International Journal of Computer and Communication Technology

Volume 2 | Issue 4 Article 9

October 2011

Time Analysis of the State Space of Real-time Preemptive Systems

Abdelkrim Abdelli
LSI laboratory - Computer Science department - USTHB university of Algiers, Abdelli@lsi-usthb.dz

Time Analysis of the State Space of Real-time Preemptive Systems.

Abdelkrim Abdelli
LSI laboratory - Computer Science department - USTHB university of Algiers.
Abdelli@lsi-usthb.dz

Abstract

We present in this paper an algorithm making it possible an efficient time analysis of the state space of preemptive real time systems modeled using Time Petri Nets with inhibitor arcs. For this effect, we discuss how to determine from the reachability graph linear and quantitative properties of the remote model. Then, we propose an algorithm to compute an approximation of the minimal and the maximal time distances of any firing sequence. Contrarily to other techniques, our algorithm enjoys a linear complexity time cost and can be performed on the fly when building the reachability graph without requiring to extend the original model with observers.

1 Introduction

Preemptive systems are systems whose tasks have strict temporal constraints and which can be stopped for a while and resumed afterwards (*stopwatch mechanism).* To prove the correctness of such systems, various models, as extensions of Time Petri Nets (7) have been proposed in the literature (5)(11)(9). For instance, in (11) the authors defined the *ITPN* (*Inhibitor Time Petri Nets)* model, wherein the progression and the suspension of time is driven by using standard and inhibitor arcs. Then, the state space of the model is computed by applying the state class graph method (3) in the same way as for a TPN. Each class E of this graph is a pair consisting of a marking M and a set of inequalities D. However, unlike in *TPN* where D is always given in the form of a DBM *(Difference Bound Matrix)* system (6), for an *ITPN* the system D can enjoy a polyhedral form which can not be encoded in DBM. In this case, D needs complex data structures to be represented in memory and requires a much higher time to be solved[1]. As a result, the exact state class graph computation algorithm (9) has reported memory overflows and prohibitive calculation times. To circumvent this issue, DBM approximation techniques (1)(5)(11) have proposed to over-approximate the system D by the tightest DBM subsystem including it. These approaches make it possible to build efficiently in a lesser time, a graph which can however derive additional firing sequences that are not accessible in the exact graph. This construction makes it possible to preserve a subset of properties than can be sufficient to model-checking the system.

Within this contest, one of the main property of interest is to check over *WCRT* or *BCRT* (Worst and Best case response times) of an action or a run. For this effect, the authors in (4)(12)(11) have proposed to extend the original model with an observer containing additional places and transitions modeling the quantitative property. Then they need to compute the reachabiliy graph of it in order to workout whether the property holds or not. This method is quite costly as it requires, for each property to check, to extend the net with the appropriate observer before computing its reachability graph wherein the property is worked out.

In (5), the authors proposed an interesting method for quantitative timed analysis. They compute first the DBM approximation of the graph. Then, given an untimed transition sequence from the over-approximated state class graph, they can obtain the feasable timings between the firing of the transitions of the sequence as the solution of a linear programming problem. In particular, if there is no solution, the transition sequence has been introduced by the over-approximation and can be cleaned up, otherwise the solution set allows to check timed properties on the firing times of transitions. However, this method needs, for each sequence analysis, an exponential complexity time as a result of solving a linear programming problem.

Within this context, we propose in this paper an algorithm making it possible the real time analysis of preemptive systems modeled by using the *ITPN* model. This consists in computing an over approximation of the minimal and the maximal time distances of any firing sequence of the graph in a linear complexity time. Moreover, our algorithm is performed only once and can be either applied on the fly when building the graph, or after its construction without requiring to extend the *ITPN* with observers.

The remainder of this paper is organized as follows: In *Section 2*, we present the syntax and the formal semantics of the *ITPN* model. In *Section* 3, we discuss of the state class graph method as well as its *DBM* over

[1]The complexity of computing a class is exponential in the number of variables whereas it is polynomial for a *DBM* system.

approximation. In *Section* 4, we show how to determine the properties of interest from the graph and then we present our algorithm to compute the quantitative properties of the model.

