ${}^\bullet p(e_i)$. The restriction of the homomorfism to $Min(CN) = \{b_{0,s} \mid s \in M_0\}$ is a bijection between $Min(CN)$ and M_0. This shows that p is a homomorfism from CN to the underlying net system of TPN. Thus, (CN, p) is a process of the underlying net system of TPN.

In addition, we must show that τ is a valid timing. We first examine the enabling time of transitions and the function I in order to show that Eqn. 28 holds in the constructed time process. Let transition t be enabled after firing $t_1, \ldots, t_j$. It became enabled when the last of its preplaces received a token, at time

$$\max(\left\{\Sigma_{k=1}^{l}\theta_k \mid 1 \leq l \leq j \wedge s \in t_l{}^\bullet \wedge s \in {}^\bullet t\right\} \cup \{0\}). \tag{53}$$

Let s and l be the values of the free variables at which the maximum is reached. There cannot be any k such that $l < k \leq j$ and $s \in {}^\bullet t_k$. Otherwise, t would not be enabled, or there would be some k' with $k \leq k'$ and $s \in t_k{}^\bullet$ and k' would be the point of maximum, not l. That is, $\nexists k : (l < k \leq j \wedge s \in {}^\bullet t_k)$. This implies $\nexists k : (l < k \leq j \wedge (b_{l,s}, e_k) \in G)$, because according to Eqn. 47, $(b_{l,s}, e_k) \in G$ implies $s \in {}^\bullet t_k$. Moreover, $(b_{l,s}, e_k) \in G$ cannot hold unless $l < k$. Together, this means

$$\nexists k : (0 \leq k \leq j \wedge (b_{l,s}, e_k) \in G). \tag{54}$$

Since this is always true for the maximum, we can insert it into the formula of Eqn. 53, which equals

$$\begin{aligned}
&= \max(\{\Sigma_{k=1}^{l}\theta_k \mid 1 \leq l \leq j \wedge s \in t_l{}^\bullet \wedge \\
&\qquad\qquad \nexists k : (l < k \leq j \wedge (b_{l,s}, e_k) \in G) \wedge s \in {}^\bullet t\} \cup \{0\}) \\
&= \max(\{\tau(e_l) \mid b_{l,s} \in Cut(E_j) \wedge l \geq 1 \wedge s \in {}^\bullet t\} \cup \{0\}) \\
&= \max(\{\tau({}^\bullet b) \mid b \in Cut(E_j) \setminus Min(CN) \wedge p(b) \in {}^\bullet t\} \cup \{0\}). \\
&= TOE(Cut(E_j), t).
\end{aligned} \tag{55}$$

This is the enabling time of t. Consequently, the value of $I(t)$ after the jth firing is

$$I_j(t) = \tau(e_j) - TOE(Cut(E_j), t). \tag{56}$$

Second, we examine the marking and corresponding set of events in order to show that Eqn. 27 holds in the constructed time process. The sum of the changes in marking caused by the firings of $t_1, \ldots, t_j$ equals the sum of tokens produced and consumed by the events $e_1, \ldots, e_j$. Thus, the set $Cut(E_j)$ is isomorfic with the marking after the jth firing, M_j. This implies Eqn. 27, $M_j = p(Cut(E_j))$, and that the set of conditions $Cut(E_j)$ maps injectively to places.

Third, we show that Eqn. 25 of Lemma 20 holds in the constructed time process. ${}^\bullet e_{j+1} \subseteq Cut(E_j)$ because if $b_{k,s} \in {}^\bullet e_{j+1}$ and also $b_{k,s} \in {}^\bullet e_l$ for some $l \leq j$, then $|b_{k,s}{}^\bullet| \geq 2$, which is impossible in a causal net. This and the injective mapping of $Cut(E_j)$ to places imply

$$TOE({}^\bullet e_{j+1}, p(e_{j+1})) = TOE(Cut(E_j), p(e_{j+1})). \tag{57}$$

Finally, with these results, it is possible to show that τ is a valid timing. Let $e_i \in E$. Choose $j = i - 1$ and $t = p(e_i)$ in Equations 57 and 56. Together, they give

$$I_{i-1}(p(e_i)) = \tau(e_{i-1}) - TOE({}^\bullet e_i, p(e_i)). \tag{58}$$

Eqn. 6 of the firing condition can be transformed to

$$\tau(e_i) \geq \tau(e_{i+1}) - I_{i-1}(p(e_i)) + Eft(p(e_i)). \tag{59}$$

Substituting the value of $I_{i-1}(p(e_i))$ from Eqn. 58, we get Eqn. 18, the first validness criterion.

The second validness criterion is a little more complicated. Let still $e_i \in E$, and let $j = \max(\{k \mid \tau(e_k) < \tau(e_i)\} \cup \{0\})$. Then, $0 \leq j < i$.

$$Earlier(e_i) = \{e_k \in E \mid \tau(e_k) < \tau(e_i)\} = Earlier(e_{j+1}) = E_k. \tag{60}$$

For all k, $k > j$ implies $\tau(e_k) = \tau(e_i)$. Denote

$$C = Cut(Earlier(e_i)) = Cut(Earlier(e_{j+1})) = Cut(E_j), \tag{61}$$

and let $t \in Enabled(p(C))$. Rewriting the firing condition on the latest firing time (Eqn. 7), and substituting the value of $I_{i-1}(t)$ from Eqn. 56,

$$\begin{aligned} \tau(e_i) = \tau(e_{j+1}) &\leq \tau(e_j) - I_{i-1}(t) + Lft(t) \\ &= TOE(C, t) + Lft(t). \end{aligned} \tag{62}$$

This shows that the requirement on latest firing times in the definition of valid timing (Eqn. 19) holds. Thus, τ is a valid timing of the process, and (CN, p, τ) is a time process of CN.

It remains to show that the time process has an interleaving ρ for which $\sigma = FS(\rho)$. The interleaving is simply $\rho = e_1, e_2, \ldots, e_n$. It is an interleaving since every event is in the sequence exactly once, the process is constructed in such a way that the partial order does not conflict with the order of indices, $e_i < e_j \Rightarrow i < j$, and most importantly, the choice of τ directly guarantees that $\tau(e_i) < \tau(e_j) \Rightarrow i < j$. FS straightforwardly maps ρ on σ. This concludes the proof of Theorem 23. □

So far, we have shown that there is a surjective mapping FS from the interleavings of time processes to the firing schedules fireable from the initial state of the net. The next theorem will complete the proof that the FS is bijective. As long as the time Petri net is contact-free, there is a unique process corresponding to each firing schedule, up to isomorfism of processes.

