R. GALLARD (*)

RESUMEN

El presente trabajo propone una extensión en la definición de la ejecución de Redes de Petri a fin de eliminar las limitaciones de estas redes en la modelización de sistemas reales, cuando uno desea respuestas a preguntas que van mas alla de un "si" o un "no" al analizar la aceptabilidad del comportamiento del sistema modelizado.

El mayor contenido de información en las respuestas del modelo se logra permitiendo al usuario entrar información acerca del conocimiento que el mismo tiene sobre algunas características de los requerimientos del sistema.

En consecuencia sera necesaria la introducción de un vector de probabilidades de ejecución de eventos habilitados, como una componente extra en cada nodo del arbol de alcanzabilidad y el uso de técnicas de Montecarlo.

ABSTRACT

In the present work an extension in the definition of a Petri Net execution is given in order to eliminate the limitations of these nets in the modellins of real systems, when one wishes to answer to questions beyond a "yes" or a "no", in analysins the acceptable behaviour of a modelled system.

The greater content of information in the answers of the model is achieved by allowins the user to enter information about the knowledge he has on some characteristics of system requirements, before the execution of the net.

Therefore, the introduction of an Execution Probability Vector for enabled events, as an extra component in each node of the Reachability Tree and the aid of Montecarlo techniques, are necessary.

(*) Computador Científico (UNBA 1975), MSc in Computer Science (University of Aston in Birminsham, U.K. 1982), Profesor Titular en la Escuela de Informatica de la Universidad Nacional de San Luis, Cátedras de Sistemas Operativos y Analisis Comparativo de Sistemas.
Chacabuco y Pedernera , 5700 - San Luis Argentina.

This paper is a consequence of the work of the author and coworkers in the study and implementation of Software for the sintax control, execution and analysis of Petri Nets.

In previous works it became clear for the group, as Peterson and other authors stated, the ability of these nets to detect pathological behaviours (deadlocks, violations of mutual exclusion restrictions, etc) in systems that exhibit concurrent activities.

For this purpose a Petri Net is a simple and powerful tool which can serve as a basis for ambitious projects, as that of Nelson, Haibt and Sheridan: "Specification, Design and Implementation via Annotated Petri Nets" [4]. In general, we notice that a Petri Net is a highly efficient medium for either the diagnosis of systems susceptible of exhibiting this type of failures or the elimination of these failures during the debugging stages.

Nevertheless, and this is a limitation in the definition of the execution of a Petri Net, this does not occur when one wishes to more faithfully model the behaviour of a real system.

Such a limitation is a consequence of the basic assumption that says that, any transition in a set of enabled transitions can fire. That "any" is translated, in the software which implements the execution, as a random selection of the transition to be fired with a uniform probability distribution (equiprobable).

This type of implementation is reasonable in the cases where one wishes to show,by any means (e.g the construction of the reachability tree or the execution of a graph), the reachability of an abnormal state in the system.

But, it is clear that for the one who wants to obtain more information than the mere detection of failures, such type of implementation is unsatisfactory.

For example, if a set of enabling tokens in some places (conditions which describes the state of the system) determines that certain transitions (possible events in the system) can fire, it is not always true that all the enabled events are equally probable to occur in the real system.

Otherwise, it is true that in the presence of many requeriments of a scarce resource from many processes there would be a priority policy to grant them. It is also true that the access to a particular type of resource could be more frequent than the access to other type of resources.

Such pre-stablished preferences in the real system are not possible to model with the simple assumption that any transition of an enabled set can fire without addings anything else.

The present work has the purpose of defining a selection function, to select a transition of a set of enabled transitions, based on the knowledge the designer or analyst has about the system and of explaining briefly, some details of a possible implementation.

## PREVIOUS CONCEPTS

Given a marked Petri Net  M = <S,μ>

where
μ is an initial marking and S is the Petri Net Structure defined by:

S = <P,T,I,O>

with  P = {p1,p2,...,pn}  Set of n places, n >= 0
T = {t1,t1,...,tm}  Set of m transitions, m >= 0
I : T --> P'  Input function
O : T --> P'  Output function
P' Set of all possible bags formed with elements of P,

the state space R(S,μ) of the network is formed by the markings which are reachable from the initial marking μ.

This Reachability Set, R(s,μ), has a finite representation by means of a Reachability Tree. Each node in this tree represents either a reachable marking or a set of reachable markings if the Petri Net is unbounded (presence of ω symbol in the tree).

In addition, for each of these nodes the subset Tau of enabled transitions is uniquely determined. This is true even in the cases where the ω symbol is present in a particular node, because the presence of extra tokens in a place affects neither the enabling nor the firing of transitions.

Consequently there is a unique set Tau of transitions that can fire for a particular marking (state of the Petri Net).

After highlighting these concepts we go now to the construction of the selection function E.

## CONSTRUCTION OF THE SELECTION FUNCTION "E"

We describe now the building-up process for the selection function.

i)
For a given marking μ' ∈ R(S,μ), we define as an Enabling Vector a m-vector H = (h1,h2,...hm) such that

$$h_i = \begin{cases} 1 & \text{if } t_i \text{ is enabled in } \mu' \\ 0 & \text{otherwise} \end{cases} \qquad 1 \leq i \leq m$$

ii)

We associate to each enabling vector a weighting vector W which indicates the weighting that one knows is made in the real system, about the possibility of execution for each of the competitive and enabled events in the currently modelled state of the net (described by the enabled transitions).

Example :

> If for a given marking we have M = (1,0,0,1) and the associated weighting vector is W = (3,0,0,1), we would be describing a situation where, being enabled transitions t1 and t4, t1 has a probability 3 times greater than t4 to be fired.