2 Time Petri Net with inhibitor arcs

Time Petri nets with inhibitor arcs (ITPN) (11) extends time Petri nets(7) to *Inhibitor arcs* and stopwatches (8). Formally, an $ITPN$ is defined as follows:

Definition 1. *An ITPN is given by the tuple* (P, T, B, F, M^0, I, IH) *where:* P *and* T *are respectively two nonempty sets of places and transitions;* B *is the backward function* [2] $: B : P \times T \longrightarrow \mathbb{N} = \{0, 1, 2, ..\}$; F *is the forward function* $F : P \times T \longrightarrow \mathbb{N}$; M^0 *is the initial marking function* $M^0 : P \longrightarrow \mathbb{N}$ *; I is the delay mapping* $I : T \longrightarrow \mathbb{Q}^+ \times \mathbb{Q}^+ \cup \{\infty\}$, *where* $\mathbb{Q}^+$ *is set of non negative rational. We write* $I(t) = [tmin(t), tmax(t)]$ *such that* $0 \leq tmin(t) \leq tmax(t)$; $IH : P \times T \longrightarrow \mathbb{N}$ *is the inhibitor arc function; there is an inhibitor arc connecting the place p to the transition t, if* $IH(p, t) \neq 0$.

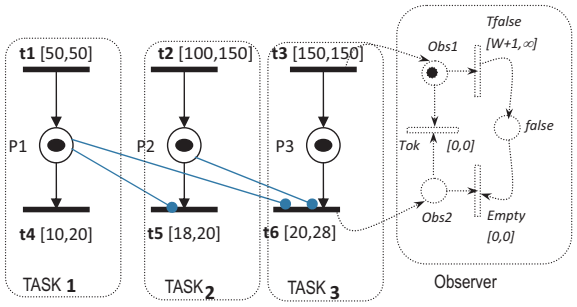

Figure 1. *ITPN modeling two periodic tasks and a sporadic one.*

For instance, let us consider the $ITPN$ model, drawn in continuous lines in *Figure 1.* This example describes periodical and sporadic tasks executed in parallel (5). This example models three independent tasks that are conflicting for a common resource (CPU): Two periodic tasks 1 and 3 (of period 50 and 150 time units), and one sporadic task with a minimum and maximum inter-arrival times of $[100, 150]$. The task 1 (modeled by the transitions t_1 and t_4), has a higher priority than that of two other tasks, and the sporadic task has a higher priority than that of the third task. The priorities are modeled by using inhibitor arcs. The *inhibitor arcs* are the arcs ended by a circle that connect the places p_1, p_1 and p_2 to respectively the transition t_5, t_6 and t_6. Initially, the place p_1 is marked; hence t_4 is enabled and activated. However, t_5 and t_6 are enabled but inhibited. The transition t_5 remains inhibited as long as there will exist a token in place p_1. However t_6 still remain inhibited as long as both the places p_1 and p_2 are marked. For more details, the formal semantics of the $ITPN$ model is introduced in the next section.

Let $RT := (P, T, B, F, M_0, I, IH)$ be an $ITPN$.

- We call a *marking* the mapping, noted M, which associates with each place a number of tokens: $M : P \rightarrow \mathbb{N}$.

- A transition t is said to be *enabled* for the marking M, if $\forall p \in P, B(p, t) \leq M(p)$; the number of tokens in each input place of t is greater or equal to the valuation of the arc connecting this place to the transition t. Thereafter, we denote by $Te(M)$ the set of transitions *enabled* for the marking M.

- A transition t is said to be *inhibited* for a marking M, if it is *enabled* and if there exists an inhibitor arc connected to t such that the marking satisfies its valuation $(t \in Te(M)) \wedge \exists p \in P, 0 < IH(p, t) \leq M(p)$. We denote by $Ti(M)$ the set of transitions that are *inhibited* for the marking M.

- A transition t is said to be *activated* for a marking M, if it is enabled and not inhibited, $(t \in Te(M)) \wedge (t \notin Ti(M))$; we denote by $Ta(M)$ the set of transitions that are *activated* for the marking M.

- Let M be a marking ; two transitions t_i and t_j enabled for M are said to be *conflicting* for M, if $\exists p \in P, \quad B(p, t_i) + B(p, t_j) > M(p)$.

For instance, let us consider again the $ITPN$ of *Figure 1*; its initial marking is equal to $M^0 : \{p_1, p_2, p_3\} \rightarrow 1$; the sets of enabled, inhibited, and activated transitions for M^0 are respectively $Te(M^0) = \{t_1, t_2, t_3, t_4, t_5, t_6\}$, $Ti(M^0) = \{t_5, t_6\}$, and $Ta(M^0) = \{t_1, t_2, t_3, t_4\}$.