Theorem 24 Let σ be a firing schedule of a time Petri net TPN. The process of TPN with interleaving ρ, such that $\sigma = FS(\rho)$, is unique up to renaming of elements. □

Proof The proof is done by comparing an arbitrarily chosen process with the one in the proof of Theorem 23. Let (CN', p', τ') be an a time process of TPN with an interleaving ρ' such that $FS(\rho') = \sigma$. Here, $CN' = (B', E', G')$. Obviously, the interleaving has the same number of events as ρ, and they can be renamed $e_1, \ldots, e_n$ in the same order as in ρ. Since the events map to the same firings, $p(e_i) = p'(e_i)$ for all events e_i. Thereafter, also the postset of an event maps to the same places of the net in both processes. The conditions B' can be renamed as in Eqn. 45. For the minimal conditions that correspond to the initial marking, there must be corresponding tokens in B' and, since they all are in different places, they can be renamed $b_{o,s}$. After the renaming, $p(b_{i,s}) = p'(b_{i,s})$ for all conditions.

It remains to show that the flow relation G' is isomorfic to G. For the connections $(e_i, b_{i,s}) \in G$ this follows simply from the way we have renamed the postplaces. For the preplaces, it is essential that the time Petri net is contact-free. Then, there are no two conditions mapping to the same place in $Cut(Earlier(e_k))$ for any event e_k. The only choice for preconditions of an event are the ones last produced, as in Eqn. 47. Thus, also G' is isomorfic to G. □

As we have seen, in a contact-free time Petri net, the relation between firing schedules fireable from the initial state and interleavings of time processes is bijective. The results of this section are not surprising, but they assure that the time processes correctly represent the behavior of the system. The timing and the slightly more general notion of contact-freeness make the proofs complicated in comparison to similar ones for untimed contact-free net systems.

5 All valid timings of a process

It would be desirable to somehow characterize the set of all valid timings for a given process. The definition of valid timing only gives a way to check the validness, but it does not help much in constructing timing functions. Still, the need of having the entire set of valid timings is obvious. From the set, one can answer questions like, what is the longest time that can pass between two events, can an event occur after another, and so on. We will see that the set of all timings can be computed in a fairly general setting, but that the cost of the computation may be high.

In section 5.1, an alternative criteria for the validness of a timing are derived. It is used in Section 5.2 to construct an algorithm for computing the set all valid timings of a causal process or deciding if any valid timing exists. In Section 5.3, we return to question of when the validity of a timing in a process can be decided by finding an invalid initial subprocess. An example of maximizing time separation of events is given in Section 5.4. Finally, in Section 5.5, we briefly discuss the improvements in efficiency that can be achieved in a restricted class of nets.

5.1 Deciding events

We will now give an alternative characterization of valid timings. It is based on the idea that a decision has to be made about the firing of all potentially enabled transition and that this decision is made by some actual event in the system. The concepts choice transition and deciding event will be defined. The approach of this section is aimed at computing the set of all of valid timings. Therefore, it cannot take advantage of the structure of a single timing function like the definition of validness does.

In processes of untimed net systems, all antichains of conditions represent subsets of reachable markings. The is not the case in processes of time Petri nets, but the antichains are still useful, because all reachable markings are represented by some of them. Antichains are traditionally called co-sets, *co* meaning a concurrency relation between process elements [22].

Definition 25 (co-set, cut) A set B' of conditions in a causal net is a *co-set* if no two conditions in the set are causally related.

$$\forall b, b' \in B' : (\neg(p \leq q) \wedge \neg(q \leq p)) \tag{63}$$

A maximal co-set is called a *cut*. □

We saw in Section 4.2 that in time processes, the sets $p(Cut(Earlier(e))$ correspond to markings of the time Petri net. In that light, the following lemma is not surprising.

Lemma 26 Let (CN, p, τ) be a time process, where the causal net is $CN = (B, E, G)$. $Cut(E)$ and $Cut(Earlier(e))$ for all $e \in E$ are co-sets and so are all their subsets. □

Proof Assume that $b, b' \in Cut(E)$ and $b' < b$. Then , there exist an event $e' \in E$ such that $b'\ G\ e'\ G \ldots G\ b$. But then $b \notin Cut(E)$, which contradicts with the assumption.

Similarly, assume that $b, b' \in Cut(Earlier(e))$ and $b' < b$. There exist events e' and e'' such that $b'\ G\ e'\ G \ldots G\ e''\ G\ b$. It has to be the case that ${}^\bullet b = e'' \in Earlier(e)$. Since $\tau(e') \leq \tau(e'')$, also $e' \in Earlier(e)$ and $b' \notin Cut(Earlier(e))$, again a contradiction. □

When a transition becomes enabled in a time Petri net, it will eventually fire by its latest firing time, unless it is disabled by the firing of another transition. Furthermore, it is possible for the disabling to occur before the transition becomes fireable. And, the transition may never become enabled if it is "disabled" by removing some tokens before the others have been all inserted to the places. In a net system, there would be an interleaving of the events where the marking is reached, but the timing restrictions may exclude such interleavings.

Since our definition does not require a processes to continue infinitely or until all enabled transitions have been fired, a process is not necessarily maximal. The first transitions left out at the end of the process are naturally enabled at the marking corresponding to $Cut(E)$, but no decision is made about firing or disabling them. In the next definition, the potentially enabled events at co-sets are called choice transitions and the undecided ones at the end of the process are extension transitions. The decision about firing or disabling a choice transition is quite naturally done by some event of the process.

Definition 27 (choice, extension, deciding event) Let (CN, p) be a causal process of a time Petri net, where $CN = (B, E, G)$. Transition t is a *choice* at $B' \subseteq B$ iff B' is a co-set that maps injectively to places and ${}^\bullet t = p(B')$. The choice is an *extension* of the process iff $B' \subseteq Cut(E)$.

