Monograph:
Jalel, N.A. and Nicholson, H. (1990) Petri Nets and Fault Diagnosis in Nuclear Reactors. Research Report. Acse Report 413 . Dept of Automatic Control and System Engineering. University of Sheffield

Petri Nets and Fault Diagnosis in Nuclear Reactors

by

N. A. Jalel and H. Nicholson

Department of Control Engineering
University of Sheffield
Mappin Street
Sheffield S1 3JD

Research Report No. 413
November 1990

1 Abstract

The possibility of applying Petri nets (Pns) as a modelling tool to represent any fault or accident that might occur in the Loss Of Fluid Test (LOFT) reactor, small scale pressurised water reactor, is discussed and analysed. Pns are developed to assist the nuclear reactor operator in identifying any fault or alarm that might arise in the power station.

2 Introduction

Petri nets have been used in a range of applications including computer operating systems, computer software, computer architecture, compiler optimization, legal systems, formal language theory, communication protocols, chemical systems, interrelationships of mathematical structures, office information systems, performance evaluation and fault tolerant systems, industrial process control, distributed database systems and formal verification of parallel programs. The applications of Pns as a modelling tool in the field of computer hardware and software are given by Hura [1,3].

The problems of performance evaluation and scheduling, fault diagnosis and isolation, fair regulation in resource-sharing systems and a distributed database model have been analysed and solved using Pns, timed Pns and coloured Pns [6]. The possibility of using Pns as interpretation tools which involve representing systems in a top-down fashion to represent operating systems and compilers such as the flow of control in programs, to model distributed data bases and communication protocols and to represent computer hardware are illustrated by Agerwala [8].

In the field of fault detection, Pns can be used to represent fault trees and the analytic properties of the resulting Pn can be determined using the reachability concepts. These properties will be used to provide useful information for fault tree analysis and to solve the problems of fault detection and fault propagation. Fault tree analysis offers a logical method for evaluating the reliability of a system in terms of component states and failure probabilities [4].

In this paper the idea and characteristics of Pns are illustrated and discussed, the modelling of logic operations is analysed, the new approach of using Pns for fault diagnosis and alarm analysis in nuclear reactors is illustrated and finally the ability of the algorithm is discussed and examined for a hypothetical accident.

3 Petri nets

Petri nets originated in C. A. Petri's doctoral dissertation "Communication with automata" in 1962 at the University of Bonn, West Germany, and were used as a new modelling tool to represent interacting concurrent and synchronous components within a system. During the last few years interest has grown in the use of Pns which provide a very useful and powerful modelling tool.

Pns are an ideal tool for modelling various classes of systems, especially the class of discrete event systems which involve parallel computations and concurrent processes. Analysis of the Pn can then reveal important information about the structure and dynamic behaviour of the represented system.

Petri nets have received wide attention and seem to be gaining increased acceptance and preference over other modelling tools especially for concurrent processing systems. They are simple in concept but have powerful representation properties which allow them to show concurrency in a simple and natural way. They provide a graphical and precise representation scheme so it is easy to understand the overall system, and can model the system hierarchically. They provide an analysis of the behaviour of the system through the movement of tokens and have the ability to model software concepts like parallelism, conflict and synchronization of events [2,9,7,10,11].

3.1 The Structure of Petri Nets

Petri nets are a particular kind of directed graph, composed of two basic components, a set of places (P) pictorially represented by circles (◯) and a set of transitions (T) usually represented by short lines or vertical bars (|), and describe two functions which establish the relationship between the places and transitions, the input function (I) and the output function (O). The input function I defines, for each transition t_j, the set of input places for each transition $I(t_j)$, while the output function defines the set of output places for the transition $O(t_j)$. Directed arcs may connect the different types of components but not components of the same type. An arc from a place component to a transition is called an input arc while an output arc is an arc from a transition to a place. An input place is a place connected to a transition by an input arc and a place connected to a transition by an output arc is called an output place.

The structure of Pns can be represented mathematically to help in understanding the graphical structure by defining
C = (P, T, I, O)

where
P is a finite set of places $(p_1, p_2, \ldots), p \geq 0$
T is a finite set of transitions $(t_1, t_2, \ldots), t \geq 0$
$I : T \rightarrow P_{in}$ is the input function, a mapping from transitions to input places
$O : T \rightarrow P_{out}$ is the output function, a mapping from transitions to output places
A place P_i is an input place of T_j if P_i belongs to $I(t_j)$ and a place P_i is an output place of T_j if P_i belongs to $O(t_j)$

3.2 Marked Petri Nets

A marking μ is an assignment to the places of a Pn of a non-negative integer number of marks called tokens. A place is said to have k tokens, if a non-negative integer k marking is assigned to the place. A place not containing tokens is said to be an empty place. The number and position of tokens residing in the places of the net may change during the execution of the net.