The semantics of an $ITPN$ is defined as a labeled transition system, as follows:

Definition 2. *The semantics of an ITPN is defined as a labeled transition system* $ST = (\Gamma, e^0, \rightarrow)$, *such that:*

- Γ *is the set of accessible states: Each state, noted e, pertaining to* Γ *is a pair* (M, V) *where* M *is a marking and* V *is a valuation function that associates with each enabled transition t of* $Te(M)$ *a time interval that gives the range of relative times within which t can be fired. Formally we have :* $\forall t \in Te(M), \quad V(t) := [x(t), y(t)]$

- $e^0 = (M^0, V^0)$ *is the initial state, such that:* $\forall t \in Te(M^0), \quad V^0(t) := I(t) := [tmin(t), tmax(t)]$.

- $\rightarrow \in \Gamma \times (T \times \mathbb{Q}^+) \times \Gamma$ *is a transition relation, such that we have* $((M, V), (t_f, \underline{t_f}), (M^{\uparrow}, V^{\uparrow})) \in \rightarrow$, *iff:*

[2] $\mathbb{N}$ denotes the set of positive integers. In the graphical representation, we represent only arcs of non null valuation, and those valued 1 are implicit.

(i) $t_f \in Ta(M)$.

(ii) $x(t_f) \leq \underline{t_f} \leq \underset{\forall t \in Ta(M)}{MIN} \{y(t)\}$.

and we have:

$\forall p \in P,\ M^{\uparrow}(p) := M(p) - B(p, t_f) + F(p, t_f)$.

$\forall t \in Te(M^{\uparrow})$

if $t \notin New(M^{\uparrow})$*:*

if $t \in Ta(M)$

$[x^{\uparrow}(t),\ y^{\uparrow}(t)] := [MAX(0,\ x(t) - \underline{t_f}),\ \ y(t) - \underline{t_f}]$

if $t \in Ti(M)$

$[x^{\uparrow}(t),\ y^{\uparrow}(t)] := [x(t),\ \ y(t)]$

if $t \notin New(M^{\uparrow})$

$[x^{\uparrow}(t),\ y^{\uparrow}(t)] := I(t) = [tmin(t),\ tmax(t)]$

- *where* $New(M^{\uparrow})$ *denotes the set of transitions newly enabled for the marking* $M^{\uparrow}$*. These transitions are those enabled for*$M^{\uparrow}$ *and not for* M*, or those enabled for* $M^{\uparrow}$ *and* M *but are conflicting with* t_f *for the marking* M*. Otherwise, an enabled transition which does not belong to* $New(M^{\uparrow})$ *is said to be persistent.*

If t is an enabled transition for a state e, we note $\underline{t}$ a clock associated with t that takes its values in $\mathbb{Q}^+$. t measures the residual time of the transition t relatively to the instant where the state e is reached. The time progresses only for activated transitions, whereas it is suspended for inhibited transitions. Therefore, a transition t_f can be fired at relative time $\underline{t_f}$ from an accessible state e, if (i) t_f is *activated* for the marking M, and if (ii) the time can progress within the firing interval of t_f without overtaking those of other activated transitions. After firing t_f the accessible state, noted $e^{\uparrow}$, is obtained:

- by consuming a number of tokens in each input place p of t_f (given by the value $B(p, t_f)$), and by producing a number of tokens in each output place p of t_f (given by the value $F(p, t_f)$);
- by shifting the interval of a persistent activated transition with the value of the firing time of t_f. However, the intervals of persistent inhibited transitions remain unchanged. Finally, a newly enabled transition is assigned its static firing interval.

Similarly as for TPN, the behavior of an $ITPN$ can be defined as a sequence of pairs (t, δ), where t is a transition of the net and $\delta \in \mathbb{Q}^+$. Therefore, the sequence $S^t = ((t_f^1, \delta^1), (t_f^2, \delta^2), .., (t_f^n, \delta^n))$ denotes that t_f^1 is firable after δ^1 time units, then t_f^2 is fired after δ^2 time units and so on, such that t_f^n is fired after the absolute time $\sum_{i=1}^{n} \delta^i$. Moreover, we often express the behavior of the net as an *untimed sequence,* denoted by S, obtained from a timed sequence S^t by removing the firing times: If $S^t = ((t_f^1, \delta^1), (t_f^2, \delta^2), ..., (t_f^n, \delta^n))$, then $S = (t_f^1, t_f^2, .., t_f^n)$. As the set of time values is assumed to be dense, the model ST is infinite. In order to analyze this model, we need to compute an abstraction of it that saves the most properties of interest. The *state class graph* preserves the untimed sequences of ST, and makes it possible to compute a finite graph in almost all cases.