Let e be an event and t a choice transition at B' with $B' \cap {}^\bullet e \neq \emptyset$ and

$$\tau(e) \leq TOE(B', t) + Lft(t). \tag{64}$$

Then, e *decides* t at B. □

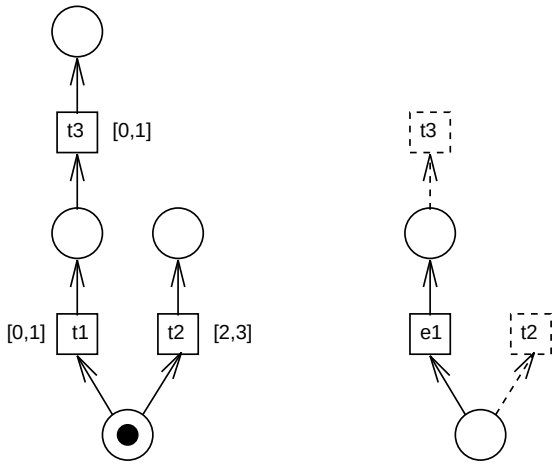

Figure 9: Choice and extension transitions

The deciding event of choice transition t at B' is either a firing of t, or it is the firing of another transition that disables t. In Figure 9, there is a time Petri net and its process. At the initial marking, $t1$ and $t2$ are choice transitions. Both are decided by the event $e1$; $t1$ is fired and $t2$ is disabled. $t3$ is an extension transition and remains undecided. In this case, $t2$ never becomes enabled, since it is disabled before its earliest firing time.

Definition 28 (complete with respect to a timing) A causal process (CN, p) of a time Petri net, where $CN = (B, E, G)$, is *complete* with respect to timing function τ iff for every extension transition t of the process,

$$\max\{\tau(e) \mid e \in E\} \;\leq\; TOE(Cut(E), t) + Lft(t). \tag{65}$$

□

When the process is complete, none of its concurrent parts "are left behind in time". That is, if there is some event in the process with time $\tau(e)$, all potential events with smaller value of τ must be included in the process. This property is also implicit in the definition of valid timing.

In untimed processes, the possibility of an event depends only on the causal past of the event, its *local configuration*. In time processes, an event can occur only if the history of the entire system allows it to become fireable. This forces us to always examine complete processes, not just local configurations.

It is computationally more efficient to state completeness in terms of the final conditions of the process. The following lemma will be used to reduce the number of instantiated inequalities created by the completeness requirement.

Lemma 29 If Eqn. 18 holds, Eqn. 65 is equivalent with

$$\forall e \in E : ((e^{\bullet\bullet} = \emptyset \wedge e \notin {}^{\bullet}B') \Rightarrow \tau(e) \leq TOE(Cut(E), t) + Lft(t)). \quad (66)$$

where $B' \subseteq Cut(E)$ is the set of conditions at which t is enabled. □

Proof When Eqn. 18 holds, the causal order of events implies time order, $e' \leq e \Rightarrow \tau(e') \leq \tau(e)$. Then, the maximum value of τ is obtained for some of the the causally maximal events in the causal order. Hence, it is enough to examine only those events in the causal order, for which $e^{\bullet\bullet} = \emptyset$. Moreover, if $e \in B' \subseteq {}^{\bullet}Cut(E)$, then $\tau(e) \leq TOE(Cut(E), t)$, and the inequality holds automatically. □

Before going to the algorithm for constructing the set of all valid timings, we prove the main theorem of this section. It characterizes the validness of a timing in an alternative way. While the definition of validness is optimized for checking known timing functions, this formulation of the validness criteria makes it possible to talk about sets of timings. The algorithm of Section 3.3 is based on the definition and it takes advantage of the structure of the single timing under inspection. The following theorem will result in another algorithm in Section 5.2 that computes the set of all valid timings, but it should not be used for checking single timings.

Theorem 30 Let TPN be a time Petri net and (CN, p) a causal process of TPN. A timing function τ is valid iff the validness criterion on the earliest firing time (Eqn. 18) holds, the process is complete with respect to the timing, and every choice transition is either an extension or decided by some event in the process. □

Proof We have to prove that the validness criterion on the latest firing time (Eqn. 19) holds iff the process is complete and every choice is either an extension or decided by an event.

First, we show that validness of a timing implies completeness of the process. Let τ be a valid timing on a causal process (CN, p) of a time Petri net, where $CN = (B, E, G)$. Assume that the process is incomplete. There is some transition t, a set of conditions B' and an event $e \in E$ such that ${}^{\bullet}t = p(B')$, $B' \subseteq Cut(E)$ and $\tau(e) > TOE(B', t) + Lft(t)$. Then, $B' \subseteq C_e = Cut(Earlier(e))$ and, since C_e maps injectively to places, $TOE(C_e, t) = TOE(B', t)$. Thus, $\tau(e) > TOE(C_e, t) + Lft(t)$, which contradicts with Eqn. 19. The assumption is false and the process is complete.

Second, we assume that a choice transition is not an extension and show that it is decided by some event. Let τ be valid. Let t be a transition and B' a co-set mapping injectively to places such that ${}^{\bullet}t = p(B')$ and $B'^{\bullet} \neq \emptyset$.

Choose some $e \in B'^{\bullet}$ such that $\tau(e) = \min\{\tau(e') \mid e' \in B'^{\bullet}\}$. Then, $B' \cap {}^{\bullet}e \neq \emptyset$. Consider first the case where $\tau(e) \leq \max(\{\tau(e') \mid e' \in {}^{\bullet}B'\} \cup \{0\}) = TOE(B', t)$. Then, $\tau(e) \leq TOE(B', t) + Lft(t)$. On the other hand, let $\tau(e) > \max(\{\tau(e') \mid e' \in {}^{\bullet}B'\} \cup \{0\})$. In that case, $B' \subseteq C_e = Cut(Earlier(e))$, because ${}^{\bullet}B' \subseteq Earlier(e)$ and the choice of e guarantees $\nexists e' \in B'^{\bullet} : \tau(e') < \tau(e)$. Thus, $t \in Enabled(C_e)$. The definition of valid timing (Eqn. 19) gives $\tau(e) \leq TOE(C_e, t) + Lft(t) = TOE(B', t) + Lft(t)$. Together with $B' \cap {}^{\bullet}e \neq \emptyset$ this suffices to show that t is decided by e.

The implication in the reverse direction remains to be shown. Let τ be a timing function, the process complete with respect to τ and every choice transition either an extension or decided by some event. Let $e \in E$ and $t \in Enabled(p(C_e))$. There exists a set $B' \subseteq C_e$ such that ${}^{\bullet}t = p(B')$. Since C_e maps injectively to places, this set is unique and $TOE(B', t) = TOE(C_e, t)$. Also, B' maps injectively to places. According to Lemma 26, B' is a co-set. Thus, t is a choice at B'. If t is an extension, completeness of the process guarantees that $\tau(e) \leq TOE(B', t) + Lft(t)$. On the other hand, if t is decided by event e', we have $\tau(e') \leq TOE(B', t) + Lft(t)$. $e' \notin Earlier(e)$ since $e' \in B'^{\bullet}$. Thus $\tau(e) \leq \tau(e') \leq TOE(B', t) + Lft(t)$. In both cases, $\tau(e) \leq TOE(C_e, t) + Lft(t)$ and Eqn. 19 holds. This concludes the proof of Theorem 30. □

5.2 Algorithm for all valid timings

A closer look at Theorem 30 reveals that all the properties required by the theorem from valid timings can be presented with inequalities. The three sources for inequalities are:

1. validness criterion on the earliest firing time (Eqn. 18),
2. deciding events of choice transitions (Eqn. 64),
3. completeness of the process (Eqn. 66).

