A STATE ENUMERATION APPROACH FOR ANALYZING TIME PETRI NETS

Bernard BERTHOMIEU, Miguel MENASCHE

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

LABORATOIRE D'AUTOMATIQUE ET D'ANALYSE DES SYSTEMES
7, Avenue du Colonel Roche 31400 TOULOUSE - France -

-

Keywords and phrases : Analysis of Time Petri Nets, Modeling and Analysis of concurrent systems with temporal constraints, Analysis of Communication Protocols.

Abstract :

This paper is concerned with the specification and proof of concurrent asynchronous systems in which time appears as a parameter : the model used here for representing these systems are Merlin's Time Petri Nets. An enumerative analysis technique is introduced, based on a derivation of classes of states, defined in the text. An algorithm is given for enumerating the classes in TPNS ; it allows to derive a finite representation of the behavior of a large family of TPNS. Among the illustrative examples provided is the verification of a communication protocol that makes use of a time out mechanism for recovering from losses of messages.

INTRODUCTION

Two sorts of times are generally dealt with in computing systems : Logical time and Physical time. Use of Logical time is made when expressing an ordering between events in the system, or a reachability relation among states of the system, it is the kind of time manipulated in Petri Nets or Transition Systems based models and also in Temporal Logic. Physical time differs from Logical time in that it may be quantified : it may be given a value.

Analysis of the behavior of concurrent asyncronous systems in which time appears as a quantifiable and continuous parameter is the object of the present paper. Among such systems are communication systems because communication protocols make a wide use of physical time in their specifications : using time-outs for specifying recovery mecanisms from losses of messages appears there in a quite natural way.

Several attempts to tackle the problem of analyzing systems in which temporal constraints are part of the specification have been made in the past and have been successful in particular fields, such as a performance evaluation. But, few references are available and so far, no general purpose technique has been devised. Such a technique is proposed here.

Most models devised in the past for representing these systems were derived from Petri Nets, probably because of their ability for representing parallel and asynchronous behaviors. Two formalisms, known as Time Petri Nets [1][2] and Timed Petri Nets[3] appeared in 1974.

Merlin defined its Time Petri Nets (TPNs) as Petri Nets in which a pair of non negative numbers (α, β), with $\alpha \leq \beta$, is associated with each transition. Numbers α and β associated with transition t are meant to be times, relative to the moment at which transition t was last enabled. Further assume that transition t has been last enabled at time τ, then t may not fire before time $\tau + \alpha$ and must fire before, or at, time $\tau + \beta$ unless it has been disabled before then by the firing of another transition. Furthermore, firing a transition takes no time to complete.

Merlin studied, using its TPNs, the recoverability problems in computing systems [1] and the design of communication protocols [2].

Ramchandani's Timed Petri Nets (TdPNs) are obtained from Petri Nets by associating a firing time with each transition of the net. The firing rule is further modified considering that firing of a transition is initiated

at the same moment it is enabled and that firing takes to complete the firing time associated with the transition.

Ramchandani's TdPNs, or slightly different but equivalent models, have been mainly used for performances evaluation. Ramchandani's results in [3] have been since extended in different directions, recent references in this area include [4][5][6][7].

Some properties of TdPNs are further investigated in [19].

Our first concern is to model systems in which temporal constraints appear and analyse their behavior : performances analysis will not be addressed in this paper. The model we choosed for representing these systems are Merlin's TPNs. Associating two times with each elementary action of the system permits to express conveniently most of the required temporal constraints on the actions, including duration ; while associating only a fixed duration makes difficult to express durations that may not be precisely quantified.

The main contribution of this paper is the concept of state class, defined in section 2. A method is given that permits to compute for an important family of TPNs a finite representation of their behavior, in terms of a set of state classes and a reachability relation in it. This enumerative approach is related to the "reachability graph" method for analyzing usual Petri Nets.

Section 1 recalls the definition of the Time Petri Nets and makes precise how their behavior may be caracterized.

States classes, and an algorithm for enumerating them, are introduced in section 2. Decidability questions for some properties of TPNs are investigated in section 3 and some examples of application of the introduced analysis method constitute section 4. In the last section are presented comments arising from the experiments carried out so far and suggestions for further work.

°0°

1. TIME PETRI NETS, DEFINITION AND BEHAVIOR

This section first recalls the definition of the time Petri Nets [1][2] and then introduces a notion of state for these nets that allows to caracterize their behavior.

Time Petri Nets

A Time Petri Net is a tuple (P,T,B,F,Mo,Is) where :

- P is a finite, non empty, set of Places ;
- T is a finite, non empty, set of Transitions,
- B is the Backward Incidence function

  $B : T \times P \rightarrow \mathbb{N}$

  where N is the set of non negative integers ;
- F is the Forward Incidence function

  $F : T \times P \rightarrow \mathbb{N}$
- Mo is the Initial Marking function

  $Mo = P \rightarrow \mathbb{N}$

  (P,T,B,F and Mo together define a Petri Net) ;
- Is is a mapping called Static Interval

  $Is : T \rightarrow Q \times (Q \cup \infty)$

  where Q is the set of rational numbers.

Further, Static Intervals satisfy the following restriction :

$Is(t) = (\alpha, \beta) \Rightarrow 0 \leq \alpha \leq \beta$.

Assume $Is(t) = (\alpha, \beta)$ for some transition t. Then :

- the closed interval of reals $[\alpha, \beta]$ will be called Static Firing Interval of transition t ;
- the left bound α will be called Static Earliest Firing Time (Static EFT for short) ;
- the right bound β will be called Static Latest Firing Time (Static LFT for short).

As mentioned in [1], Petri Nets may be considered as particular TPNs in which each transition has assigned the interval $[0, \infty]$.

It may be noticed that our definition of TPNs slightly differs from Merlin's in that we restrict numbers α and β associated with the transitions to be rational numbers, while they were defined as real numbers in [1][2]. The reason for this restriction, which in practice induces no limitations,

will appear in section 3 where properties of TPNs are investigated.

One may also notice that the terminology we use is not quite Merlin's one. This terminology has been introduced for making what follows easy to read.

Temporal interpretation

The behavior of TPNs will be made formal in the remaining of the section, but let us first introduce an example. The TPN figure 1 will be used throughout the text for illustrating the concepts and methods introduced. No particular meaning is assumed for it, it covers most of the cases one might be faced with in applying the firing rule for TPNs.

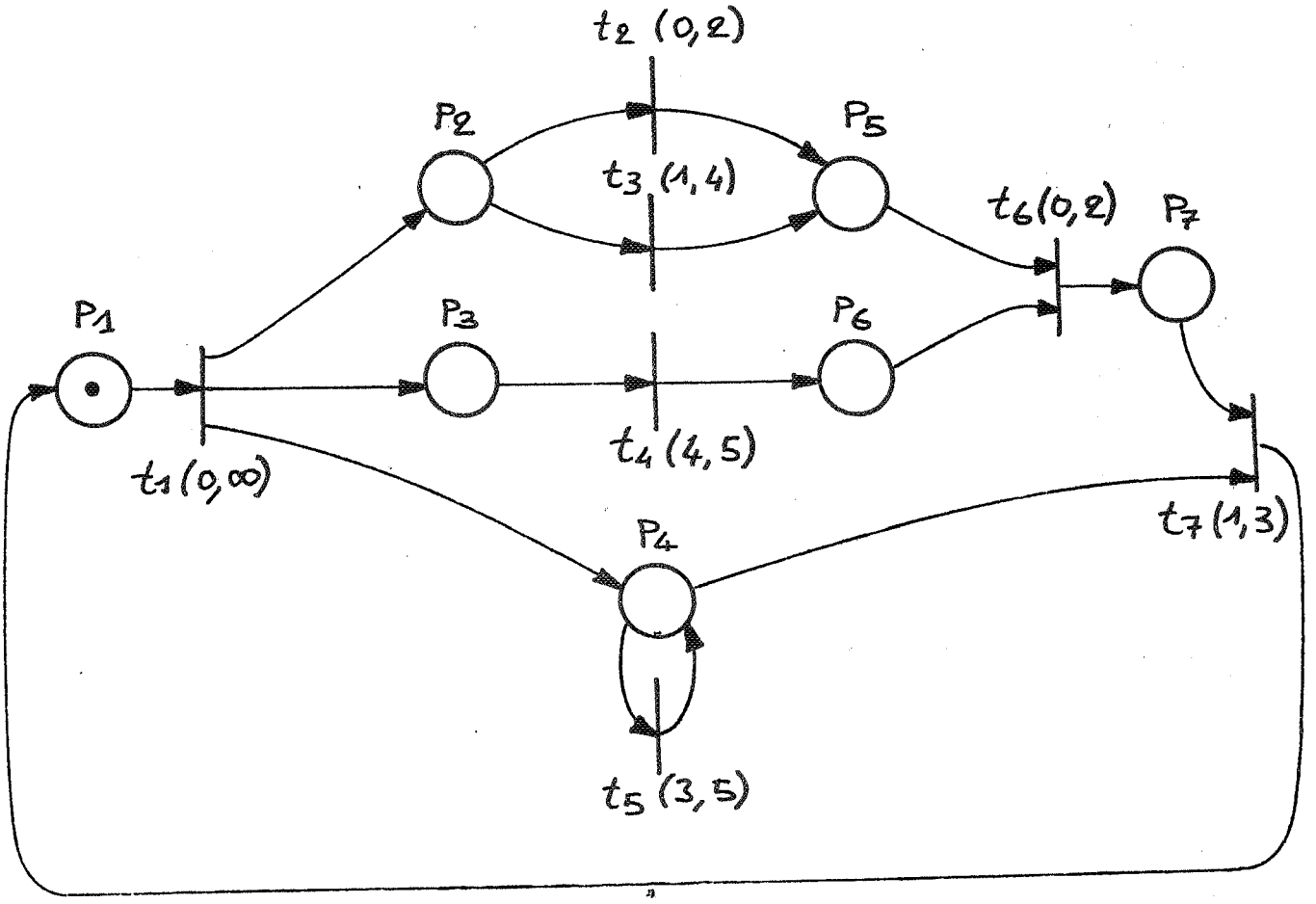


FIGURE 1 : A Time Petri Net

States in a TPN

We will make the restriction on TPNs we consider for analysis that none of their transitions may become enabled more than once "simultaneously". That is, for any marking M, the following holds for any transition t :

$$(\exists p)(M(P) < 2.\ B(t,p))$$

This non essential restriction is aimed at clarifying the statement of the firing rule ; it will be discussed at the end of the section.