3 $ITPN$ state class graph

For a $TPN(7)$, the state class graph method (3) allows to compute a symbolic graph that preserves chiefly the linear properties of the model. Likewise, this construction can be applied to an $ITPN$. This will consists in gathering in a same class all the states accessible after firing the same untimed sequence; all the states of a same class have the same marking M. Hence, a class is defined by the pair (M, D) where M is the common marking of all the states of the class, and D is a set of inequalities encoding the firing space of the class. More formally, a class of an $ITPN$ (4) is defined as follows:

Definition 3. *Let* $ST = (\Gamma, e^0, \rightarrow)$ *be the LTS associated with an* $ITPN$*. A class of states of an* $ITPN$*, denoted by* E*, is the set of all the states pertaining to* Γ *that are accessible after firing the same untimed sequence* $S = (t_f^1, .., t_f^n)$ *from the initial state* e^0*. A class* E *is defined by* (M, D)*, where* M *is the marking accessible after firing* S*, and* D *is the firing space encoded as a set of inequalities.*
For $Te(M) = \{t_1, .., t_s\}$*, we have :* $D = \widehat{D} \wedge \widetilde{D}$

$$\widetilde{D} := \begin{cases} \bigwedge_{i \neq j} & (\underline{t_j} - \underline{t_i} \ \leq \ d_{ij}) \\ \bigwedge_{i \leq s} & (d_{i\bullet} \leq \ \underline{t_i} \ \leq d_{\bullet i}) \end{cases}$$
$$\text{with } (t_j, t_i) \in Te(M)^2 \quad d_{ij} \in \mathbb{Q} \cup \{\infty\}, \quad d_{\bullet i} \in \mathbb{Q}^+ \cup \{\infty\}, d_{i\bullet} \in \mathbb{Q}^+$$

$$\widehat{D} := \bigwedge_{k=1..p} \quad (\alpha_{1k}\underline{t_1} + .. + \alpha_{sk}\underline{t_s} \ \leq \ d_k)$$
$$\text{with } d_k \in \mathbb{Q} \cup \{\infty\},\ (\alpha_{1k}, .., \alpha_{sk}) \in \mathbb{Z}^s \ \text{ and}^3$$
$$\forall k, \exists (i, j), (\alpha_{ik}, \alpha_{jk}) \notin \{(0, 0), (1, -1)\}$$

We denote by the element $\{\bullet\}$ the earliest instant at which the class E is reached. Therefore, the value of the clock $\underline{t_i}$ expresses the time relative to the instant $\bullet$ at which the transition t_i can be fired. Thus for each valuation ψ satisfying the system D, it corresponds a unique state $e = (M, V)$ accessible in ST after firing the sequence S.

In case of a TPN, the system D is reduced to the subsystem $\widetilde{D}$. The inequalities of the latter have a particular form, called DBM (*Difference Bound Matrix*)(6). For $TPN's$, the firing space of a class can always be encoded as a DBM system. This form makes it possible to apply an efficient algorithm to compute a class, whose overall complexity is $O(m^3)$, where m

[3] Z denotes the set of relative integers.

is the number of enabled transitions. However, for TPN augmented with stopwatches, the set of valuations pertaining to a given class can not be encoded anymore with $DBMs$. Actually, inequalities of general form (called also *polyhedra)*, are needed to encode the firing space of a class. These constraints given by the subsystem $\widehat{D}$, induce a higher complexity that can be exponential in the worst case.

To tackle this issue, the DBM over approximation technique has been proposed as an alternative solution to analyze preemptive real time systems (11)(5)(1). This approach consists in cutting off the inequalities of the subsystem $\widehat{D}$ when the latter appears in D; it thereby keeps only those of the subsystem $\widetilde{D}$ to represent an over approximation of the space of D. This solution makes it possible to build a less richer graph than the exact one, but nevertheless with lesser expenses in terms of computation time and memory usage. Besides, the DBM over-approximation may compute an infinity of unreachable markings while the exact construction is indeed bounded. Formally, a DBM approximated class graph can be defined as follows:

Definition 4. *Let R be an ITPN. The DBM-approximated class graph of R, noted $\widetilde{GR}$, is the typle $(\widetilde{CE}, \widetilde{E^0}, \longmapsto)$ where: $\widetilde{CE}$ is a set of approximated classes such that each approximated class, noted $\widetilde{E}$, is a pair $(M, \widetilde{D})$ such that : M is a marking and $\widetilde{D}$ is a DBM system ; $\widetilde{E^0} = (M^0, \widetilde{D^0})$ is a special class of $\widetilde{CE}$ called the initial class ; $\longmapsto$: is relation between classes defined on $\widetilde{CE} \times T \times \widetilde{CE}$.*