Source 1 gives a set of inequalities on occurrence times that must be satisfied by all valid timings. Source 3 produces inequalities with the function TOE in them. Since TOE is defined as a maximum over a set of events, these can be expanded to sets of alternative inequalities with only occurrence times as variables. At least one of the alternatives in each set must hold in every valid timing. The same applies to Source 3, but these inequalities have to be instantiated with all the events satisfying the left side of the implication in Eqn. 66.

The existence of a valid timing for a causal process can be determined by an algorithm that nondeterministically chooses one elements from each set of alternative inequalities. If any one of the nondeterministic choices results in a set of inequalities with a feasible solution, the solution is a valid timing. Figure 10 shows the complete nondeterministic algorithm for deciding the existence of a valid timing on a causal process (CN, p), where $CN = (B, E, G)$.

The algorithm starts with an empty set of inequalities IE. For the earliest firing times (Source 1), it enumerates all events e in Eqn. 18. The inequalities have the form $\tau(e) \geq TOE({}^\bullet e, t) + Lft(p(e))$. There are $|E|$ such inequalities in total. For every inequality with TOE, ones of the form $\tau(e) \geq \tau(e') + Lft(p(e))$ are added to IE, one for each of the at most β preceding events $e' \in {}^{\bullet\bullet}e$, where β is the branching factor of the net. The number of inequalities added to IE is at most $\beta|E|$ and they have only occurrence times of events as variables.

Then, the choice transitions t at different co-sets B' are enumerated. The number of such transitions is less than $|T||B|^\beta$. For each choice that is not an extension (Source 2), a potential deciding event e is selected nondeterministically. There are at most β deciding events to select from. Eqn. 64 instantiated with e, B' and t has the form $\tau(e) \leq TOE(B', t) + Lft(t)$. Since TOE is defined as a maximum on the set B', the inequality can be satisfied in $|B'| \leq \beta$ ways. For each inequality containing TOE, a condition b is chosen nondeterministically from the set B' and an inequality of the form $\tau(e) \leq \tau({}^\bullet b) + Lft(t)$ is inserted to IE. For each extension transition (Source 3), the inequality in Eqn. 66 is instantiated with t, B' and all necessary different values of e. Again, one of the alternatives for the maximum in every TOE is chosen nondeterministically and the inequalities are added to the set IE. In the worst case, all the choices are extension. In that case, at most $|T||B|^\beta|E|$ inequalities with TOE are obtained making at most $|T||B|^\beta|E|$ nondeterministic selections among up to β alternatives on the way. The number of instantiated inequalities added to IE is also $|T||B|^\beta|E|$.

The resulting inequalities in IE are all linear. There are up to $(|T||B|^\beta + \beta)|E|$ of them. In addition, all variables $\tau(e)$ are nonnegative. The existence of a feasible solution for the set of a polynomial number of linear inequalities is decidable in polynomial time [7]. There exists a valid timing for the process iff the set of inequalities IE in one of the nondeterministic execution paths of the algorithm has a solution. If the branching factor β is constant for a class of Petri nets, the described algorithm decides in nondeterministic polynomial time the existence of a valid timing. We state this as a theorem.

Theorem 31 In a class of time Petri nets where the branching factor β is bounded by a constant, the existence of a valid timing can be decided by an algorithm that is NP with respect to the size of the process. □

```
IE : set of inequalities
EVS : set of events

IE := ∅
(* Eft *)
for each e ∈ E do
    if ••e ≠ ∅ then
        for each e' ∈ ••e do
            insert inequality τ(e) − τ(e') ≥ Lft(p(e)) into IE
    else
        insert inequality τ(e) ≤ Lft(p(e)) into IE
(* Lft *)
for each t ∈ T do
    for each B' ⊆ B such that |B'| = |•t| do
        if •t = p(B') then
            if B'• ≠ ∅ then
                (* choice *)
                choose nondeterministically e ∈ B'•
                EVS := {e}
            else
                (* extension *)
                EVS := {e ∈ E | e•• = ∅ ∧ e ∉ •B'}
            for each e ∈ EVS do
                if •B' ≠ ∅ then
                    choose nondeterministically e' ∈ •B'
                    insert inequality τ(e) − τ(e') ≤ Lft(t) into IE
                else
                    insert inequality τ(e) ≤ Lft(t) into IE
for each e ∈ E do
    insert inequality τ(e) ≥ 0 into IE
return true if IE has a feasible solution
```

Figure 10: Deciding existence of a valid timing

The algorithm can easily be modified to compute the longest possible time separation between two events. Furthermore, it can be generalized to optimize a linear combination of the occurrence times of the events of a process. This is done by solving a linear optimization problem for all different sets of inequalities given by different choices of deciding events and latest preconditions. These sets are obtained by enumerating all different alternatives for the nondeterministic choices in the algorithm. The longest distance between two events e' and e is a special case of such linear optimization, where the maximum of $\tau(e) - \tau(e')$ is computed.

The constraints of the optimization problems have a very special structure. Every variable $\tau(e)$ is nonnegative and the rest of the inequalities have the following forms:

A. $\tau(e) - \tau(e') \geq C$.

B. $\tau(e) - \tau(e') \leq C'$,

C. $\tau(e) \leq C''$,

The inequalities of type A come from Source 1. They are always all included in a set of inequalities; there are no alternatives. The inequalities of type B are obtained from Sources 2 and 3. Type C is a special case of type B, caused by transitions enabled at the initial marking. These inequalities are in sets of alternatives. One alternative from every set has to be chosen for each optimization problem. Thus, the optimization problems can be presented in conjunctive normal form:

$$\begin{aligned}
&(INEQ_1)\\
\wedge&(INEQ_2)\\
\wedge&(INEQ_3)\\
&\cdots\\
\wedge&(INEQ_k)\\
\wedge&(INEQ_{k+1,1} \vee INEQ_{k+1,2} \vee \cdots \vee INEQ_{k+1,m_{k+1}})\\
\wedge&(INEQ_{k+2,1} \vee INEQ_{k+2,2} \vee \cdots \vee INEQ_{k+2,m_{k+2}})\\
\wedge&(INEQ_{k+3,1} \vee INEQ_{k+3,2} \vee \cdots \vee INEQ_{k+3,m_{k+3}})\\
&\cdots\\
\wedge&(INEQ_{n,1} \vee INEQ_{n,1} \vee \cdots \vee INEQ_{n,m_n}).
\end{aligned} \tag{67}$$

The number of inequalities in the disjunctions vary. Many have only one.

