A Memo on Computability in Time Petri Nets

Extended Abstract

Louchka Popova-Zeugmann

Institut für Informatik, Humboldt-Universität zu Berlin,
popova@informatik.hu-berlin.de

1 Introduction

Time Petri nets were introduced by Merlin in [5] in order to study recoverability problems in computer systems and the design of communication protocols. Berthomieu and Menasche in [2] res. Berthomieu and Diaz in [1] provide a method for the analysis of the qualitative behavior of the net. They divide the state spaces in state classes which are describe by a marking and time domain given by inequalities. The reachability graph is defined as a directed graph where the vertices are the reachable state classes and the edges are labeled by transitions. The net time is taken into account in the reachability graph by means of the inequalities which describe the state classes. This is a disadvantage for a quantitative analysis of the TPN. The reachability graph has the property, that the graph is finite iff the TPN is bounded. A similar definition for a reachability graph for a TPN delivers [3].

The property from above namely that the graph is finite iff the TPN is bounded is also true for the reachability graph defined here. In contrast to above, the time appears explicitly in this reachability graph as weights on some edges of the graph. Moreover, the vertices contain no information. Thus, our reachability graph is a usual directed weighted graph. Therefore, it is now possible to use the graph theory in order to accomplish quantitative analysis.

2 Basics

2.1 Notations and Definitions

In this note we use following notations: $\mathbb{N}$ is the set of natural numbers, $\mathbb{N}^+ := \mathbb{N} \setminus \{0\}$. $\mathbb{Q}_0^+$ res. $\mathbb{R}_0^+$ is the set of nonnegative rational numbers res. set of nonnegative real numbers . Let g be a given function from A to B. T^* denotes the language of all words over the alphabet T, including the empty word e; $l(w)$ is the length of the word w.

Definition 1 (Petri net). *The structure $N = (P, T, F, V, m_o)$ is called a Petri net (PN) iff*

(a) P, T, F *are finite sets with*
$P \cap T = \emptyset$, $P \cup T \neq \emptyset$, $F \subseteq (P \times T) \cup (T \times P)$ *and* $dom(F) \cup cod(F) = P \cup T$
(b) $V : F \longrightarrow \mathbb{N}^+$ *(weight of the arcs)*
(c) $m_o : P \longrightarrow \mathbb{N}$ *(initial marking)*

A marking of a PN is a function $m : P \longrightarrow \mathbb{N}$, such that $m(p)$ denotes the number of tokens at the place p. The pre-sets and post-sets of a transition t are given by ${}^\bullet t := \{p \mid p \in P \wedge (p,t) \in F\}$ and $t^\bullet := \{p \mid p \in P \wedge (t,p) \in F\}$, respectively. Each transition $t \in T$ induces the marking t^- and t^+, defined as follows:

$$t^-(p) = \begin{cases} V(p,t) & \text{,iff} \quad (p,t) \in F \\ 0 & \text{,iff} \quad (p,t) \notin F \end{cases} \qquad t^+(p) = \begin{cases} V(t,p) & \text{,iff} \quad (t,p) \in F \\ 0 & \text{,iff} \quad (t,p) \notin F \end{cases}$$

Moreover, Δt denotes $t^+ - t^-$. A transition $t \in T$ is enabled (may fire) at a marking m iff $t^- \leq m$ (e.g. $t^-(p) \leq m(p)$ for every place $p \in P$). When an enabled transition t at a marking m fires, this yields a new marking m' given by $m'(p) := m'(p) + \Delta t(p)$ and denoted by $m \xrightarrow{t} m'$.

Definition 2 (Time Petri net). *The structure* $Z = (P, T, F, V, m_o, I)$ *is called a Time Petri net (TPN) iff:*

(a) $S(Z) := (P, T, F, V, m_o)$ *is a PN.*
(b) $I : T \longrightarrow \mathbb{Q}_0^+ \times (\mathbb{Q}_0^+ \cup \{\infty\})$ *and* $I_1(t) \leq I_2(t)$ *for each* $t \in T$*, where* $I(t) = (I_1(t), I_2(t))$.