With this restriction, we can assume the following general form for states of a TPN : a State S will be a pair S = (M,D) constituted of :

- a marking M ;
- a Firing Domain D which is a set of vectors of possible firing times. Vectors have one component for each transition enabled by marking M. Components i in vectors of this set are the times at which the i^{th} transition enabled by M is, individually, allowed to fire.

Firing domains may be conveniently expressed as the product set of the firing intervals of the enabled transitions, with intervals appearing in the product in the same order as enabled transitions appear in the ordered set of transition.

As an example, the initial state of the net represented figure 1 will be the pair So =(Mo,Do) in which :

- Mo = P1(1) ; (P1 marked with one token) ;
- Do = $[0,\infty]$.

Interval Do is related with transition t_1 which is the unique transition enabled by Mo.

For states other than the Initial State, firing intervals in the Firing Domain will be generally different from the Static Firing Intervals of the Transitions they are related with, and so will be their lower bounds (called simply EFT) and their upper bounds (called LFT).

Enabledness of transitions

Let us assume that the current state of the TPN is S = (M,D). Some transitions may be enabled by the marking M, but not all of them may be allowed to fire due to the imperative firing of transitions between their EFT and their LFT. This "Firability condition" is formally expressed as follows :

A transition t is firable from a state S = (M,D) at a time θ iff both the following conditions hold :

(i) t is enabled by marking M :

$(\forall p)\ (M(p) \geq B(t,p))$

(ii) θ is not smaller than the EFT of transition t and not greater than the smallest of the LFTs, among those of the transitions enabled by marking M :

$EFT_t \leq \theta \leq \text{Min}\{LFT_{ti}\}$

where ti ranges over the set of transitions enabled by M.

(i) is the current enabledness for Petri Nets, (ii) translates the fact that an enabled transition may not fire before its EFT and must fire before, or at, its LFT unless another fires before then and disables it.

It must be kept in mind that delay θ is relative to the Time at which state S has been reached, that is, the absolute time at which t may be fired is : θ plus the absolute time at which state S has been reached.

Firing rule between states

Let us assume that a transition t is firable at a time θ from a state S = (M,D). Then the state S' = (M',D') reached from S by firing t at time θ is computed as follows :

(1) the new marking M' is computed as in Petri Nets :

$(\forall p)\ (M'(p) = M(p) - B(t,p) + F(t,p))$

(2) the new firing domain D' is computed in three steps :

(a) Remove from the expression of domain D the intervals that relate to the Transitions disabled while t is firing. These transitions disabled are those enabled by M and not enabled by $M - B(t,-)$; they include transition t.

(b) Shift towards the origin of times the firing intervals that remain in the expression of the domain, of the value θ and troncate them, if necessary, to non negative values.

(c) Introduce in the expression of the domain with respect to the ordering on transitions the static intervals of the transitions newly enabled. The transitions newly enabled are those not enabled by $M - B(t,-)$ and enabled by M'.

Step (a) corresponds to projecting the domain on the dimensions corresponding to the transitions that remained enabled while t fired.

At step (b), time is incremented of the value θ. This is translated by shifting the remaining intervals of the value θ. At step (c), the domain of the new state is defined as the product of the current domains of the transitions that remained enabled and of the static firing intervals of the newly enabled transitions.

Example :

Let us illustrate the firability condition and the firing rule by some firings in the net figure 1.

The initial state is So = (Mo,Do) with

- Mo = P1(1) ;
- Do = $[0, \infty]$ (t_1 enabled).

Transition t_1 may fire at any time from So. Firing t_1 at a time θ_1 would lead to a state S1 = (M1,D1) with :

- M1 = P2(1), P3(1), P4(1) ;
- D1 = $[0,2] \times [1,4] \times [4,5] \times [3,5]$.

Enabled transitions are t_2, t_3, t_4 and t_5. Their firing intervals appear in this ordering in the expression of D1. Note that time θ_1 has not been used in the computation of Domain D1 since no transition remained enabled when firing t_1.

From state S1, only transitions t_2 and t_3 may fire since the EFT of t_4 and t_5 are later than the LFT of t_2. Transition t_2 may fire at any time in its firing interval $[0,2]$ while transition t_3 may only fire in the interval $[1,2]$, between its EFT and the LFT of t_2.

Firing t_3 at a time θ_3 in the interval $[1,2]$ from state S1 would lead to the state S3 =(M3,D3) with :

- M3 = P3(1), P4(1), P5(1) ;
- D3 = $[\max(0, 4-\theta_3), 5-\theta_3] \times [\max(0, 3-\theta_3), 5-\theta_3]$
  $= [4-\theta_3, 5-\theta_3] \times [3-\theta_3, 5-\theta_3]$ since $\theta_3 \leqslant 2$.

Intervals in the domain are related with transitions t_4 and t_5, these transitions remained enabled while t_3 was firing and thus their intervals have been shifted of a value equal to the time at which t_3 was fired. Further, transition t_2 was in conflict with t_3 and no transition is newly enabled.

Caracterizing the behavior of a TPN

The sentence "transition t is firable from state S at time θ and firing it leads to State S'" will be denoted :

$$S \xrightarrow{(t,\theta)} S'$$

A Firing Schedule will be a sequence of pairs :

$$(\sigma,\Theta) = (t_1,\theta_1) \cdot (t_2,\theta_2) \cdot \ldots\ldots\ldots\ldots \cdot (t_n,\theta_n)$$

in which $t_1, t_2, \ldots, t_n$ are transitions and $\theta_1, \theta_2, \ldots, \theta_n$ are times. This firing schedule is feasible from a state S iff there exist states S1,S2, ..., Sn such that :

$$S \xrightarrow{(t_1,\theta_1)} S1 \xrightarrow{(t_2,\theta_2)} S2 \longrightarrow \cdots \longrightarrow Sn-1 \xrightarrow{(t_n,\theta_n)} Sn$$

The firing rule permits to compute states and a reachability relation among them. The set of states that are reachable from the initial state or the set of firing schedules feasible from the initial state caracterizes the behavior of the TPN, in the same fashion as the set of reachable markings or the language of firing sequences caracterized the behavior of a Petri Net.

One could think about using this set of states for analysis purposes. Unfortunately, this is not possible in general since this set is usually infinite. The reason is that, as time is continuous, a firable transition may fire at any time in its allowed interval and thus leads to an infinity of successors for the state.

Section 2 is entirely devoted to finding a finite representation for this set of states. Before ending this section, let us discuss the restriction we introduced earlier on the TPNs to be analysed.

Assume that a transition t, with static firing interval $[\alpha,\beta]$, is enabled by the current marking and that a time θ (with $\theta \leq \beta$) elapsed since it was last enabled (this is possible when other transitions may fire independently of t). The current firing interval of t is thus $[\max(0,\alpha-\theta), \beta-\theta]$.

Now assume that the last firing have made transition t twice enabled by the current marking M, that is $(\forall p)\ (2.B(t,p) \leq M(p))$ and $(\exists p)\ (M(p) < 3.B(t,p))$.

Transition t has now two intervals associated with :

- $\max[(0, \alpha - \theta), \beta - \theta]$ for the first time it was enabled ;
- $[\alpha, \beta]$ for the last time it was enabled.

The question is : which one of these intervals should be considered when t fires or when t is disabled : anyone randomly ? the oldest ? another, obeying some rule ?

As can be seen, several interpretations are possible for multiple enabledness, and the firing rule would depend on the choosen interpretation. In the most general interpretation, transitions enabled several times simultaneously could be considered as independent occurrences of the same transition. If the choice is taken of firing the occurrence with the oldest interval, then transitions are fired with, in some sense, a first-in first-out discipline.

Many other strategies may be devised and consistant meanings may be certainly found for all of them. As a consequence, unless explicitely mentionned, the nets considered in the remaining of the paper obey to the previous restriction, i.e. no transition can be enabled more than once.

2. GROUPING STATES, STATE CLASSES AND ENUMERATION ALGORITHM

This section introduces state classes and an algorithm for enumerating them. This algorithm will allow to derive for a large family of TPNs a finite representation of their behavior. This behavior will be represented as a graph whose nodes are state classes and arcs represent a reachability relation among classes.