In the sequel, we encode the system $\widetilde{D}$ as a square matrix where each line and corresponding column, are indexed by an element of $Te(M) \cup \{\bullet\}$. In concrete terms, we have:

$\forall (t_i, t_j) \in Te(M)^2 \wedge (t_i \neq t_j)$, $\widetilde{D}[\bullet, t_i] := \widetilde{d_{\bullet i}}$; $\widetilde{D}[t_i, \bullet] := -\widetilde{d_{i\bullet}}$; $\widetilde{D}[t_i, t_j] := \widetilde{d_{ij}}$; $\widetilde{D}[t_i, t_i] := 0$; $\widetilde{D}[\bullet, \bullet] := 0$.

Taking on the previous definition, if $E = (M, D)$ is a class accessible in the exact graph, noted GR, then the class $\widetilde{E} = (M, \widetilde{D})$ is an overapproximation of E, since the space of states of E is included in that of $\widetilde{E}$. Hence, by substituting $\widetilde{E}$ for E in the graph GR, it results that the class $\widetilde{E}$ may derive additional sequences that are not firable indeed in GR from E. We thereby obtain an overapproximation of the graph GR. This abstraction of the state space of an $ITPN$ allows to preserve a subset or properties which is, in general, sufficient for the analysis of the model. Furthermore, it should be noticed that the finitness of the exact state class graph is undecidable even for bounded nets (4). However, the graph obtained by DBM approximation is ensured to be finite when the net is bounded. This makes it possible to compute a finite approximation when the exact one does not terminate.

4 State class graph analysis

The aim of computing the state class graph is to deduce the main important properties of the model (e.g. reachability, deadlock, liveness,...etc). Concerning the exact graph GR, all the linear properties of the model (those that could be checked by using linear logics like LTL), are preserved. Therefore, to check whether an $ITPN$ satisfies a given property, it requires to check over it on its graph GR. On the other hand, as concerns the approximated graph $\widetilde{GR}$, the construction preserves a subset of properties of the $ITPN$; the main ones are given next:

- Any marking M, sequence S and state e accessible in GR, are accessible in $\widetilde{GR}$.
- Any marking M, sequence S and state e inaccessible in $\widetilde{GR}$, are not accessible in GR.

From previous relations, one can check over properties of *safety*: "*something bad never happens"*. Therefore, if the graph $\widetilde{GR}$ enjoys the later property, then the graph GR satisfies it too. Conversely, if the property is false in $\widetilde{GR}$ (i.e, the system is not safe), then we can extrapolate and state that there is a probability that the system is not safe with the risk to be pessemistic[4]. Moreover, schedulability and evaluation of quantitative properties are typical properties of interest for such applications. A schedulability consists in checking for instance that an occurrence of a task is always processed before the arrival of a new one; this property is satisfied if the marking is *Safe*. As concerns the example of *Figure 1,* the exact graph as as well as the over approximated one enjoy safe markings; the markings of the graphs are 1-bounded. Furthermore, to check over a quantitative property on the graph, we can extend the $ITPN$ with an observer (containing new places and transitions), making it possible to model this property. Then, we need to verify whether the marking of some places of the observer are reachable or not in the graph (11)(12)(4). The *Figure 1* shows an observer (depicted in dotted lines), to check the following property: '*The task 3 (t_3 and t_6) is always executed in less than W time units'*. The property holds if the place *false* is never marked or if the model does not fire the transition *Tfalse*. This property becomes false in the exact graph from $W = 89$ and from $W = 137$ for the over-approximated graphs. Hence, the exact value for the $WCRT$ (*Wort Case Response Time*) for this task is 88, while the approximated value is 136.

However, this technique is costly since it requires to compute the reachability graph of the extended model in order to check over the quantitative property[5]. To tackle this issue, we propose hereafter an efficient algorithm that allows to compute an approximation of

[4]We may find an additional path $\widetilde{GR}$ that does not satisfy a safety property, while all paths in GR satisfy it.

[5]Each quantitative property is modeled by a different observer; this implies to run for each a new graph construction

the maximum and the minimum time distances of any firing sequence in a linear complexity time.