A TPN is called finite Time Petri net (FTPN) iff $I : T \longrightarrow \mathbb{Q}_0^+ \times \mathbb{Q}_0^+$.

I is the interval function of Z, $I_1(t)$ and $I_2(t)$ the earliest firing time of t ($eft(t)$) and the latest firing time of t ($lft(t)$), respectively. It is not difficult to see (cf. [7]) that considering TPNs with $I : T \longrightarrow \mathbb{N} \times (\mathbb{N} \cup \{\infty\})$ will not result in a loss of generality. Therefore only such time functions I will be considered subsequently. Furthermore, *conflict* is used in the strong sense: two transitions t_1 and t_2 are in conflict iff ${}^\bullet t_1 \cap {}^\bullet t_2 \neq \emptyset$. The PN $S(Z)$ referred to as the skeleton of Z.

A state is characterized by a marking together with the momentary local time for enabled transitions or the sign $\sharp$ for the disabled transitions.

Definition 3 (state). *Let* $Z = (P, T, F, V, m_o, I)$ *be a TPN and* $h : T \longrightarrow \mathbb{R}_0^+ \cup \{\#\}$*. The pair* $z = (m, h)$ *is called a state in* Z *iff:*

(a) m *is a reachable marking in* $S(Z)$.
(b) $\forall t \; (\; (t \in T \wedge t^- \leq m) \; \longrightarrow \; h(t) \leq lft(t))$.
(c) $\forall t \; (\; (t \in T \wedge t^- \not\leq m) \; \longrightarrow \; h(t) = \#)$.

Interpretation of the notion "state" is as follows: within the net, each transition t has a clock $h(t)$. If t is enabled at a marking m, the clock of t $h(t)$ shows the time elapsed since t became most recently enabled. If t is disabled at m, the clock does not work (indicated by $h(t) = \#$). We call m a place-marking (short: p-marking) and h - a transition-marking (short: t-marking).

Now the dynamic aspects of TPNs – changing from one state into another – can be introduced: The state $z_o := (m_o, h_o)$ with $h_o(t) := \begin{cases} 0 & \text{iff} \quad t^- \leq m_0 \\ \# & \text{iff} \quad t^- \not\leq m_0 \end{cases}$ is set as the *initial state* of the TPN Z. A transition t is *ready to fire* in the state $z = (m, h)$, denoted by $z \xrightarrow{t}$, iff $t^- \leq m$ and $eft(t) \leq h(t)$. A transition $\hat{t}$, which is ready to fire in the state $z = (m, h)$, may fire yielding a new state $z' = (m', h')$, defined by $m' = m + \Delta\hat{t}$ and

$$h'(t) =: \begin{cases} \# & \text{iff} \quad t^- \not\leq m \\ h(t) & \text{iff} \quad t^- \leq m \wedge t^- \leq m' \wedge {}^\bullet t \cap {}^\bullet \hat{t} = \emptyset \\ 0 & \text{otherwise} \end{cases}.$$

Thus, the firing mode can be defined now as follow: A state $z = (m, h)$ is *changed* into the state $z' = (m', h')$ *by firing* the transition $\hat{t}$, denoted by $z \xrightarrow{\hat{t}} z'$. And, the state $z = (m, h)$ is *changed* into the state $z' = (m', h')$ *by the time duration* $\tau \in \mathbb{Q}_0^+$, denoted by $z \xrightarrow{\tau} z'$, iff $m' = m$ and the time duration τ is possible – formally: $\forall t \; (\; (t \in T \wedge h(t) \neq \#) \longrightarrow h(t) + \tau \leq lft(t)$ and $h'(t) := \begin{cases} h(t) + \tau & \text{iff} \quad t^- \leq m \\ \# & \text{iff} \quad t^- \not\leq m \end{cases}$). $z = (m, h)$ is called an integer-state iff $h(t)$ is an integer for each enabled transition t in m.