In many cases, the number of inequalities can be significantly reduced.

- If there are two equivalent inequalities in a disjunction or two equivalent disjunctions in the conjunction, one can be removed.
- If one of two alternative inequalities implies the other, the more restrictive one can be removed. For instance, when $\tau(e_2) - \tau(e_1) \leq 4$ and $\tau(e_2) - \tau(e_1) \leq 6$ are in the same disjunction, the former is superfluous. The same applies to $\tau(e_2) - \tau(e_1) \leq 4$ and $\tau(e_2) \leq 4$.
- If an inequality without alternatives implies one of the inequalities in a disjunction, the entire disjunction can be removed. For example, $(\tau(e_1) \leq 3 \vee \tau(e_2) - \tau(e_1) \leq 5) \wedge (\tau(e1) \leq 2)$ can be reduced to $\tau(e1) \leq 2$.
- If the upper bound of an inequality of type B or type C is ∞, the entire disjunction containing it can be removed.

The linear optimization problems will be formed by choosing one of the alternatives in each disjunction. If one inequality in such a problem implies another, the stronger one can again be removed. If this rule is applied exhaustively, the number of inequalities will be at most $|E|^2 + \beta|E|$. This is because there are $|E|(|E| - 1)$ ways to pick e and e' in inequalities of type B and $|E|$ possible e in type C. From the algorithm it can be seen that there are at most $\beta|E|$ inequalities of type A. In addition to these inequalities, all variables are nonnegative.

In addition to the inequalities given by the algorithm, it is possible to impose other linear constraints on the system under optimization. New inequalities can simply be added to the sets. For instance, one can maximize the separation of two events, $\tau(e_8) - \tau(e_9)$, when another event happens earlier than a threshold time, $\tau(e_2) < 100$. This is done by adding the inequality to all alternative optimization problems and by performing the optimization with the new constraints.

While the size of the linear optimization problems grows reasonably with the size of the process ($O(|E|^2)$ inequalities in each problem), the number of optimization problems is exponential in the size of the process. Fortunately, the solutions to the optimization problems can be computed by changing only one of the constraint inequalities at a time. The linear optimization algorithm should be able to take advantage of this. In practice, most alternative sets of inequalities have the same optimum points. This can be utilized by giving the previous solution as an initial point for the next optimization problem. Most of the time, the initial value will be the optimum. Also, one could take advantage of the structure of the inequalities, for example, by using the Bellman-Ford algorithm [7] for finding feasible solutions to the systems of difference constraints. There may be further ways of optimizing the procedure.

It is, for instance, possible to start with partial sets of inequalities and save the intermediate solutions when adding alternative constraints. In that way, large groups of infeasible problems could be eliminated simultaneously. Obviously, when we are only interested in the existence of a valid timing, it is sufficient to find the first set of inequalities with a feasible solution. All alternatives need to be considered only if no valid timing exists.

5.3 More about invalid timings in growing processes

In Section 3.5, we promised to return to the question of invalid timings in growing processes. With the terminology introduced above, the ideas there can be stated in more compact form. The following Theorem says essentially the same thing as Theorem 17, but is more oriented towards the characterization of validness in Theorem 30.

Theorem 32 Let (CN, p), where $CN = (B, E, G)$, be a causal process with timing function τ. Assume that there is a choice transition t at a set B_t that is neither an extension nor decided by any event of E. If (CN, p) is an initial subprocess of another process $(\hat{CN}, \hat{p})$ such that $B_t{}^\bullet$ is equal in both processes, then $(\hat{CN}, \hat{p})$ does not have any valid timing function whose restriction to E equals τ. □

Proof It is easy to see that the candidates for deciding events are the same in both processes. If none is added, the process remains invalid. □

The most common case where $B_t{}^\bullet$ is equal in both processes is the one where $b^\bullet \in E$ for all $b \in B_t$. This is because when all conditions of the set B_t are consumed by some events in the smaller process, there is not room to add any new deciding events in the larger one. Consequently the invalid situation cannot be corrected by extending the process. This is the case for the process of Fig. 6(a). We can now easily complete the proof that was delayed in Section 3.2.

Proof of Theorem 17 This is a special case of Theorem 32, exactly the above discussed situation where all all conditions of the set B_t are already consumed by some event of E. □

The same problem with growing processes appears when solving the sets of inequalities constructed in Section 5.2. When do we know that there does not exist any valid timing, even if the process is generated further? Also, in optimization problems, we would often like to find the optimum value in a given process and all larger processes.

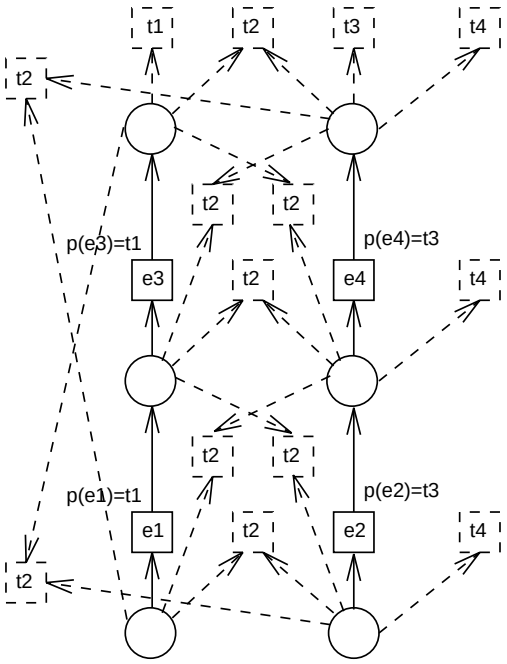

Figure 11: Computing all timings of a process

Suppose we have constructed the constraints of an optimization problem for a process (CN, p), where $CN = (B, E, G)$. The inequalities that will remain in any process having (CN, p) as a subprocess are

1. all inequalities from Source 1,
2. the alternative inequalities from Source 2, corresponding to choice a transition t at B_t, only if $b^\bullet \in E$ for all $b \in B_t$.