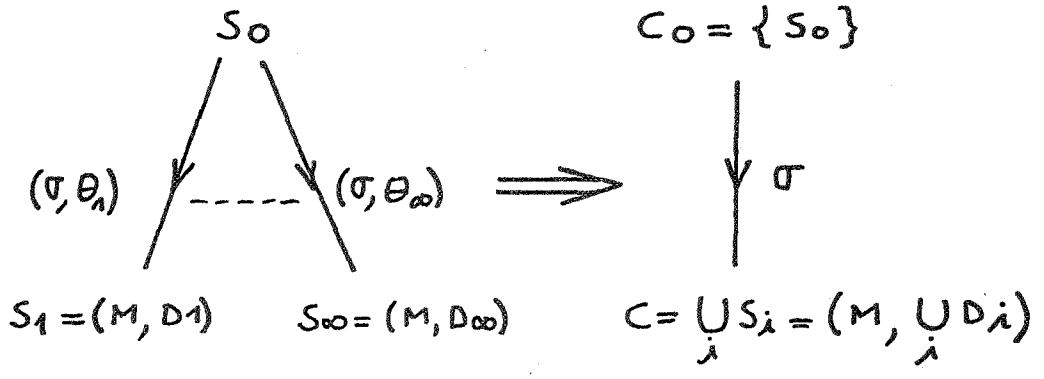
State classes

Rather than considering the state reached from the initial state by firing a firing schedule (σ, θ), we will consider the set of states reached from the initial state by firing all feasible schedules with the same firing sequence σ. This set of states will be called the State Class associated with the firing sequence σ.

These classes will be pairs C = (M,D) in which :

- M is the marking of the class, it is clear that all states in the class have the same marking ;
- D is the firing domain of the class, it is defined as the union of the firing domains of all the states in the class.

The step from states to State classes in given in the diagram below :



Computing classes

In practice, we are interested in computing recursively the set of classes, i.e. deriving the class associated with sequence $\omega.t$ from the class asociated with sequence ω, the initial class being defined as the class that contains only the initial state.

Let us assume that classes have the following general form C = (M,D) with :

- M : marking ;
- D : Firing Domain, defined as the solution set of some system of inequalities in which variables are 1 to 1 associated with the transitions enabled by marking M :

$$D = \{ \underline{t} \mid A.\ \underline{t} \geqslant \underline{b} \}$$

(A is a matrix, $\underline{b}$ a vector of constants and $\underline{t}$ the vector of variables).

It may be shown easily (in the next section) that the Firing Domain of the initial class may be expressed as above. Further, the statement of the firing rule will make clear that it maintains the general form above for domains of the classes.

Unlike the expressions of domains in the states, the expressions of domains for state classes allow to introduce relationships between the firing times of several transitions.

Enabledness of transitions from classes

Let us assume that the current state class of the net is C = (M,D), D being the solution set of some system of linear inequalities

$$A \cdot \underline{t} \geqslant \underline{b} \ .$$

A transition t is firable from the class C = (M,D) iff both the following conditions hold :

(i) t is enabled by marking M :

$$(\forall p) \ (M(p) \geqslant B(t,p)) \ ;$$

(ii) Assuming that transition t is the i^{th} transition enabled by marking M, then the following system of inequalities is consistent :

$$A \cdot \underline{t} \geqslant \underline{b}$$
$$\underline{t}\,(i) \leqslant \underline{t}\,(j) \text{ for all } j,\ j \neq i$$

where $\underline{t}\,(j)$ denotes the j^{th} component of vector $\underline{t}$.

Condition (i) is the usual enabledness, condition (ii) means that among the set of vectors of firing times that constitutes the domain of class C, at least one is such that transition t may be fired before, or at the same time than, any other enabled transitions.

It would be difficult to express condition (ii) for classes using the EFTs and LFTs of the transitions, as we did for states in section 1. This is because it exists possible relationships between the firing times of different transitions.

Firing rule between classes

Let us assume that transition t is firable from a class C = (M,D) with :

$$D = \left\{ \underline{t} \mid A \cdot \underline{t} \geqslant \underline{b} \right\}.$$

The class C' = (M',D') reached from class C = (M,D) by firing the transition t is computed as follows :

(1) As in Petri Nets, the new marking is defined by :

$$(\forall p)\ (M'(p) = M(p) - B(t,p) + F(t,p))\ ;$$

(2) Domain D' is computed from domain D in four steps :

(a) Add to the system $A \cdot \underline{t} \geqslant \underline{b}$, that defines domain D, the firability condition for transition t. This leads to the following system (assuming that t is the i^{th} transition enabled) :

$$\begin{cases} A \cdot \underline{t} \geqslant \underline{b} \\ \underline{t}(i) \leqslant \underline{t}(j) \text{ for all } j,\ j \neq i. \end{cases}$$

(b) First, make the following change of variables : express all variable $\underline{t}(j)$, with $j \neq i$, as the sum of component $\underline{t}(i)$ and a new variable $\underline{t}"(j)$:

$$\underline{t}(j) = \underline{t}(i) + \underline{t}"(j), \text{ for all } j,\ j \neq i.$$

Second, eliminate from the system the variable $\underline{t}(i)$.

The resulting system may be written :

$$\begin{cases} A" \cdot \underline{t}" \geqslant \underline{b}" \\ 0 \leqslant \underline{t}" \end{cases}$$

with A", b" computed from A,b, the equations that define the new variables and using Fourier's method (see [8] for instance) for eliminating variable $\underline{t}(i)$.

(c) Eliminate from the system obtained after step (b) all variables corresponding to the transitions disabled when t is fired : these transitions are those enabled by M and not enabled by M - B(t,—) i.e. before computing the new marking M - B(t,—) + F(t,—).

(d) Augment the system obtained after step (c) with new variables, one associated with each transition newly enabled, and constrain these variables to belong to the Static Firing Intervals of the transitions they are associated with, respectively. The newly enabled transitions are those not enabled by M - B(t,—) and enabled by M'.
This system may be written :

$$A' \cdot \underline{t}' \geqslant \underline{b}'.$$

It has as many variables as there are transitions enabled by marking M' and its solution set defines D'.

Some comments may be worth to explain these different steps :

(a) : Assuming that class C is the class associated with the firing of some sequence σ, then the solution set of the system found in (a) is the union of the firing domains of the states belonging to class C, from which transition t is firable and restricted by $\underline{t}$ (t) $\leq$ $\underline{t}$(others).

(b) : Considering component $\underline{t}$ (i) as a parameter, the system expressed with the new variables (after the first sub-step) gives, for a given value x of the parameter, the possible vectors of firing times for all enabled transitions distinct from t with, as new origin for times, the parameter x, which is the moment at which transition t is fired.

Further, eliminating variable $\underline{t}$ (i) from the system gives a system whose solution set is the set of all possible vectors of firing times for these transitions, relative to the moment at which transition t is fired, this for all possible values of the firing time of t.

(c) : The solution set of system found at step (C) is the projection, on the remaining dimensions, of the solution set of the solution set of the system found at step (b).

Relationships between variables that remain in the system are not changed by the elimination operation.

(d) : The resulting solution set is the product of the set found at step (c) and the Static Firing Intervals of the newly enabled transitions.

These comments should suffice to convince the reader that, if C was the class associated with the firing sequence σ, then C' = (M',D') as computed above is actually the class associated with sequence σ.t. For a more rigourous demonstration, we should detail the proofs of the assertions expressed in the comments.

How classes express the behavior of the TPN

Using the firing rule, a tree of classes can be built. Its root is the initial class and there is an arc labelled with t from class C to class C' if t is firable from class C and if firing it leads to class C'.

It may be noticed that in this tree of classes, each class has only a finite number of successors, at most one for each transition enabled by the marking of the class.

It is clear from the definition of the classes that any sequence of transitions firable in the TPN will be a path in the above tree of classes. Further, the existence of a path labelled w between two classes C, associated with sequence T, and C' of the tree must be interpreted as follows :

There exist feasible firing schedules (σ, θ_1) and (ω, θ_2) such that :

$$So \xrightarrow{(\sigma, \theta_1)} S \xrightarrow{(\omega, \theta_2)} S'$$

with state S belonging to class C and state S' belonging to class C'.

Checking state classes for equality

Two state classes will be defined equal iff both their markings are equal and their firing domains are equal. Checking firing domains for equality require some comments since domains are defined as solution sets of systems of linear inequalities.

It may be shown that, in the general case, comparing for equality the solution sets of two systems of linear inequalities, with same variables and n and m inequalities respectively may be done by solving m systems with (n + 1) inequalities and n systems with (m + 1) inequalities with same variables as the systems to compare. This method is a straight forward application of the set equality, the solution set of a system of inequalities being the intersection of the solution sets of its constituting inequalities.

Though possible, this method would be rather unefficicent since every domain computed has to be compared pair-wise with each previously computed domain. A better method should be first to put the systems that define the domains into some canonical form, as soon as they are computed, and then comparing for identity the canonical forms of the systems. This canonical form for systems should have obviously the property that canonical forms of two systems are identical if and only if the solution sets of the systems are equal.

We will not address this problem in the general case. For our particular case, it may be shown (this will be done in section 3) that the systems defining the firing domains are systems with at most two variables per inequation and have the following general form :

$$GF \begin{cases} \alpha_i \leq \underline{t}(i) \leq \beta_i & \text{for all } i ; \\ \underline{t}(j) - \underline{t}(k) \leq \delta_{jk} & \text{for all } j, k \text{ with } j \neq k \end{cases}$$

where $\underline{t}$ (i), $\underline{t}$ (j) and $\underline{t}$ (k) are variables and α_i, β_i, γ_{jk} are constants.

So, let us assume that we start from a system with the above form. A canonical form for this system would be the following :

$$(CF)\quad \begin{cases} \alpha_i^* \leqslant \underline{t}\ (i) \leqslant \beta_i^* & \text{for all } i\ ; \\ \underline{t}\ (j) - \underline{t}\ (k) \leqslant \gamma_{jk}^* & \text{for all } (j,k) \text{ with } j \neq k. \end{cases}$$

with : α_i^* : smalllest possible value of variable t_i,

β_i^* : largest possible value of variable t_j,

γ_{jk}^* : largest possible value of the difference $t_j - t_k$.

The solution set of a given system of form (GF) may be associated with only one system of form (CF) since transforming into an equality any inequality in system (CF) gives an equality the solution set of which contains a part of the boundary of the domain.