2.2 Time Petri nets and Turing Machines

It is well known that classical Petri nets are not equivalent to the Turing Machines. However, most of the time a small extension like inhibitor arcs, firing in maximal steps etc. makes the modified classical Petrinets already equivalent to the Turing machines. The proof can be done using register machines.

2.3 State Space

The state space of an arbitrary Time Petri net is the set of all reachable states of the net, starting at z_0. Of course, this set is in general infinite: Already, the set of all reachable states for a fixed m-marking is in general infinite (and dense). Nevertheless, it is possible to pick up some "essential" states so that a qualitative and quantitative analysis is possible. In [7] it is shown that the essential states are the integer-states.

The graph $RG_Z(z_o)$ is called the reachability graph of the TPN Z iff its vertices are the reachable integer-states and its edges are defined by the triples (z, t, z') and (z, τ, z') where $z \xrightarrow{t} z'$ and $z \xrightarrow{\tau} z'$, respectively. This graph is finite iff the set of the reachable markings of the net is finite. Moreover, this set is finite, if the set of reachable markings of the skeleton – the timeless net – is finite. The other direction is not true in general.

The state space of a given TPN Z can be defined recursively as follow:

Definition 4 (state space).
Let Z be a TPN. The set $StSp(Z)$ is called the state space of Z iff $StSp(Z) = \bigcup C$, where each set C is defined recursive as follow:

Basis: $C_0 := \{z \mid \exists\tau(\tau \in \mathbb{R}_0^+ \wedge z_0 \xrightarrow{\tau} z)\}$

Step: *Let C be already defined. Then C' is derived from C by firing $\hat{t}$*
(formally $C \xrightarrow{\hat{t}} C'$), iff
$C' := \{z \mid \exists z_1 \exists z_2 \exists\tau(z_1 \in C \wedge \tau \in \mathbb{R}_0^+ \wedge z_1 \xrightarrow{\hat{t}} z_2 \xrightarrow{\tau} z)\}$.

The sets C are called here classes of reachable states in Z.

3 Firing Sequences

3.1 Parametric description

In order to study transition sequences in a TPN we introduce the notion parametric description of a sequence as defined and ample studied in [7].

It is clear that a transitions sequence in a given TPN is a chronology of firing transitions.This means that it is a "what – when entity" in a TPN. Furthermore, it is also clear that even if the "what" is fixed the "when" is in general still variable. In order to mirror this fact we define the notion parametric description of a transition sequence.

In short, the parametric description of a sequence $\sigma = (t_1, ..., t_n)$ is a triple $\delta(\sigma) = [m_\sigma, \Sigma_\sigma, B_\sigma]$. The time that can/must elapse between the firing of two in σ successive transitions t_i and t_{i+1} is variable. We describe it with the variable x_i. Of course, x_i has to fulfill certain constrains, which are defined in B_σ. m_σ is the p-marking reached after the firing of σ. Σ_σ is a vector where the number of its components is equal to the number of thansitions in the net, and each component is a sum of variables. Σ_σ is a parametrically defined t-marking. B_σ is a set of inequalities. Σ_σ and B_σ describe parametrically all passible times for the enabled transitions in m_σ. The pair $(m_\sigma, \Sigma_\sigma)$ is a state, which is reachable after firing σ, when the variables x_i satisfy the inequalities in B_σ. For a detailed definition see [7].

3.2 Integer sequences

As already mentioned repeatedly, the paper [7] considers firing in TPN's. The main results achieved in this paper are important for this study and therefore we resume them shortly:

Let us consider an arbitrary TPN. Furthermore, we consider an arbitrary transition sequence $\sigma = (t_1, ..., t_n)$ and an arbitrary concrete realization (run) of it: Starting at the initial state of the net it is possible to fire the sequence by alterneting concrete elapsing times τ_i and transition firings. Thus, the run of σ has the form $\sigma(\tau) = (\tau_1, t_1, ..., \tau_n, t_n)$ and all τ_i's are real numbers.