If this reduced set of constraints has a feasible solution, a larger process may have a valid timing. On the other hand, if there is no solution for this smaller problem, no processes having (CN, p) as an initial subprocess has any valid timing. Also, if the reduced constraints allows for a better result of an optimization problem than the original constraints, the optimum in (CN, p) may be limited by the size of the generated process. When the optimum value is not enhanced by reducing the set of constraints, the value is, in fact, optimal for all processes having (CN, p) as an initial subprocess.

5.4 Example

In Figure 11, there is a causal process of the time Petri net of Fig. 5(a), with all choice transitions. We are interested in the set of valid timings for this process. The algorithm of Section 5.2 gives the following constraints:

$$
\begin{array}{ll}
(\tau(e_1) \geq 1) & \text{Eft} \\
\wedge\ (\tau(e_3) - \tau(e_1) \geq 1) & \\
\wedge\ (\tau(e_2) \geq 1) & \\
\wedge\ (\tau(e_4) - \tau(e_2) \geq 1) & \\
\wedge\ (\tau(e_4) - \tau(e_3) \leq 3) & \text{Lft: extensions} \\
\wedge\ (\tau(e_3) - \tau(e_4) \leq 10) & \\
\wedge\ (\tau(e_1) \leq 3) & \text{Lft: events} \\
\wedge\ (\tau(e_3) - \tau(e_1) \leq 3) & \\
\wedge\ (\tau(e_2) \leq 10) & \\
\wedge\ (\tau(e_4) - \tau(e_2) \leq 10) & \\
\wedge\ (\tau(e_1) \leq 2 \ \vee\ \tau(e_2) \leq 2) & \text{Lft: other choices} \\
\wedge\ (\tau(e_3) - \tau(e_1) \leq 2 \ \vee\ \tau(e_3) - \tau(e_2) \leq 2 \ \vee\ \tau(e_4) - \tau(e_1) \leq 2 \ \vee\ \tau(e_4) - \tau(e_2) \leq 2) & \\
\wedge\ (\tau(e_1) - \tau(e_2) \leq 2 \ \vee\ \tau(e_4) - \tau(e_2) \leq 2) & \\
\wedge\ (\tau(e_3) - \tau(e_1) \leq 2 \ \vee\ \tau(e_2) - \tau(e_1) \leq 2) & \\
\wedge\ (\tau(e_3) - \tau(e_1) \leq 2 \ \vee\ \tau(e_3) - \tau(e_4) \leq 2) & \\
\wedge\ (\tau(e_4) - \tau(e_2) \leq 2 \ \vee\ \tau(e_4) - \tau(e_3) \leq 2) & \\
\wedge\ (\tau(e_2) - \tau(e_3) \leq 2) & \\
\wedge\ (\tau(e_1) - \tau(e_4) \leq 10) &
\end{array}
$$

The first four constraints are the earliest firing times of the events. The next two guarantee the completeness of the process. Then come the alternative reasons why each choice transition does not reach its latest firing time. Four of the choices are decided by events that are firings of the choice transition.

These inequalities have a solution and, thus, there exists a valid timing. We can, for instance, find the maximum of $\tau(e_3) - \tau(e_2)$. The maximum is 5, at $\tau(e_1) = 3$, $\tau(e_2) = 1$, $\tau(e_3) = 6, \tau(e_4) = 4$. We can also find maximum firing time of event e_4. The result is 7, at $\tau(e_1) = 2$, $\tau(e_2) = 6$, $\tau(e_3) = 4, \tau(e_4) = 7$. The latter result may appear surprising, but in order for $\tau(e_4)$ to be higher, other parts of the process should be extended. Thus, these results are only a maximum in this process, not necessarily in larger processes. The reduced set of constraints introduced in Section 5.3 becomes:

$$
\begin{array}{ll}
(\tau(e_1) \geq 1) & \text{Eft} \\
\wedge\, (\tau(e_3) - \tau(e_1) \geq 1) & \\
\wedge\, (\tau(e_2) \geq 1) & \\
\wedge\, (\tau(e_4) - \tau(e_2) \geq 1) & \\
\wedge\, (\tau(e_1) \leq 3) & \text{Lft: events} \\
\wedge\, (\tau(e_3) - \tau(e_1) \leq 3) & \\
\wedge\, (\tau(e_2) \leq 10) & \\
\wedge\, (\tau(e_4) - \tau(e_2) \leq 10) & \\
\wedge\, (\tau(e_1) \leq 2 \vee \tau(e_2) \leq 2) & \text{Lft: other choices} \\
\wedge\, (\tau(e_3) - \tau(e_1) \leq 2 \vee \tau(e_3) - \tau(e_2) \leq 2 \vee \tau(e_4) - \tau(e_1) \leq 2 \vee \tau(e_4) - \tau(e_2) \leq 2) & \\
\wedge\, (\tau(e_1) - \tau(e_2) \leq 2 \vee \tau(e_4) - \tau(e_2) \leq 2) & \\
\wedge\, (\tau(e_3) - \tau(e_1) \leq 2 \vee \tau(e_2) - \tau(e_1) \leq 2) &
\end{array}
$$

In this set, the maximum of $\tau(e_3) - \tau(e_2)$ is still 5. The result indicates that it is the maximum in all processes having the process of Fig. 11 as an initial subprocess. On the other hand, the maximum of e_4 differs considerably from the previous result. It is now 20, obtained for example at $\tau(e_1) = 1$, $\tau(e_2) = 10$, $\tau(e_3) = 2, \tau(e_4) = 20$. In order to learn the maximum firing time of event e_4, we would have consider all different ways in which the process can be extended. The value obtained from the reduced set of inequalities, 20, is only an upper bound for $\tau(e_4)$.

5.5 Extended free choice time Petri nets

Since the computation of all valid timings is expensive in the general case, it is necessary to find classes of models where the complexity is reasonable. We will see that in a restricted class of time Petri nets, where no confusion occurs, the validness criteria can be simplified and, thus, computing valid timings is much easier. In practice, one has to find a reasonable balance between modelling power of the net class and cost of analysis.

Extended free choice nets [3] are usually used to avoid *confusion*, a situation where a conflict occurs or does not occur depending on the order of causally unrelated firings. We will see that in the confusion-free net class timing analysis in significantly easier than in the general case. Why does confusion complicate the timing analysis so much? The reason is twofold. First, timing determines the order of causally unrelated events. Second, a conflict in a time Petri net does not mean that all of the alternative transitions can fire, because the possibility of firing depends on the relative timing of these events. Together, these two mechanisms create complex dependences between events.