Details of the method will not be given here, but computing this canonical form from a system with form (GF) may be done by a series of eliminations and simple comparisons. About the complexity of computing the canonical form, it is conjectured at the moment that this can be done in polynomial time. Strong convictions for this claim come from a result in [12] in which it is shown that solving systems of inequalities with at most two variables per inequalities can be done in polynomial time.

A state class may thus be caracterized by its marking M, constants α_i^* and β_i^* for each enabled transition and constant γ_{jk}^* for each distinct pair of enabled transitions. Computation of the canonical form may be put as an additionel step to the firing rule ; comparing classes for equality will be then comparing per equality strings of numbers.

When the tree of classes of a TPN will have a bounded number of distinct nodes, we will associate a graph to the net, obtained by grouping equal classes of the tree into a single class. This graph will be called the reachability graph of the TPN.

Example :

Before giving as an example the classes and the graph of classes for the TPN represented figure 1, let us introduce a (non essential) technical trick we use for computing classes : let us suppose that some transition may

be fired from a class C, but that the earliest time at which a transition may fire is a strictly positive time α. Then firing a transition t from C at a time θ (θ will be necessarily greater or equal than α) may be decomposed into :

- "advance" time doing nothing, until time α is reached. Let us denote C' the class reached at that step ;
- fire transition t at time $\theta - \alpha$ from the class C'.

When computing classes, the class C' above will be choosen as representative for class C. Computation of time α and of the firing domain of class C' are easily carried out from the canonical form of class C.

The following table 1 gives the list of representative classes for the net represented figure 1, table 2 gives the reachability graph for this net.

CLASS C0 :

M = P1(1)
I = $0 \leq T1 \leq \infty$

CLASS C1 :

M = P2(1), P3(1), P4(1)
I = $0 \leq T2 \leq 2$
$1 \leq T3 \leq 4$
$4 \leq T4 \leq 5$
$3 \leq T5 \leq 5$

CLASS C2 :

M= P3(1),P4(1),P5(1)
I= $1 \leq T4 \leq 4$
$0 \leq T5 \leq 4$
$T4 - T5 \leq 2$
$T5 - T4 \leq 1$

CLASS C3 :

M = P4(1), P5(1), P6(1)
I = $0 \leq T5 \leq 1$
$0 \leq T6 \leq 2$

CLASS C4 :

M = P4(1), P5(1), P6(1)
I = $3 \leq T5 \leq 5$
$0 \leq T6 \leq 2$

CLASS C5 :

M= P4(1), P7(1)
I= $0 \leq T5 \leq 4$
$0 \leq T7 \leq 2$

CLASS C6 :

M = P4(1), P7(1)
I = $2 \leq T5 \leq 4$
$0 \leq T7 \leq 2$

CLASS C7 :

M = P4(1), P7(1)
I = $0 \leq T5 \leq 1$
$1 \leq T7 \leq 3$

CLASS C8 :

M= P3(1),P4(1),P5(1)
I= $0 \leq T4 \leq 2$
$3 \leq T5 \leq 5$

CLASS C9 :

M = P4(1), P5(1), P6(1)
I = $1 \leq T5 \leq 5$
$0 \leq T6 \leq 2$

CLASS C10 :

M = P4(1), P5(1), P6(1)
I = $3 \leq T5 \leq 5$
$0 \leq T6 \leq 1$

CLASS C11 :

M= P4(1), P7(1)
I= $1 \leq T5 \leq 4$
$0 \leq T7 \leq 2$

CLASS C12 :

M = P4(1), P7(1)
I = $0 \leq T5 \leq 5$
$1 \leq T7 \leq 3$

CLASS C13 :

M = P3(1), P4(1), P5(1)
I = $1 \leq T4 \leq 3$
$0 \leq T5 \leq 3$
$T4 - T5 \leq 2$
$T5 - T4 \leq 1$

where Ti is the variable associated with transition ti.

TABLE 1 : List of State Classes for the net figure 1

C0 ⟶ t1/C1
C1 ⟶ t2/(S=1)/C2, t3/(S=1)/C13
C2 ⟶ t4/C3, t5/C8
C3 ⟶ t5/C4, t6/C7
C4 ⟶ t6/(S = 1)/C5
C5 ⟶ t5/(S = 1)/C6, t7/C0

C6 ⟶ t5/(S = 1)/C6, t7/C0
C7 ⟶ t5/(S = 1)/C6, t7/C0
C8 ⟶ t4/C9
C9 ⟶ t5/C10, t6/C12
C10 ⟶ t6/(S = 1)/C11
C11 ⟶ t5/(S = 1)/C6, t7/C0
C12 ⟶ t5/(S = 1)/C6, t7/C0
C13 ⟶ t4/C3, t5/C8

C0 is the initial class. Line 6, for instance, must be read : firing t5 from class C5 leads to a class that, once shifted of a time 1, is equal to class C6; firing t7 from class C5 leads to class C0.

TABLE 2 : Graph of classes for the net figure 1

The graph represented table 2, together with the content of the classes given table 1 caracterize the behavior of the net represented figure 1. Feasible firing schedules are not made explicit in these tables but may be deduced with little effort. We did not give, so far, conditions under which a given TPN has a finite number of classes. This problem is addressed in the following section.

3. SOME PROPERTIES OF TIME PETRI NETS

Let us introduce some notations and terminology we will use in this section :

- The set of markings of a TPN that can be reached from its initial marking will be denoted R(Mo) ;
- The reachability problem is whether or no a given marking belongs to R(Mo) ;

- The boundedness problem is whether or no all markings in R(Mo) are bounded, i.e. :

$$(\exists k \in \mathbb{N})\ (\forall M \in R(Mo))\ (\forall p \in P)\ (M(p) \leqslant k)\ ;$$

- A TPN will be said T-bounded if it exists a natural number k such that none of its transitions may be enabled more than k times simultaneously by any reachable marking, i.e. :

$$(\exists k \in \mathbb{N})\ (\forall M \in R(Mo))\ (\forall t \in T)\ (\exists p \in P)\ (M(p) < (k + 1)\ .\ E(t,p))$$

It may be noted that Boundedness implies T-boundedness but that the converse is not true .

- The property T-safe is the particular case of the above T-bounded property, with k = 1.

Theorem 1 recalls an undecidability result for TPNs :

THEOREM 1 :

The reachability and boundedness problems for TPNs are undecidable.

Proof : A direct proof is produced in reference [9]. Others (indirect) proofs may be produced : it can be shown that TPNs can simulate Inhibitor Nets and Priority Petri Nets, and have equivalent reachability and boundedness problems. Since these problems are known undecidable for the two latter classes of nets, (see [10] for instance), it may be infered that they are also undecidable for TPNs. QED

A straightforward consequence of theorem 1 is that the finiteness of the set of classes for TPNs is undecidable since classes are pairs (marking, firing domain).

Several sufficient conditions for boundedness will be given later in the section, but we need first to state some theorems.

Lemma 1 : General form Lemma

The firing domains of state classes for any T-Safe TPN may be expressed as solution sets of systems of inequalities of the following form :

$$\begin{cases} \alpha_i \leqslant \underline{t}\,(i) \leqslant \beta_i & \text{for all } i\ ; \\ \underline{t}(j) - \underline{t}(k) \leqslant \gamma_{jk} & \text{for all } j, k \text{ with } j \neq k \end{cases}$$

where $\underline{t}\,(i)$ is the variable associated with the i^{th} transition enabled by the marking of the class.

Proof : The initial firing domain fits into this general form. It may be expressed as a set of inequations : $\alpha_i^s \leq \underline{t}\,(i) \leq \beta_i^s$, where $\underline{t}\,(i)$ is associated with the i^{th} transition initially enabled and α_i^s and β_i^s are the Static EFT and LFT of that transition, respectively (β_i^s may be unbounded).

Default values for γ_{jk} may be provided with $\gamma_{jk} = \beta_j^s - \alpha_k^s$, these redundant inequations will not affect the solution set of the system.

The computation of the domains in the firing rule for classes defined in section 2 consisted of four steps. For the purpose of the proof, let us express it as three steps and show that those steps produce, from a system with the general form, a new system that have also this general form.

STEP 1 : Consists of steps (a) and (b) of the firing rule. Starting from a system with form above, step 1 transforms as follows this system; $\underline{t}\,(f)$ being the variable associated with the transition fired :

(OP1)
- $-\alpha_i$ becomes max $(0, -\gamma_{fi}, \alpha_i - \beta_f)$
- $-\beta_i$ becomes min $(\gamma_{if}, \beta_i - \alpha_f)$
- $-\gamma_{jk}$ becomes min $(\gamma_{jk}, \beta_j - \alpha_k)$
- All inequations containing variable $\underline{t}\,(f)$ disappear from the system.

STEP 2 : Eliminations

Corresponds to step (c) of the firing rule. It consists of successive eliminations in the system of the variables associated with the transitions disabled while $\underline{t}\,(f)$ was firing.

Each elimination, for instance of variable $\underline{t}\,(e)$, corresponds to the following transformation of the system :

(OP2)
- $-\alpha_i$ becomes max $(\alpha_i, \alpha_e - \gamma_{ei})$
- $-\beta_i$ becomes min $(\beta_i, \beta_e + \gamma_{ie})$
- $-\gamma_{jk}$ becomes min $(\gamma_{jk}, \gamma_{je} + \gamma_{ek})$
- All inequations containing variable $\underline{t}\,(e)$ disappear from the system.

STEP 3 : Introductions

Corresponds to step (d) of the firing rule. It consists of successive introductions in the system of the variables and inequations relative to the newly enabled transitions.