4.1 Minimal and maximal time distances of a firing sequence

In the sequel, the notations assume that the approach applies to the approximated graph $\widetilde{GR}$. However, the technique can be applied too to the exact graph GR. With this intention, it needs to consider for each class E accessible in GR the subsystem $\widetilde{D}$ in place of the general polyhedral system D. Considering a given sequence included in both graphs $\widetilde{GR}$ and GR, the computed distances are exact regarding the graph $\widetilde{GR}$ whereas they are overapproximated for the exact graph GR.

So, let us consider the firing of a sequence of transitions $S_i^n = \left(t_f^{i+1}, .., t_f^n\right)$; S_i^n describes a path in the graph $\widetilde{GR}$ going from the node representing the class $\widetilde{E^i}$ to the node which represents the class $\widetilde{E^n}$. Therefore, the time distance of the sequence $(t_f^{i+1}, .., t_f^n)$ is given by the sum $\underline{t_f^{i+1}} + .. + \underline{t_f^n}$, and the minimum and the maximum of the latter are worked out within the firing space of the sequence S_i^n, defined as follows:

Definition 5. *Let $\widetilde{E^n}$ be a class accessible in $\widetilde{GR}$ from the class $\widetilde{E^i}$ after firing the sequence $S_i^n = \left(t_f^{i+1}, .., t_f^n\right)$. We denote by space$(S_i^n)$ the firing space defined on $(\mathbb{Q}^+)^{n-i}$ of vectors $(\theta^{i+1}, .., \theta^n)$ such that the sequence S_i^n can be fired at relative times $\underline{t_f^{i+1}} = \theta^{i+1}, .., \underline{t_f^n} = \theta^n$.*

$$space(S_i^n) := \left\{ \begin{array}{l} \left(\underline{t_f^{i+1}}, .., \underline{t_f^n}\right) := (\theta^{i+1}, .., \theta^n) \in (\mathbb{Q}^+)^{n-i} \mid \\ \forall e^n \in \widetilde{E^n}, \forall j \in \{i, .., n-1\} \quad \exists e^j \in \widetilde{E^j}, \\ \quad e^i \xrightarrow{t_f^{i+1}} .. e^j \xrightarrow{t_f^{j+1}} .. \xrightarrow{t_f^n} e^n \end{array} \right\}$$

A vector $(\theta^{i+1}, .., \theta^n)$ is an admissible solution if there exists a state $e^i = (M^i, V^i)$ in $\widetilde{E^i}$ which can fire sequentially the transition t_f^{i+1} at relative time $\underline{t_f^{i+1}} := \theta^{i+1}$, t_f^{i+2} at relative time $\underline{t_f^{i+2}} := \theta^{i+2}$, .., and finally the transition t_f^n at relative time $\underline{t_f^n} := \theta^n$ to reach the state $e^n = (M^n, V^n)$ accessible in $\widetilde{E^n}$. Moreover, as a class contains all the states accessible after firing the same sequence of transitions, therefore for each valuation of $(\underline{t_f^{i+1}}, .., \underline{t_f^n})$ taken from $space(S_i^n)$, it corresponds a unique state among those accessible in $\widetilde{E^n}$. Therefore, the space $space(S_i^n)$ determines also the space of the class $\widetilde{E^n}$. Hence the next definition shows how we can express the firing space of the class $\widetilde{E^n}$ in terms of that of the class $\widetilde{E^{n-1}}$.

Definition 6. *The space $space(S_i^n)$ can be written in terms of $space(S_i^{n-1})$, as follows: If $space(S_i^{n-1}) \neq \varnothing$, then $space(S_i^n) :=$*

$$\left\{ \begin{array}{l} (\theta^{i+1}, .., \theta^n) \mid \left(\underline{t_f^{i+1}}, .., \underline{t_f^{n-1}}\right) := (\theta^{i+1}, .., \theta^{n-1}) \in space(S_i^{n-1}) \wedge \\ \left(\underline{t_f^n} = \theta^n\right) \wedge \left(x(t_f^n) \leq \underline{t_f^n} \leq \underset{\forall t \in Ta(M)}{MIN} \{y(t)\}\right) \end{array} \right\}$$

Put in another way, if t_f^n is firable within $space(S_i^{n-1})$, then $space(S_i^n)$ represents all the vectors of $space(S_i^{n-1})$ (states of $\widetilde{E^{n-1}}$), that satisfy the firing condition of t_f^n and the restriction of the space $space(S_i^n)$ to the vectors $\left(\underline{t_f^{i+1}}, .., \underline{t_f^{n-1}}\right)$ is the sub set of $space(S_i^{n-1})$ that satisfies these conditions.