This W vector is subsequently "normalized" to W' where

$$W' = W / \sum_i wi = (3/4,0,0,1/4)$$

Briefly, W' defines a probability distribution for the sthocastic variable "transition to be fired".
Each component wi' of W' describes the firing probability of transition i

$$wi' = Pr \ ( \text{ firing ti } ).$$

This distribution is a probability one because

$$0 <= wi' <= 1 \qquad \text{and} \qquad \sum_i wi' = 1$$

iii)

Once the vector W' is obtained and by means of the well known application of the Montecarlo method, we have a decision rule to determine the transition to be fired.

If we call TAU to the class of all sets (Tau) of enabled transitions to be generated during Petri Net execution, we see that our selection function, so constructed, takes a Tau and for this Tau allows us, based in the knowledge one has about the system, to define a vector W' which is a descriptor of a probability distribution f to which applying the Montecarlo techniques returns us the transition to be fired.

The last paragraph can be rewritten as

$$E : TAU \longrightarrow Tau$$

that can be decomposed in

$$E1 : TAU \xrightarrow{m} M$$
$$E2 : M \xrightarrow{m} N$$
$$E3 : N \xrightarrow{m} Q$$
$$E4 : Q \xrightarrow{m} F$$
$$E5 : F \longrightarrow Tau$$

. . E = E5 . E4 . E3 . E2 . E1

where

TAU : The class of all sets of enabled transitions for a given Petri Net, where each set corresponds to a particular reachable markings.

$H^m$ : The set of all m-vectors whose elements are binary digits.

$N^m$ : The set of all m-vectors whose elements are positive or null integers.

$Q^m$ : The set of all m-vectors whose elements are positive rationals, less than or equal to 1 and they add up to one.

F : The set of all discret probability distribution functions with m elements in their domain.

Tau : The set of enabled transitions for a given reachable markings in the Petri Net.

## SOME SUGGESTIONS FOR THE IMPLEMENTATION

i)

The concept of a node in the reachability tree, will be extended for a routine of tree generation, in order that each of these nodes will be formed by the pair:

$$(\mu, H) = (\text{ Current Markings, Current Enabling Vector })$$

ii)

Afterwards it will be necessary to write a routine (*) that, in traversing the tree and based on the user's knowledge, creates the corresponding weighting vector W for that particular state in the net. From this, the probability vector W' can be determined and then the enabling vector can be replaced in the node, which now takes this form

$$\text{Tree node} = (\mu, W')$$
$$= (\text{ Current Markings, Probability Vector associated with this markings })$$

iii)

During Petri Net execution, in each one of its states, the vector W' associated with the current marking will be fetched and a routine to apply the Montecarlo method will be called. In this way the transition to be fired is determined.

(*) It could be the same decribed in i).

## CONCLUSIONS

As stated in the introduction, this type of implementation in the execution of a Petri Net would allow a behaviour of the model, granularly closer to that of the system state modelled in each instance of the net.

In the cases in which the tokens' distribution in the net is modelling the requirements of certain resources, it is possible for us to observe the accumulation of tokens in some places, and this fact would be modelling the feeding of requirements queues, whose parameters (mean and maximum queue lengths, mean waiting time in the queue meassured in number of firings, etc) could be registered for further analysis.

The assignation of a firing probability equal to 1 or 0 to one of the enabled transitions, would naturally describe a priority policy without needing the use of inhibitor arcs.

As regards to the analysis of Liveness and Coverability, related with the problem of reachability, we see now that in the case of being able to answer affirmatively to questions such as :

- Is $\mu'$ reachable from $\mu$ ?
- Is there a reachable marking $\mu'$ from $\mu$, such that $\mu'$ covers a given marking $\mu''$?
- Is it possible for the net to arrive to state where no transition can fire ?
- Is there any reachable marking where the mutual exclusion restrictions are violated ?

we are now able not only to answer "yes", but also we would further precise our answer specifiying the arrival probability to such a state after k firings.
This is possible because,

$$\text{if } \mu' = \hat{\delta}(\mu, \text{Sigma})$$

where $\hat{\delta}$ is the extended next state function and Sigma is a firing sequence of length k ,then the arrival probability to $\mu'$ would be given by the product of the probabilities associated with the firing transitions in each of the succesive markings (states) through which the Petri Net goes.

Finally, we believe that this extended definition of a Petri Net execution improves its modelling ability allowing a more faithful representation of real systems behaviour and a better understanding of them by answering an important set of questions to which the traditional approach has no answer.

ACKNOWLEDGEMENTS

This work would not have been possible without the motivation supplied by Dr Alan J. Harget of the University of Aston in Birmingham (U.K.) who in 1982 introduced me in the subject.

I am also particularly grateful to the students Susana G. Barbenza and Mirtha S. Escudero for their colaboration in the development of a Petri Net Software and in general to the project group in Operating Systems of the Universidad Nacional de San Luis for their work in the area.

San Luis, Argentina Septiembre de 1984

BIBLIOGRAPHY

[1] C. Petri, "Kommunikation mit Automaten", Ph. D. dissertation, University of Bonn, Bonn, West Germany.

[2] J. Peterson, "Petri Nets", Computer Surveys, Volume 9, Number 3, September 1977.

[3] J. Peterson, "Petri Net Theory and the Modeling of Systems", Prentice Hall, 1981

[4] R. A. Nelson, L.M. Haibt and P.B. Sheridan, "Specification Design and Implementation via Annotated Petri Nets", IBM T. J. Watson Research Centre Yorktown Heights RC 9317 March 1982.

[5] R. Gallard, "A Set of Petri Net Routines for System Modelling" University of Aston in Birmingham, Birmingham, U.K.,July 1982.