Each introduction, of variable $\underline{t}$ (n) for instance, corresponds to the following transformation of the system :

(OP3)
- For all i, j, k, distinct from n, α_i, β_i and γ_{jk} do not change.
- A new variable $\underline{t}$ (n) is introduced, constrained by the inequations:
  $$\alpha_n^s \leq \underline{t}\ (n) \leq \beta_n^s$$
  in which α_n^s and β_n^s denote the Static EFT and LFT of the transition associated with tn respectively. Thus α_n is α_n^s and β_n is β_n^s.
- Further inequations may be provided for relationships between variable $\underline{t}$ (n) and the others. These inequations are :
  $$\underline{t}\ (n) - \underline{t}\ (k) \leq \gamma_{nk} \qquad \text{for all } k\ ;\ k \neq n$$
  $$\underline{t}\ (j) - \underline{t}\ (n) \leq \gamma_{jn} \qquad \text{for all } j\ ;\ j \neq n$$

  Default values for γ_{nk} and γ_{jn} must be choosen such that these inequations are redundant. This is acheived by :
  $$\gamma_{nk} = \beta_n^s - \alpha_k$$
  $$\gamma_{jn} = \beta_j - \alpha_n^s$$

For acheiving the proof of lemma 1, it must be shown that firing a transition keeps the general form for the systems that define the firing domains. Firing a transition consists of an application of transformation (OP1) followed by a number of applications of (OP2) and a number of applications of (OP3). Each of these transformations, individually applied, keeps the general form, thus so does their combination. QED.

Lemma 2 :

The constants α_i, β_i and γ_{jk} of any system computed from the initial system by the firing rule are linear combinations with integer coefficients of the Static EFTs and LFTs associated with the transitions of the TPN. i.e. the following hold :

$$(\forall i)\ (\exists\ d_1, \ldots, d_{2n} \in \mathbb{Z})$$
$$(\alpha_i = d_1 \alpha_1^s + \cdots + d_n \alpha_n^s + d_{n+1} \beta_1^s + \cdots + d_{2n} \beta_n^s)$$

(and similarly for each β_i and γ_{jk}).

Proof : The proof is straight forward : Lemma 2 is true for the initial system and this property is kept when the three basic transformations OP1, OP2 and OP3 are individually applied. QED.

Lemma 3 :

The constants α_i, β_i and γ_{jk}, for all i, j, k, of any system computed from the initial system by the firing rule have the following bounds :

$$0 \leq \alpha_i \leq \alpha_i^s$$
$$0 \leq \beta_i \leq \beta_i^s$$
$$-\alpha_k^s \leq \gamma_{jk} \leq \beta_j^s .$$

Sketck of the proof :

Upper bound for β_i lower bound for α_i and upper bound for γ_{jk} are easily proven by induction on the basic transformations : they are satisfied by the initial system and they remain satisfied when any of these transformation is applied. Consistency of the systems implies the lower bound for β_i. Upper bound for α_i requires that the smallest possible value of t_i (denoted α_i^*) may only decrease when firing a transition. Using this fact and invoking consistency of the systems allow to prove the upper bound for α_i and lower bound for γ_{jk}. QED.

It may be noticed that, if the Static LFT β_i^s is not bounded for some transition i, then components β_i and γ_{ij} (for all j, $j \neq i$) are initially, and will remain, unbounded.

Lemma 4 :

Let A and B be two bounded real constants and $q_1, \ldots, q_n$ be a finite set of rational constants.

Then there are only a bounded number of linear combinations of numbers $q_1, \ldots, q_n$, with integer coefficients, between numbers A and B.

I.e. the number of rational numbers X such that

$$X = \lambda_1 q_1 + \cdots + \lambda_n q_n$$

and $\lambda_1, \ldots, \lambda_n \in \mathbb{Z}$

and $q_1, \ldots, q_n \in \mathbb{Q}$

and $A \leq X \leq B$

is bounded.

Proof : Let us denote d the common denominator of rational numbers $q_1, \ldots, q_n$, and Q_i the product $d.q_i$. The problem above is equivalent to proving that there are only a bounded number of linear combinations of integers $Q_1, \ldots Q_n$, with integer coefficients, between the bounds d.A and d.B, which is obviously true. QED.

Theorem 1 addressed the problem of the finiteness of the set of markings. Theorem 2 below is concerned with the finiteness of the set of firing domains of the state classes.

THEOREM 2 :

If a TPN is T-bounded, then the set of all the firing domains of its state classes is finite.

Proof : The proof is carried out for the T-Safe case and then extended, with an adequate interpretation, to the T- bounded case.

The possible α_i, β_i and γ_{jk} for systems that describe the domains are linear combinations with integer coefficients of the static EFTs and LFTs (from lemmas 1 and 2), they are either unbounded (and in this case remain unbounded) or have upper and lower bounds (lemma 3). Further, the Static EFTs and LFTs are rational numbers (definition of TPNs). Thus, using lemma 4 only a bounded number of distinct α_i, β_i and γ_{jk} may be computed with the firing rule.

Consequently, only a bounded number of systems may be computed, corresponding to a bounded number of firing domains for classes.

This proves theorem 2 for the T-Safe case.

For extending the proof to the T-bounded case, let us consider the (possibly many) variables associated with a given transition as independent. Firing the transition is then firing one of its occurrences (all possible alternatives must be taken) and disabling applies to one or several occurrences of the same transition, (all combinations for remaining enabled occurrences must be considered).

The firing rule needs not to be modified further. The only difference with the T-safe case is then that there may be more variables in the system that there are enabled transitions. But, since the net is T-bounded, the total number of variables is bounded and, using a similar proof that for the T-safe case, it comes that only a bounded number of firing domains may be computed for the TPN. QED.

Remark :

Restricting the static EFTs and LFTs of transitions to be rational numbers (instead of real numbers in Merlin's definition) is essential. Theorem 2 does not hold if EFTs and LFTs are real numbers.

Figure 2 below shows one of those nets which is bounded (it has only one marking), but which has an unbounded number of classes of states.

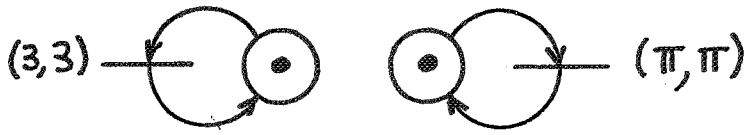


FIGURE 2

Theorem 2 is interesting in that it permits to infer theorem 3 below :

THEOREM 3 :

A TPN has a bounded number of state classes if and only if it is bounded.

Proof : If not bounded, then it has an unbounded number of classes since classes are pairs (marking, firing domain).

If bounded, then it is T-bounded and it has both a bounded number of markings (consequence of the boundedness property) and a bounded number of firing domains(consequence of theorem 2). QED.

So, any sufficient condition for the boundedness property will provide also a sufficient condition for the finiteness of the set of classes. Three of these sufficient conditions, of increasing strength, are discussed in the sequel. They are stated for T-Safe TPNs only.

The notation $M' \gtrdot M$ will be used in the following theorems ; it is defined as :

$$(\forall p \in P)\ (M'(p) \geqslant M(p)) \wedge (\exists p \in P)\ (M'(p) \neq M(p)).$$

Lemma 5 :

If a T-Safe TPN is not bounded, then a firing sequence of unbounded length, going through a sequence S of states classes in which all classes are pairwise different, is firable from its initial class.

Proof : From theorem 3 it comes that an unbounded TPN has an unbounded number of reachable classes. Further, since each state class may only have a bounded number of successors by the firing rule, its set of classes must necessarily contain such a sequence S. QED.

THEOREM 4 : Sufficient Condition 1 (SC1) :

(SC1) A T-Safe TPN is bounded if no pair of state classes $C = (M,D)$ and $C' = (M',D')$ are reachable from its initial state class and are such that.

(i) C' is reachable from C ;

(ii) $M' \gtrdot M$.

Proof : Suppose the TPN unbounded and consider the unbounded sequence S used in lemma 5. Since the net is T-Safe, it admits only a bounded number of firing domains (theorem 2) and, as classes are pairs (marking, domains), the unbounded sequence S must contain an unbounded subsequence S' in which all markings of the classes are pairwise different. Further, using [11], such an unbounded subsequence S' would necessarily contain two classes $S'_i = (M,D)$ and $S'_{i+n} = (M',D')$ with $M' \gtrdot M$.

So, SC1 is sufficient for boundedness. However it is not necessary : sufficient condition 1 fails for the TPN represented figure 3 below, though this net admits only two state classes.

The reason is that for TPNs, if a firing sequence is firable from a marking M, it does not implies that the same sequence is firable from a marking M' with $M' \gtrdot M$.

It may be noticed that SC1 holds for any TPN such that the Petri Net we obtain from it by removing the time constraints on all transitions is bounded (such as the TPN figure 1). This does not implies that SC1 is worthless since it also holds for some TPNs that do not satisfy this property.

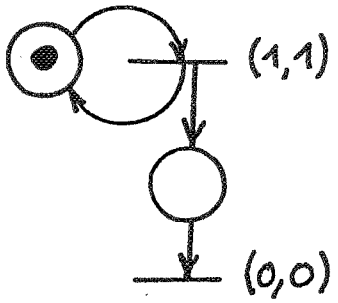


FIGURE 3

THEOREM 5 : Sufficient Condition 2

A T-Safe TPN is bounded if no pair of state classes C = (M,D) and C' = (M',D') are reachable from its initial class such that :

(SC2)
(i) C' is reachable from C ;
(ii) $M' \geqslant M$;
(iii) $D' = D$.