Therefore, thanks to the introduction of the firing space, the minimum and the maximum time distances of a sequence S_i^n can be defined as follows:

Definition 7. ***(Time distance function)*** *Let $\widetilde{E^n}$ be a class accessible in $\widetilde{GR}$, after firing the sequence $S_i^n = \left(t_f^{i+1}, .., t_f^n\right)$. For point (n), we define the time distance function, noted DS_n, that computes the minimum and the maximum time distances of all subsequences starting from points $i \in \{0, .., n\}$ up to point (n).*

$$DS_n : (\{0, .., n\} \cup Te(M))^2 \longrightarrow \mathbb{Q} \cup \{\infty\}$$

$$\forall i \in \{0, .., n-1\}, \qquad \forall t \in Te(M)$$

$$DS_n[i, n] := \underset{space(S_i^n)}{MAX} \{\underline{t_f^{i+1}} + .. + \underline{t_f^n}\};$$

$$DS_n[n, i] := -\underset{space(S_i^n)}{MIN} \{\underline{t_f^{i+1}} + .. + \underline{t_f^n}\};$$

$$DS_n[i, t] := \underset{space(S_i^n)}{MAX} \{\underline{t_f^{i+1}} + .. + \underline{t_f^n} + y^n(t)\};$$

$$DS_n[t, i] := -\underset{space(S_i^n)}{MIN} \{\underline{t_f^{i+1}} + .. + \underline{t_f^n} + x^n(t)\};$$

$$DS_n[i, i] := 0; \qquad DS_n[n, n] := 0.$$

Each coefficient of the function DS_n needs to find out the vector $(\theta^{i+1}, .., \theta^n)$ of $space(S_i^n)$ so that the related expression is optimized. Hence if t is an enabled transition for $\widetilde{E^n}$, then $DS_n[t, i]$ (respectively, $DS_n[i, t]$), represents the opposite value of the minimum time distance from the firing point (i) up to the minimum bound of the transition t (respectively, the maximum time distance from the firing point (i) up to the upper bound of the transition t). Further, $DS_n[i, n]$ (respectively, $DS_n[n, i]$), computes the maximum time distance (respectively, the opposite value of the minimum time distance), between the firing points (i) and (n). The computation formulae of the function DS^n are performed recursively by making projections on previous firing spaces, as defined in next proposition.

Proposition 1. *Let $\widetilde{E^{n-1}} = (M^{n-1}, \widetilde{D^{n-1}})$ be a class accessible in $\widetilde{GR}$, from the class $\widetilde{E^i} = (M^i, \widetilde{D^i})$ after firing the sequence $S_i^{n-1} = \left(t_f^{i+1}, .., t_f^n\right)$, and let $\widetilde{E^n} = (M^n, \widetilde{D^n})$ be a class accessible from $\widetilde{E^{n-1}}$ after firing the transition t_f^n. The function DS_n associated with $\widetilde{E^n}$ is computed recursively from DS_{n-1}, as follows:*

$DS_n[n, n] := 0$; $DS_n[n, t] := \widetilde{D^n}[\bullet, t]$; $DS_n[t, n] := \widetilde{D^n}[t, \bullet]$.

$\forall i \in \{0, .., n-1\}$,

$$DS_n[i,n] := \underset{\forall t \in Ta(M^n)}{MIN} \{DS_{n-1}[i,t]\};$$
$$DS_n[n,i] := DS_{n-1}[t_f^n, i].$$

$\forall i \in \{0,..,n-1\}, \quad \forall t \in Te(M^n)$

If t is persistent

If $t \notin Ti(M^{n-1})$ (t is not inhibited for the point $n-1$)

$$DS_n[i,t] := MIN(DS_{n-1}[i,t],\ DS_n[i,n] + \widetilde{D^n}[\bullet, t]).$$

$$DS_n[t,i] := MIN(DS_{n-1}[t,i],\ DS_n[n,i] + \widetilde{D^n}[t, \bullet]).$$

If $t \in Ti(M^{n-1})$ (t is inhibited for the point $n-1$)

$$DS_n[i,t] := MIN \left\{ \begin{array}{l} DS_{n-1}[i,t] + \underset{\forall t' \in Ta(M^{n-1})}{MIN} \left\{ \widetilde{D^{n-1}}[\bullet, t'] \right\} \\ DS_n[i,n] + \widetilde{D^n}[\bullet, t] \end{array} \right.$$

$$DS_n[t,i] := MIN \left\{ \begin{array}{l} DS_{n-1}[t,i] + \widetilde{D^{n-1}}[t_f^n, \bullet] \\ DS_n[n,i] + \widetilde{D^n}[t, \bullet] \end{array} \right.$$

If t is newly enabled.

$$DS_n[i,t] := DS_n[i,n] + tmax(t)$$
$$DS_n[t,i] := DS_n[n,i] - tmin(t)$$

To compute the minimum and maximum time distances of the sequence S_i^n we need to scroll up the path starting from the node representing the class $\widetilde{E^i}$ to the node representing the class $\widetilde{E^n}$.