Then it is **always** possible to **find** a further **concrete realization** (called **run**) $\sigma(\tau^*) = (\tau_1^*, t_1, ..., \tau_n^*, t_n)$ with:

(1) all elapsing times τ_i^* are **integers**

(2) for all i's the difference between τ_i and τ_i^* is always **smaller then 1**.
(3) for each transition t enabled after firing $\sigma(\tau^*)$ is true that the clock $h(t)$ shows an **integer** time (i.e. if $h(t) \neq \sharp$ then $h(t)$ is an integer) and
(4) the difference between the times showing by the clock of each transition t after firing $\sigma(\tau)$ and after firing $\sigma(\tau^*)$ is always **smaller then 1**. In particular,
   (a) it is possible to find such a run $\sigma(\tau^*)$, so that the time showing by each clock $h(t)$ (for enabled transition t) is always smaller then the time showing after firing $\sigma(\tau)$. In other words, the run $\sigma(\tau^*)$ can be chosen so, that the time shown by each clock $h(t)$ is exactly the **integer part** of the time shown by the same clock after the firing of the run $\sigma(\tau)$.
   (b) analogosly to prior item, it is possible to find such a run $\sigma(\tau^*)$ but the time showing by each clock $h(t)$ (for enabled transition t) is now always greater then the time showing after firing $\sigma(\tau)$. That means, the run $\sigma(\tau^*)$ can be chosen so, that the time shown by each clock $h(t)$ is exactly **1 plus the integer part** of the time shown by the same clock after the firing of the run $\sigma(\tau)$.
(5) the time difference between the time used for the firing of $\sigma(\tau)$ and the time used by $\sigma(\tau^*)$ is always **smaller then 1** (i.e. the difference between the sum of all τ_i^*'s and the sum of all τ_i's is smaller then 1). And particularly, in case of 4.a and 4.b it holds:
   (a) in case of 4.a. the time used by $\sigma(\tau)$ is smaller then the time used by $\sigma(\tau^*)$ and that means of course, that the first time here is the **integer part** of the second one.
   (b) in case of 4.b. the time used by $\sigma(\tau)$ is greater then the time used by $\sigma(\tau^*)$ and that means here, that the first time here is **1 plus the integer part** of the second one.

4 Reachability of states

It is clear that the reachability of an arbitrary state in an arbitrary TPN is undecidable due to of the equivalence between TPN's and Turing machines. Nevertheless, using necessary conditions, which are only we can prove the non-reachability of an arbitrary state in an arbitrary TPN.

The set of all reachable p-markings of an arbitrary TPN is a part of the set of all reachable markings of its skeleton. Thus, the most general necessary condition for the reachability of a p-marking in a TPN is the condition that the marking is reachable in its skeleton. Because the skeleton is a classical PN the last reachability problem is solvable.

When a TPN is bounded the reachability of a state is decidable. In order to see this, let as consider an arbitrary state $z = (m, h)$.

- If all clocks $h(t)$ are **integers**, then z is reachable iff it belongs to the reachability graph of the net. And this one is finite, because the TPN is bounded.

- In the case of all $h(t)$ are **non-negative rational numbers**, then the considered TPN can be transformed into a further TPN with the same structure. The time intervals now are derived from the old net by multiplication of all eft's and lft's with $r =$ LCM $\{h(t) \mid t$ enabled by $m\}$. The second TPN is bounded, too. The state z is reachable in the old TPN iff the state $z^* = (m, h^*)$ is reachable in the new one, where $h^*(t) = h(t) \cdot r$ for enabled transitions t and $h^*(t) = h(t)$ for disabled transitions. Obviously, the state z^* is an integer one. Therefore, z^* is a vertex in the reachability graph of the new TPN iff z^* is reachable in them. For more see [6].
- The last case is that all $h(t)$ are **real numbers**. Because the TPN is bounded, there is a finite number of vertices in the reachability graph with the marking m. Let $z^* = (m, h^*)$ be such a vertex. Now, we can find all path in the reachability graph from z_0 to z^*, and the paths do not contained any loops. In other words, we consider all paths, where each vertex appears once at most. The number of these paths is finite. For each such path σ we consider the parametric description. Thus, we obtain with B_σ and $\Sigma_\sigma(t) = h^*(t)$ a system of linear inequalities. It is solvable iff the state z^* is reachable in the TPN. This algorithm is NP-hard.