Proof : Suppose the TPN unbounded and consider the unbounded sequence S' used in the proof of theorem 4. Since this sequence is unbounded and the net admits only a bounded number of firing domains, sequence S' must contain an unbounded subsequence S" in which all classes have the same domain. Further, this unbounded sequence S" must contain ([11]) two classes $S"_i = (M, D)$ and $S"_{i+n} = M',D'$ with $M' \geqslant M$ and $D' = D$. So, SC2 is a sufficient condition for boundedness. However the condition is not necessary : SC2 fails for the net figure 4. Below, though this net admits only eleven state classes. QED.

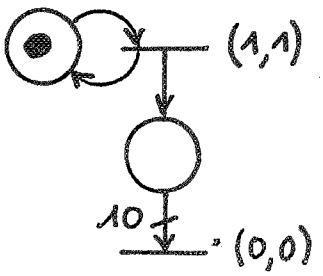


FIGURE 4

It may be noticed that SC2 permits to prove bounded the net represented figure 3.

THEOREM 6 : Sufficient Condition 3

A TPN is bounded if no pair of state classes C = (M,D) and C' = (M',D') are reachable from its initial state class and are such that :

(i) C' is reachable from C ;

(SC3) (ii) $M' \not\geq M$;

(iii) $D' = D$;

(iv) $(\forall p \in Pw(M,M'))\ (M(p) > \max_{t \in T} B(t,p))$

where $Pw(M,M') = \{p \in P \mid M'(P) > M(P)\}$.

Proof : Suppose the TPN unbounded and consider the sequence S" used in the proof of theorem 5. Using [11], and since the number of distinct subsets of P is bounded, this sequence S" must necessarily contain an unbounded subsequence $S"' = (M_i, D_i)\ i \in \mathbb{N}$ such that :

$$(\forall i)\ (M_i \not\leq M_{i+1})$$

and $$(\forall i)\ (Pw(M_i, M_{i+1}) \subseteq Pw(M_{i-1}, Mi))$$

Further, since the length of this sequence is unbounded and that the number of distinct possible Pw is bounded, there must exist in this sequence some pair of classes $S"'_i = (M,D)$ and $S"'_{i+n} = (M',D)$ such that

$$(\forall p \in Pw\ (M, M'))\ (M(p) > k(p))$$

where k is any mapping associating with any place an integer k(p). Obviously, k may be choosen such that for any place P, $k(P) = \max (F(t,p))$. So, SC3 is a sufficient condition for boundedness. However the condition is not necessary : SC3 fails for the net figure 5 below though this net admits only 13 classes. QED.

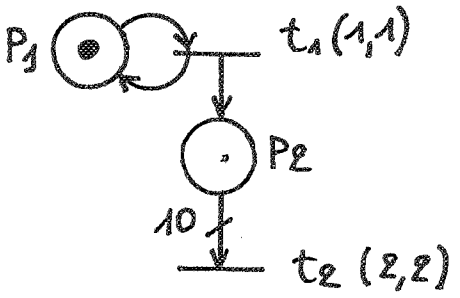


FIGURE 5

In the net figure 5, firing ten times transition t_1 leads to a state class with marking P1(1), P2(10) and firing intervals $[1,1]$, $[2,2]$, for transitions t_1 and t_2 respectively. From this class, transition t_1 may be fired two times, leading to two classes C and C' successively. Classes C and C' satisfy the three conditions expressed in theorem 6 and thus SC3 fails to prove this net bounded. But, the net is bounded : from class C', firing once more transition t_1 is not allowed, and firing t_2 leads to a previously enumerated class.

Using theorem 6, the enumeration of classes stops when a firing sequence is found that does not decrease the marking of any place and such that all places it increases the marking of had, when starting the sequence, markings greater than some bounds k(p). Though bounds k(p) may be arbitrarily choosen, it would be wise to choose them as the maximum weight among those of the outgoing arcs from the place. The idea behind this choice is that if the marking of place P reaches that value, then it will help enabling some transition that, when firing, will decrease the marking of place P. This is obviously not always the case, due to the timing constraints (as for the example figure 4), or to the fact that some transitions may be dead.

It may be noticed that SC3 permits to prove bounded the net represented figure 4. Also, it is clear from the statements of theorems 4, 5 and 6 that :

$$SC3 \Rightarrow SC2 \Rightarrow SC1.$$

In practice, the experimental computer software package we are developping for analyzing TPNs allows a user controlled enumeration when checking for boundedness. Sufficient Condition 3 is used together with conditions expressed by the user who provided the net and based upon his intuitive understanding of the behavior of the net. Typical user defined conditions are upper bounds for the markings of some places or relationships between markings of several places. This technique has been proved adequate for most of the application examples we have treated so far.

Finally, when the TPN associated with is bounded, it becomes possible, using the graph of classes, to prove specific properties that caracterize the correct behavior of the system represented. Further, liveness properties, similar to those defined for Petri Nets, may be defined for TPNs and, for bounded TPNs, proved using the graph of state classes.

4. EXAMPLES

4.1. *Analysis of Communication Protocols* :

As mentionned earlier, communication protocols make a wide use of timing constraints : recovery mechanism for losses of messages are usually implemented using time outs.

TPNs constitute a suitable tool to verify that the choosen values for the time outs are correct.

The example used for illustrating the verification of communication protocols is the Alternating Bit Protocol [13]. This protocol is concerned with transmitting messages from one place to another, allowing only one message in transit at a time : the sender process waits, before sending a new message, for the acknowledgement of the last message it sent. Hypothesis on the behavior of the transmission medium is that messages, or their acknowledgement, may be lost or damaged during transit.

A mechanism is provided for recovering from these losses : a time out is set when a message is sent and, if its acknowledgement does not arrive in time, then the message is retransmitted.

But, the above mechanism is not sufficient to prevent duplicated messages ; this is because if the acknowledgement for the last message sent is lost, then the receiver is unable to decide whether the next message is a new message or a copy of the last message it received. To overcome this problem, the protocol uses modulo-2 sequence numbers attached to the messages.

The TPN figure 6 is a representation of such a protocol. For simplifying the example, it is assumed that only losses of messages and of acknowledgements are possible. The meanings of the transitions are given in Table 3. Estimates for the duration of all elementary actions are provided, except for the sending of the first copy of the messages. Retransmissions of messages occur at a time comprised between 5 and 6 units of time after the message is sent. Equal estimates of times (between 0 and 1) are given for losses and receptions of messages and acknowledgements.

The graph of state classes produced for this net is represented figure 7. Sixteen classes have been computed ; markings and firing domains for these classes are given in Table 4.