The algorithm starts at point (i) by computing the elements $DS_i[i,t]$ and $DS_i[t,i]$. After that, for each intermediate node $j \in [i+1, n-1]$, it proceeds first to compute recursively the elements $DS_i[j,i]$ and $DS_i[i,j]$ before $DS_i[i,t]$ and $DS_i[t,i]$. Once the node (n) is reached, it has only to determine the coefficients $DS_n[n,i]$ and $DS_n[i,n]$. The latter are respectively the opposite value of the minimal time distance and the maximal time distance between the nodes (i) and (n). However, it should be noticed that previous values are over approximated relatively to the exact graph GR, namely the interval $[-DS_n[n,i], DS_n[i,n]]$ that we compute with our algorithm embeds the exact one. Moreover, the complexity of computing the last interval depends on the length of the sequence $(n-i)$, and is assessed to $o((n-i)\times(m))$ where $m = \frac{\sum_{j=i}^{n-1} |Te_j|}{n-i}$ is the average number of transitions enabled for an intermediate node (j) in the sequence S_i^n.

The application of our technique to the example of *Figure*.1 allows to check over any quantitative property without requiring to exetend the original model with an observer. By the way, it determines the same WCRT for the previous property, namely 136 whatever it is applied on the exact or the approximated graph. Moreover, it should be noticed that the function DS_n can be saved along in the expression of a class which makes it possible to check afterwards over any other quantitative property without requiring further computations.

5 Conclusion

We have proposed in this paper an algorithm making it possible an efficient time analysis of the state space of preemptive real time systems modeled using Time Petri Nets with inhibitor arcs. For this effect, we discussed how to determine from the reachability graph linear and quantitative properties of the remote model. Then, we have proposed an algorithm to compute an approximation of the minimal and the maximal time distances of any firing sequence. Contrarily to other existing techniques, our algorithm enjoys a linear complexity time cost and can be performed on the fly when building the reachability graph without requiring to extend the original model with observers.

References

[1] A.Abdelli , "Optimisation de la construction d'une approximation de l'espace d'état des systèmes préemptifs". In Journal Technique et sciences Informatiques, Hermes Lavoisier editions, VOL 28:9 2009. pp.1143-1170.

[2] Avis, D., K. Fukuda and S. Picozzi, On canonical representations of convex polyhedra. First International Congress of Mathematical Software (2002), pp. 350–360.

[3] B. Berthomieu, and M. Diaz. "Modeling and verification of time dependant systems using Time Petri Nets". IEEE TSE, 17(3):(259-273).

[4] B. Berthomieu, D. Lime, O. H. Roux, F.Vernadat: Reachability Problems and Abstract State Spaces for Time Petri Nets with Stopwatches. Discrete Event Dynamic Systems 17(2): 133-158 (2007).

[5] G. Bucci, A. Fedeli, L. Sassoli, and E.Vicario. Timed State Space Analysis of Real-Time Preemptive Systems. IEEE TSE, Vol 30, No. 2, 2004.

[6] Dill, D.L.: Timing assumptions and verification of finite-state concurrent systems; Workshop Automatic Verification Methods for Finite-State Systems. Vol 407. (1989) 197-212.

[7] P. Merlin. "A study of the recoverability of computer system". PhD thesis Dep. Comp. Science, Uni. California, Irvine, 1974.

[8] F. Cassez and K.G. Larsen. The Impressive Power of Stopwatches. LNCS, vol. 1877, pp. 138-152, 2000.

[9] D.Lime, and O.H.Roux. Expressiveness and analysis of scheduling extended time Petri nets. In 5th IFAC International Conference on Fieldbus Systems and their Applications, 2003.

[10] M. Magnin, D. Lime, O. H. Roux: An Efficient Method for Computing Exact State Space of Petri Nets With Stopwatches. ENTCS. 144(3): 59-77 (2006).