In the third case a pre-test can be done in linear time, which yields a necessary condition for the reachability of z. As a pre-test can be checked whether the states $\underline{z}$ and $\overline{z}$ belong to the reachability graph of the net. For the state $\underline{z} = (\underline{m}, \underline{h})$ holds, that $\underline{m} = m$ and $\underline{h}(t) = \lfloor h(t) \rfloor$ for all t enabled in m, for disabled t is $\underline{h}(t) = h(t)$. Analogously $\overline{z}$ is the state $(m, \overline{h})$ with $\overline{h}(t) = \lceil h(t) \rceil$ for enabled transitions and $\overline{h}(t) = h(t)$ for disabled. According to [7] it is true, that if z is reachable, than $\underline{z}$ and $\overline{z}$ are reachable, too. This property is true for unbounded nets, too. Obviously, the $\underline{z}$ and $\overline{z}$ are integer states and if they are reachable then they belong to the reachable graph of the net. Thus, if $\underline{z}$ or $\overline{z}$ are not reachable in the TPN Z than the state z is not reachable in Z, too.

5 Shortest and Longest Time Paths

In the quantitative analysis of TPNs the question "How much time elapses between the appearance of two p-markings" is basic one. In general a TPN is a model of a certain time dependent system. The minimal and the maximal time demand reflects time bounds for occurring events in the system. These bounds are important for ascertaining deadlines.

Generally a TPN is unbounded. Moreover, it is clear that in the case of bounded TPN methods from the graph theory can be used for analysis due to the finiteness and the structure of the reachability graph,

The shortest time path between two markings m_1 and m_2 is the time that elapse between two states z_1 and z_2. The p-marking of z_1 is m_1 and the p-marking of z_2 is m_2.If there is a run between z_1 and z_2 then z_1 is the last state with the p-marking m_1 and z_2 is the first state with the p-marking m_2 in the run. The shortest time path is the minimum over all runs. In other words, the shortest

path between m_1 and m_2 is the minimal time elapsing between the reachability of two states in the TPN which markings are m_1 and m_2, respectively. Using the results from [7], it is easy to see that the minimal time for reaching a state with a certain marking is an integer one and the achieved state is an integer state. Thus, the minimal time between the appearance of two markings is also integer and the corresponding states are integer states. Therefore, when the TPN is bounded, the states which are decisive for the computing of the shortest path belong to the reachability graph.

The longest time path between two p-markings is defined in an analogous manner to the shortest time path. The only difference is that z_1 is the first appearance of m_1 and z_2 – the last appearance of m_2. If there is a cycle starting with z_1 and before reaching z_2 then we say that longest time path is infinite. This definition is not the usual one for longest paths. Newertheless, such paths are important for the check of keeping deadlines. For more cf. [8].

The problems of shortest and longest time paths between two markings are the problems of finding the minimal and the maximal paths between two finite sets of states. Each of the both sets consists of all reachable integer states with the same p-marking. The problem of the shortest path can be solved by an "'all-pairs shortest path"'-algorithm (cf. [4]). The problem of the longest path can be solved by an similar defined "'all-pairs longest path"'-algorithm. The running time is polynomial.