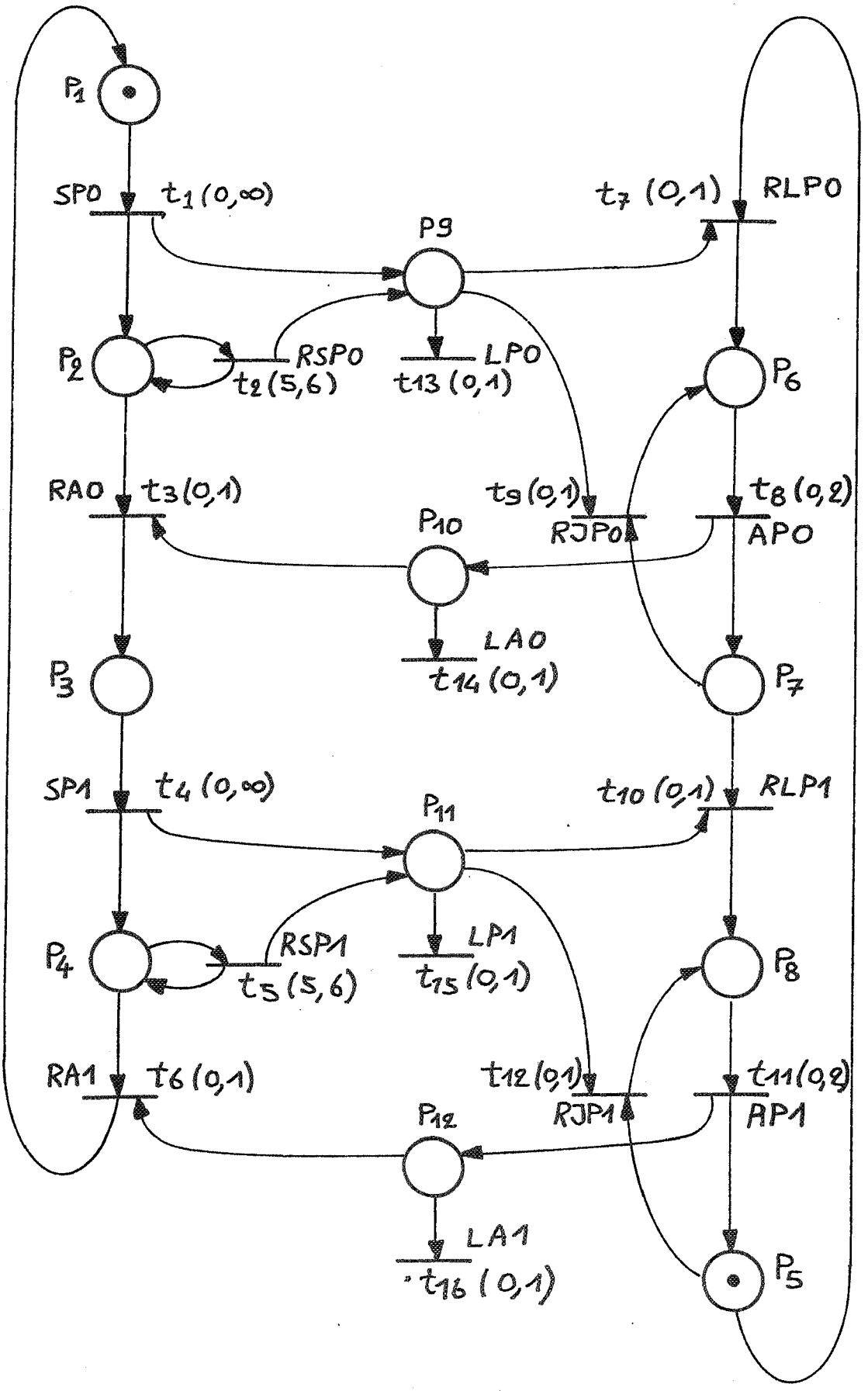


FIGURE 6 : A TPN for the Alternating Bit Protocol Example

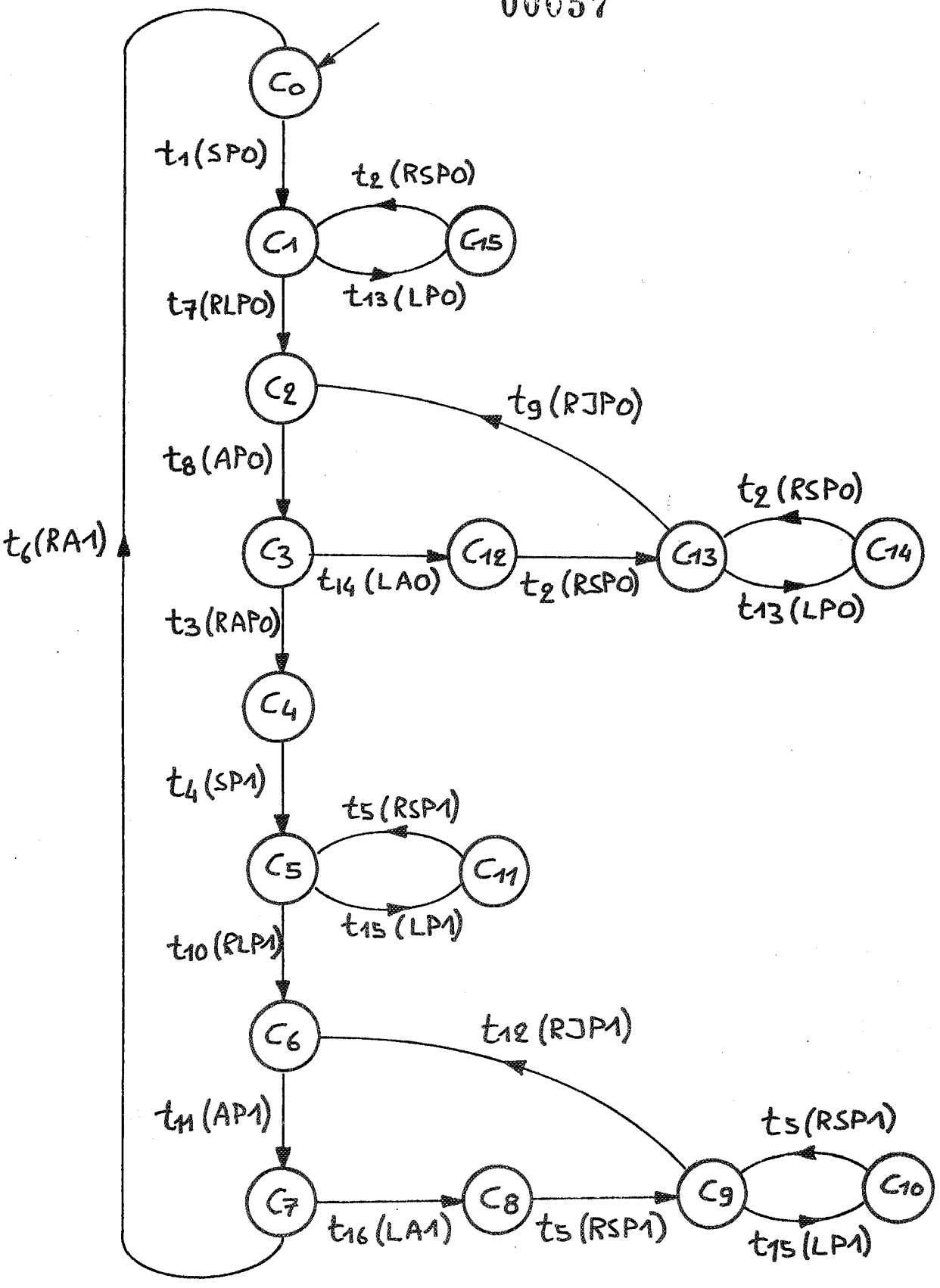


FIGURE 7 : Graph of state classes for the Alternating Bit Protocol

T1 : Send packet O
T2 : Resend packet O
T3 : Receive ack O
T4 : Send packet 1
T5 : Resend packet 1
T6 : Receive ack 1
T7 : Receive/Release packet O
T8 : Send ack O
T9 : Receive/Reject packet O
T10: Receive/ Release packet 1
T11: Send ack 1
T12: Receive/Reject packet 1
T13: Lose packet O
T14: Lose ack O
T15: Lose packet 1
T16: Lose ack 1

TABLE 3 : Meanings of the transitions for the Alternating Bit Protocol Example

CLASS O
M = P1(1), P5(1)
I : O ⩽ T1

CLASS 1
M = P2(1), P5(1), P9(1)
I = 5 ⩽ T2 ⩽ 6
O ⩽ T7 ⩽ 1
O ⩽ T13 ⩽ 1

CLASS 2
M = P2(1), P6(1)
I = 4 ⩽ T2 ⩽ 6
O ⩽ T8 ⩽ 2

CLASS 3
M = P2(1), P7(1), P10(1)
I = 2 ⩽ T2 ⩽ 6
O ⩽ T3 ⩽ 1
O ⩽ T14 ⩽ 1

CLASS 4
M = P3(1), P7(1)
I = O ⩽ T4

CLASS 5
M = P4(1), P7(1), P11(1)
I = 5 ⩽ T5 ⩽ 6
O ⩽ T10 ⩽ 1
O ⩽ T15 ⩽ 1

CLASS 6
M = P4(1), P8(1)
I = 4 ⩽ T5 ⩽ 6
O ⩽ T11 ⩽ 2

CLASS 7
M = P4(1), P5(1), P12(1)
I = 2 ⩽ T5 ⩽ 6
O ⩽ T6 ⩽ 1
O ⩽ T16 ⩽ 1

CLASS 8
M = P4(1), P5(1)
I = O ⩽ T5 ⩽ 5

CLASS 9
M = P4(1), P5(1), P11(1)
I = 5 ⩽ T5 ⩽ 6
O ⩽ T12 ⩽ 1
O ⩽ T15 ⩽ 1

CLASS 10
M = P4(1), P5(1)
I = O ⩽ T5 ⩽ 2

CLASS 11
M = P4(1), P7(1)
I = O ⩽ T5 ⩽ 2

CLASS 12
M = P2(1), P7(1)
I = O ⩽ T2 ⩽ 5

CLASS 13
M = P2(1), P7(1), P9(1)
I = 5 ⩽ T2 ⩽ 6
O ⩽ T9 ⩽ 1
O ⩽ T13 ⩽ 1

CLASS 14
M = P2(1), P7(1)
I = O ⩽ T2 ⩽ 2

CLASS 15
M = P2(1), P5(1)
I = O ⩽ T2 ⩽ 2

TABLE 4 : List of state classes for the Alternating Bit Protocol

It is clear from Table 4 that only one message or acknowledgement will be in transit at a time (places P9, P10, P1 1 and P12 are, at most, marked with one token) : this assures that the time out is correctly set. Further, no duplicate messages may be delivered (transitions T7 and T10 alternate in all paths of the graph) and the net is live.

Among other communication protocols we have verified the properties of, is a much more complex bus allocation protocol taken from the local network REBUS developed at LAAS [14].

4.2. *Resource allocation mechanisms* :

This second example shows how starvation problems in resource allocation mechanism may be avoided by using temporal constraints.

The mechanism we start from is a "readers/writer" scheme. The corresponding net is represented in figure 8. The graph of classes for this net (here isomorphic to the graph of markings of the non timed net) is represented in figure 9.

The readers/writer problem represented involves three processes : two readers and one writer. Reader processes access the resource in a shared mode and the writer process accesses it in an exclusive mode. The trouble with the net represented in figure 8 is that the reader processes may act together such that the writer process is never allowed to access the resource.

This appears clearly in the graph figure 9 : there exists a cycle in this graph going only through classes in which at least one reader process is active and consequently the resource cannot be accessed by the writer.

Many possible methods can be used for overcoming this starvation problem, including access priority for the writer and queing of the requests for the resource. Here, we will use instead the (authoritative) method of limiting the durations of the read and update operations and the frequencies of access to the resource by the reader processes.

Let us set static firing intervals for transitions t_2 and t_4 to $[1,2]$ and for transition t_6 to $[2,4]$, reading then takes between 1 and 2 units of time to complete and writing takes between 2 and 4 units. Further, let us set a static firing interval of $[3,\infty]$ for transitions t_1 and t_3, this means that readers have to wait at least three unit of times before accessing the resource.

With these values for intervals, the TPN admits now 21 state classes and it can be checked on the graph of classes that any cycle in the graph contains a class from which the writer process is allowed to access the resource ; thus starvation is avoided.