When only restricted forms of synchronization are allowed, the dependences do not arise. The notion of extended free choice extends to time Petri nets without any change.

Definition 33 (extended free choice) A time Petri net is *extended free choice* iff for all two transitions t and t', ${}^\bullet t \cap {}^\bullet t' \neq \emptyset$ implies ${}^\bullet t = {}^\bullet t'$. □

This structural property is conserved by process generation.

Lemma 34 All processes of an extended free choice time Petri net are also extended free choice. If a process is extended with all of its choice transitions, the resulting net is also extended free choice. □

Proof Assume the contrary: two events e and e' have intersecting presets, ${}^\bullet e \cap {}^\bullet e' \neq \emptyset$, but they are not equal, ${}^\bullet e \neq {}^\bullet e'$. In that case, ${}^\bullet p(e) \cap {}^\bullet p(e') \neq \emptyset$, but ${}^\bullet p(e) \neq {}^\bullet p(e')$. This conflicts with the extended free choice property. Similar reasoning applies to the choice transitions. □

The following Theorem is similar to Theorem 30, but — because of the extended free choice property — computationally much easier to handle.

Theorem 35 Let $EFCTPN$ be an extended free choice time Petri net and (CN, p) a causal process of $EFCTPN$, where $CN = (B, E, G)$. A timing function τ is valid iff the criterion on the earliest firing time (Eqn. 18) holds, the process is complete with respect to the timing, and

$$\forall e \in E : \tau(e) \leq TOE({}^\bullet e, p(e)) + \min\{Lft(t) \mid {}^\bullet t = {}^\bullet p(e)\}\,. \qquad (68)$$

□

Proof A seen by comparing Theorems 30 and 35, we need to prove that Eqn. 68 holds iff every choice transition that is not an extension has a deciding event.

Assume first that Eqn. 68 holds and that transition t is a choice but not an extension at a set of conditions B'. There is at least one event e with ${}^\bullet e \cap B' \neq \emptyset$. In an extended free choice process this means ${}^\bullet e = B'$ and $TOE({}^\bullet e, p(e)) = TOE({}^\bullet e, t) = TOE(B', t)$. Eqn. 68 gives

$$\tau(e) \;\leq\; TOE({}^\bullet e, p(e)) + Lft(t) \;=\; TOE(B', t) + Lft(t). \qquad (69)$$

Hence, e decides t at B'.

On the other hand, assume that every choice transition that is not an extension has a deciding event. Let e be an event and t a transition such that ${}^\bullet t = {}^\bullet p(e)$.

The set of conditions ${}^\bullet e \subseteq Cut(E_k)$ is a co-set and it maps injectively to places. Thus, t is a choice at ${}^\bullet e$ and it must be decided by some event. Since there are no conflicts in a process, e is the only event in $({}^\bullet e)^\bullet$. Therefore, e has to decide t, $\tau(e) \leq TOE({}^\bullet e, t) + Lft(t)$. This is true for all transitions t with ${}^\bullet t = {}^\bullet p(e)$ and, hence, Eqn. 68 holds. □

If the above validness criteria is used instead of that of Theorem 30, the number of alternatives in the sets on inequalities characterizing all valid timings of a process will be significantly smaller. This is because with the above validness criteria, it is not necessary to consider different alternatives for deciding events. Unfortunately, the efficient timing analysis for extended free choice systems does not come without disadvantages. The kinds of systems that can be modelled by extended free choice time Petri nets are limited. While the cost of timing analysis is high in the general case and the modelling power of the extended free choice nets is limited, it seems that processes with only few choice transitions violating the extended free choice property can exhibit complex behavior and be still analyzable in reasonable time.

6 Conclusions

The objective of this thesis is to give time Petri nets a partial order semantics, like the nonsequential processes of untimed net systems. As the main contribution, we introduce time processes for representing the behavior of contact-free time Petri nets. The time processes are defined as causal processes with a valid timing on the events. The relationship between the time processes and the usual firing schedule semantics is examined in detail with the conclusion that the approaches are compatible. We present algorithms for checking validness of known timings and for constructing the set of all valid timings of a process for use in optimization problems.

Time Petri nets are a class of Petri nets where transitions have earliest and latest allowed firing times. They can be used to analyze absolute limits of timing and effects of time constraints on the state space of the system. We assume time Petri nets to be contact-free and have divergent time. According to our definition, a time Petri net can be contact-free even though the underlying untimed net system is not.

Processes are partial orders of events and conditions representing firings and marked places in the net, respectively. Unlike firing sequences, processes preserve concurrency and causal relationships between events. The central new concept in this thesis, a time process, is defined as a causal processes with valid timing. This means that the events of the process are labeled with occurrence times which must satisfy specific validness criteria. The complexity of timing analysis lies in dependences between causally unrelated events, created by the latest firing times of transitions. With the partial order approach, these dependences must be explicitly dealt with. They force us to look at other enabled transitions, not only the causal past of an event, when assessing validness of the occurrence time of the event.

An efficient algorithm for verifying validness of timings is presented. The algorithm is especially effective on processes with many immediate events. This is important, because models of practical systems often have large untimed parts where firings take no time. Also, a sufficient condition is given for when the invalidity of timings for larger processes can be inferred from their initial subprocesses. The relationship between interleavings of time processes of a time Petri net and firing schedules of the net is bijective. The relation is built on interleavings, linearizations of the process partial order that respect both the causal and time order of events. The results indicate that the time process and firing schedule presentations of system behavior have essentially the same meaning.

The definition of valid timing is optimized for checking validness of known

timings. As such, it does not give any direct way for constructing time processes. Therefore, an alternative characterization for validness is presented. It is based on the idea that a decision has to be made about firing or disabling every even potentially enabled transition, and the decision is made by an actual event of the process. With the alternative formulation of the validness criteria, the existence of a valid timing for a given process can be decided in nondeterministic polynomial time. The algorithm can also be used for constructing the set of all valid timings. While the algorithm for verifying validness takes advantage of the structure of the particular process and timing to minimize the number of comparisons, an algorithm constructing the set of all valid timings has to go through a lot more alternatives. The timings are presented as sets of alternative linear inequalities. With these, one can answer questions like, what is the maximal time separation between two events. In a net class with restricted forms of synchronization, extended free choice time Petri nets, the validness criteria can be simplified significantly.