5.1 Shortest and Longest Path in an arbitrary TPN

Let us consider an arbitrary TPN Z and two markings m_1 and m_2. If the net is unbounded then the reachability graph is infinite. Hence the both sets we spoke above, can be infinite. Therefore the number of sequences leading from the first to the second one can be also infinitely. In this case we can compute the minimal and the maximal time distance between m_1 and m_2 only if there is an additional information: the moments (states) of considering the p-markings, i.e. two states $z_1 = (m_1, h_1)$ and $z_2 = (m_2, h_2)$, and a transition sequence $\sigma = (t_{i_1}, ..., t_{i_k})$ with $\xrightarrow{\sigma}$. Solving two LPs we can compute the minimal and the maximal elapse of times between m_1 and m_2 in σ. The running time is polynomial.

5.2 Shortest and Longest Path in a bounded TPN

Let Z be a bounded TPN and $R_Z(z_0)$ be the reachability graph of Z. As shown in [6] this reachability graph is finite.

Now we consider two integer states and study the time distance between them.

Let z_1 and z_2 be two integer states in Z. Hence, each of them is a vertex in $R_Z(z_0)$. Following, the problems of finding shortest res. longest paths between two integer states is finding the shortest res. longest paths between two vertices in a weighted directed graph. In general, the shortest path can be find in polynomial time as shown e.g. in [8].

The longest path from z_1 to z_2 $lp(z_1, z_2)$ can be find in linear time. In the following we are going to present an algorithm for this:

(1) Remove all output edges of z_2 from $R_Z(z_0)$. The new graph let be called $\tilde{R}$ for short.
(2) Compute allstrongly connected components (SCCs) of $\tilde{R}$.
(3) Check if z_1 belongs to a SCC $Q_{z_1} = (V_1, E_1)$ with $\mid V_1 \mid \geq 2$.
if yes **then** go to (7) **else** go to (4).
(4) Compute the acyclic component graph $\tilde{R}^{SCC}$.
(5) Check in $\tilde{R}^{SCC}$ if it is possible starting at Q_{z_1} to reach a vertex P (P is a SCC in $\tilde{R}$ so let $P = (V_P, E_P)$ be) and $\mid V_P \mid \geq 2$.
if yes **then** go to (7) **else** go to (6).
(6) Compute longest path from Q_{z_1} to Q_{z_2}. STOP.
(7) Set $lp(z_1, z_2) = \infty$. STOP.

6 Conclusion

In this note we considered the decidability of the reachable states in a TPN. For this purpose we defined a parametric description of transition sequences and introduced integer sequences. Afterwards, we discussed a computation of the shortest and longest paths between two markings. Subsequently, we spoke about computation of the shortest and longest paths between two states. Eventually, we presented an algorithm for computing of the longest path between two (integer) states with a linear running time.

References

1. B. Berthomieu and M. Diaz. Modeling and Verification of Time Dependent Systems Using Time Petri Nets. In *Advances in Petri Nets 1984*, volume 17, No. 3 of *IEEE Trans. on Software Eng.*, pages 259–273, 1991.
2. B. Berthomieu and M. Menasche. An Enumerative Approach for Analyzing Time Petri Nets. In *Proceedings IFIP Congress*, September 1983.
3. H. Boucheneb and G. Berthelot. Towards a simplified building of time petri net reachability graphs. In *Proceedings of Petri Nets and Performance Models PNPM 93*, Toulouse France, 1993. IEEE Computer Society Press.
4. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, second edition, 2001.
5. Philip M. Merlin. *A Study of the Recoverability of Computing Systems*. PhD thesis, University of California, Computer Science Dept., Irvine, January 1974.
6. L. Popova. On Time Petri Nets. *J. Inform. Process. Cybern. EIK 27(1991)4*, pages 227–244, 1991.
7. L. Popova-Zeugmann and D. Schlatter. Analyzing Path in Time Petri Nets. *Fundamenta Informaticae (FI) 37, IOS Press, Amsterdam*, pages 311–327, 1999.
8. L. Popova-Zeugmann and M. Werner. Extreme runtimes of schedules modelled by time petri nets. *Fundamenta Informaticae (FI) 67, IOS Press, Amsterdam*, pages 163–174, 2005.