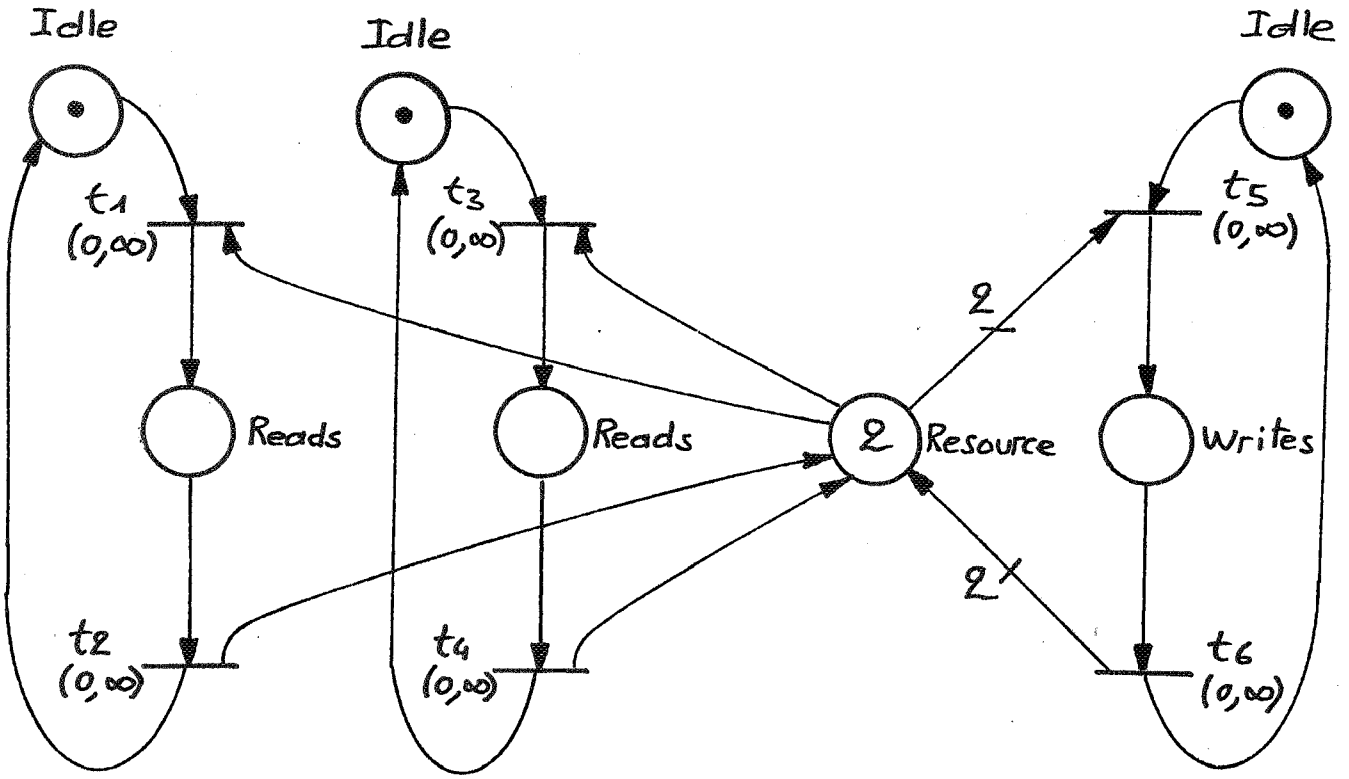


FIGURE 8 : the readers/writer example

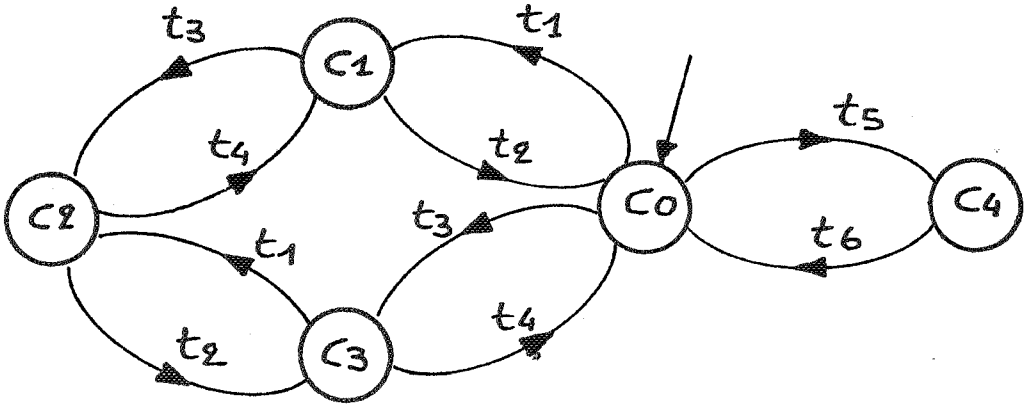


FIGURE 9 : Graph of classes for the TPN represented figure 8

4.2. *Other examples* :

The previous examples do not give, by far, an exhaustive view of what can be done using TPNs and the analysis technique introduced. This technique is general enough so it can be used for various purposes, from behavior analysis to performance evaluation.

We are currently investigating specific methods, based on the graph of classes, for this last field of applications. Using the graph of classes, and with some additional computations, questions of the following kind may be answered : in what states may the system stand after a given time elapsed since initialization or, how long may it take to reach a given state from another.

°0°

CONCLUSION

The analysis technique for TPN introduced in sections 2 and 3 suffers some limitations :

- A first limitation comes from the fact that no necessary and sufficient condition may be stated for the boundedness property, and this property is required for acheiving analysis. The only thing one could do for weakening this limitation would be developping sufficient conditions stronger than those stated in section 3 and checking user defined requirements on markings together with the boundedness property, so the enumeration would stop as early as possible if the behavior of the net is not that expected.
- A second limitation, also typical of the usual enumerative approach for analysing Petri Nets, is that the set of classes, even if it is bounded, may be very large. Using information contained in a very large graph of classes may be quite difficult, not talking about computation time or storage problems. Here again few can be done. However, an expert in Petri Nets will generally produce a net with a manageable number of classes when this can be done . One could think of using alternative analysis techniques for TPNs, such as, reduction techniques [15] or structural analysis techniques (use of place or transition invariants) [16][17] developed for Petri Nets.

For implementing the analysis technique, it is worth having a computer software package for enumerating the classes. We are developing, for carrying out experiments, a prototype of such a package, written in APL language. Several months of work convinced us that it would be worth to extend the Fortran written OGIVE/OVIDE [18] software product developed at LAAS (and soon to be made commercial by the Company SYSECA), so that it would allow also analyzing Time Petri Nets. This extension is scheduled.

°0°

REFERENCES

1 P. MERLIN

A study of the recoverability of computer systems

Ph.D. Thesis University of California, IRVINE Computer Science, 1974

2 P. MERLIN, D.J. FARBER

Recoverability of communications protocols

IEEE Transactions on Communications, September 1976, pp. 1036-1043

3 C. RAMCHANDANI

Analysis of asynchronous concurrent systems by timed Petri-Nets

Project MAC TR 120, Massachussets Institute of Technology,

February 1974

4 J. SIFAKIS

Use of Petri-Nets for performance evaluation in measuring modelling and evaluating computer systems

North Holland Publ. CO. 1977, pp. 75-93

5 W.M. ZUBEREK

Timed Petri-Nets and Preliminary Performance Evaluation

7th Annual Symposium on Computer Architecture, May 6-8.1980,

pp. 88-96

6 P. CHRETIENNE

Some Results on The Control of Timed Petri-Nets

2nd workshop on Applications and Theory of Petri-Nets

Bad Honnef, September 1981

7 P. CASPI, N. HALBWACHS

An Approach V. Real Time Systems Modelling

2nd workshop on Applications and Theory of Petri-Nets

Bad Honnef, September 1981

8 G.B. DANTZIG
Linear Programming and Extensions
Princelon University Press, 1963

9 N.D. JONES, L.H. LANDWEBER, Y.E. LIEN
Complexity of Some Problems in Petri-Nets
Theoretical Computer Science 4, 1977, pp. 277-299

10 M. HACK
Petri-Nets Languages. Computation Structures Group, memo 124.
Massachussets Institute of Technology Project MAC, June 1975

11 R.M. KARP and R.E. MILLER
Parallel Program Schemata
Journal of Computer and Systems Sciences 3 (1969) pp. 147-195

12 B. ASPVALL, Y. SHILOACH
A Polynomial Time Algorithm for Solving Systems of Linear Inequalities with two Variables for Inequality. 20th Annual Symposium on Foundations of Computer Sciences, Oct. 1979, pp. 205-217

13 K.A. BARTLETT, R.A. SCANTLEBURY, P.T. WILKINSON
A Note on Reliable Full-Duplex Transmission over Half-Duplex Link
Communication ot the ACM, Vol. 12, n° 5, May 1969, pp. 260-261

14 REBUS, J.M. AYACHE J.P. COURTIAT, M. DIAZ,
A Fault-Tolerant Distributed System for Industrial Real-Time Control
IEEE Transactions on Computers, Vol. C-31, n° 7, July 1982

15 G. BERTHELOT
Vérification des Réseaux de Petri
Thèse 3ème Cycle, Université Pierre et Marie Curie (Paris VI),
January 1978

16 K. LAUTENBACH, H.A. SCHMIDT
Use of Petri-Nets for Proving Correctness of Concurrent Process Systems
IFP Congress 1974, North Holland, Publ. Co. 1974, pp. 187-191

17 B. BERTHOMIEU
Carrying Proofs on Petri-Nets using their Structural Properties
(To Appear in IEEE Transactions on Software Engineering)

18 CHEZALVIEL-PRADIN B.
Un Outil Graphique Interactif pour la Vérification des systèmes à Evolution parallèle décrit par Réseaux de Petri
Thèse Docteur Ingénieur Université Paul Sabatier Toulouse, Décember 1979

19 S. GHOSH
Some Comments on Timed Petri-Nets
Journées sur les Réseaux de Petri, Paris 1977, AFCET.

°0°