The presented techniques could be improved with various optimizations, for example, in solving the sets of linear inequalities. It seems that efficient algorithms could be developed for timing analysis in extended free choice time Petri nets. Also, other restricted net classes should be considered. Interesting topics for future research include incorporating the timing analysis with process generation to compute properties of whole nets, not only single processes, and extending the presented methods to time stream Petri nets. Also, the idea of branching processes for time Petri nets is intriguing, although it is difficult to see how the different time processes could be embedded into one partial order.

References

[1] Bernard Berthomieu and Michel Diaz. Modelling and verification of time dependent systems using time Petri nets. *IEEE Transactions on Software Engineering*, 17(3):259–273, 1991.

[2] Bernard Berthomieu and Miguel Menasche. A state enumeration approach for analyzing time Petri nets. In *Proceedings of the 3rd European Workshop on Applications and Theory of Petri Nets*, pages 27–56, September 1982.

[3] Eike Best. Structure theory of Petri nets: the free choice hiatus. In *Petri Nets: Central Models and Their Properties*, volume 254 of *LNCS*, pages 168–205. Springer-Verlag, 1986.

[4] Eike Best and Raymond Devillers. Sequential and concurrent behavior in Petri net theory. *Theoretical Computer Science*, 55:87–136, 1987.

[5] Eike Best and César Fernández. *Nonsequential Processes, A Petri Net View*, volume 13 of *EATCS Monographs on Computer Science*. Springer-Verlag, 1988.

[6] Hanifa Boucheneb and Gérard Berthelot. Toward simplified construction of time Petri nets reachability graph. In *Proceedings of the 5th International Workshop on Petri Nets and Performance Models*, pages 46–55, October 1993.

[7] Thomas H. Cormen, Charles E Leiserson, and Ronald L. Rivest. *Introduction to algorithms*. MIT Press, 1990.

[8] Michel Diaz and Patrick Sénac. Time stream Petri nets, a model for timed multimedia information. In *Proceedings of the 15th International Conference on Application and Theory of Petri Nets 1994*, volume 815 of *LNCS*, pages 219–238. Springer-Verlag, June 1994.

[9] Joost Engelfriet. Branching processes of Petri nets. *Acta Informatica*, 28:575–591, 1991.

[10] Javier Esparza. *Model Checking Using Net Unfoldings*. Number 14/92 in Hildesheimer Informatikberichte. Universität Hildesheim, Institut für Informatik, Hildesheim, Germany, October 1992.

[11] Javier Esparza. Model checking using net unfoldings. *Science of Computer Programming*, 23:151–195, 1994.

[12] Carlo Ghezzi, Sandro Morasca, and Mauro Pezzè. Validating timing requirements for TB net specifications. Technical report, Politecnico di

Milano, Department of Electronic Engineering and Information Sciences, June 1994.

[13] U. Goltz and W. Reisig. The non-sequential behavior of Petri nets. *Information and Control*, 57(1):125–147, 1983.

[14] Henrik Hulgaard and Steven M. Burns. Efficient timing analysis of a class of Petri nets. In *Computer Aided Verification 7th International Workshop CAV'95*, volume 939 of *LNCS*, pages 423–436. Springer-Verlag, 1995.

[15] K.L. McMillan. Using unfoldings to avoid the state explosion problem in the verification of asynchronous circuits. In *Computer Aided Verification, 4th International Workshop CAV'92*, volume 663 of *LNCS*, pages 164–177. Springer-Verlag, 1992.

[16] Philip M. Merlin and David J. Farber. Recoverability of communication protocols — implications of a theoretical study. *IEEE Transactions on Communications*, 24(9):1036–1043, 1976.

[17] Michael Karl Molloy. *On the Integration of Delay and Throughput Measures in Distributed Processing Models.* PhD thesis, University of California, Los Angeles, 1981.

[18] Tadao Murata. Petri nets, properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, 1989.

[19] ISO/IEC JTC1/SC21/WG1 N1053. Enhancements to LOTOS, 1995.

[20] Morgens Nielsen, Gordon Plotkin, and Glynn Winskel. Petri nets, event structures and domains, part 1. *Theoretical Computer Science*, 13:85–108, 1981.

[21] J. L. Peterson. *Petri Net Theory and the Modeling of Systems.* Prentice-Hall, Inc., Englewood, NJ, 1981.

[22] C. A. Petri. Nichtsequentielle prozesse. Technical report, GMD, Institut für Informationssystemforschung, June 1977.

[23] Louchka Popova. On time Petri nets. *Journal Inform. Process. Cybern. EIK*, 27(4):227–244, 1991.

[24] C. Ramchandani. Analysis of asynchronous concurrent systems by timed Petri nets. Technical report, Project MAC, TR 120, MIT, February 1974.

[25] Wolfgang Reisig. *Petri Nets: An Introduction*, volume 4 of *EATCS Monographs on Theoretical Computer Science.* Springer-Verlag, 1985.

[26] Wolfgang Reisig. Distributed algorithms, modelling and analysis with Petri nets. An unpublished manuscript, March 1995.

[27] Patrick Sénac, Michel Diaz, and Pierre de Saqui-Sannes. Toward a formal specification of multimedia synchronization scenarios. *Ann. Télécommun.*, 49(5–6):297–314, 1994.

[28] Peter H. Starke. *Analyse von Petri-Netz-Modellen.* B. G. Teubner, Stuttgart, 1990.

[29] P. S. Thiagarajan. Elementary net systems. In *Petri Nets: Central Models and Their Properties*, volume 254 of *LNCS*, pages 26–59. Springer-Verlag, 1986.

[30] Valentin Valero, David de Frutos, and Fernando Cuartero. Timed processes of timed Petri nets. In *Proceedings of the 16th International Conference on Application and Theory of Petri Nets 1995*, volume 935 of *LNCS*. Springer-Verlag, June 1995.

[31] Józef Winkowski. Algebras of processes of timed Petri nets. In *CONCUR'94: Concurrency Theory*, volume 836 of *LNCS*, pages 195–209. Springer-Verlag, August 1994.

[32] Glynn Winskel. Event structures. In *Advances in Petri Nets 1986, Part II*, volume 255 of *LNCS*, pages 325–392. Springer-Verlag, September 1986.

[33] Tomohiro Yoneda, Atsfumi Shibayama, Bernd-Holger Schlingloff, and Edmund M. Clarke. Efficient verification of parallel real-time systems. In *Computer Aided Verification, 5th International Workshop CAV'93*, volume 697 of *LNCS*, pages 321–332. Springer-Verlag, 1993.