A token can be thought of as representing initial conditions or the holding of some data items associated with that place and pictorially represented by a black dot • in the circle which defines the place of a Pn.

Mathematically the marking μ can be defined as a vector

$$\mu = (\mu_1, \mu_2, \ldots, \mu_n)$$

where $\mu_i \, (i = 1, \ldots, n)$ is the number of tokens in place P_i.
A marked Pn is defined as a Pn C = (P, T, I, O) with a marking vector μ and written as M = (P, T, I, O, μ).

The execution of a Pn is controlled by the number and distribution of tokens in the net and is achieved by firing a transition. A transition may fire if it is enabled, where a transition T is said to be enabled if each input place P of T is marked with at least as many tokens as the number of arcs from P to T, which means that all the input places to a transition should contain at least one token. A transition fires by removing one token (per arc) from each input place, and then deposits one token (per arc) to each output place. Firing a transition will change the marking of the Pn from μ to a new marking $\acute{\mu}$. Only one transition may fire at a time and in the case of more than one transition enabled, the transition to fire is chosen randomly.

Using the concept of conditions and events to model fault trees, an event is represented by a transition, input places containing the token correspond to the conditions for the occurrence of the event, and output places denote the conditions after the occurrence of the event which is signalled by the firing of a transition.

Figure 1 shows a simple marked Pn which can be defined as

M = (P, T, I, O, μ)

P = (p_1, p_2, p_3, p_4, p_5)

T = (t_1, t_2, t_3, t_4, t_5)

I(t_1) = (p_1), O(t_1) = (p_2, p_3)

I(t_2) = (p_2), O(t_2) = (p_5)

I(t_3) = (p_2), O(t_3) = (p_5)

I(t_4) = (p_3), O(t_4) = (p_4)

I(t_5) = (p_4), O(t_5) = (p_1)

μ = (1, 0, 0, 0, 0)

In the Pn of figure 1, only transition t_1 is enabled. When t_1 fires, a token is removed from place p_1 and a token is placed in p_2 and p_3, resulting in a new marking $\acute{\mu}$ = (0, 1, 1, 0, 0) as shown in figure 2.

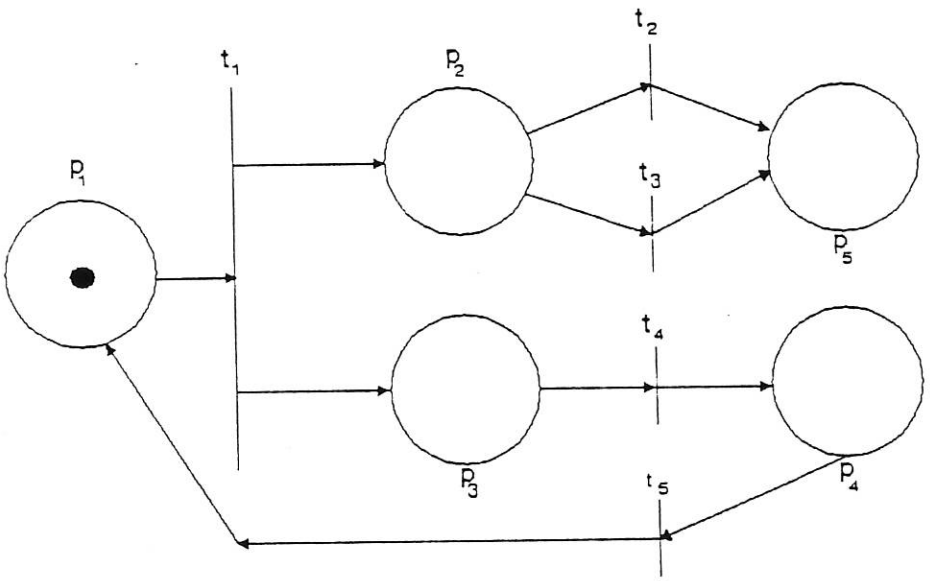

Figure 1: Marked Petri net transition t_1 is enabled

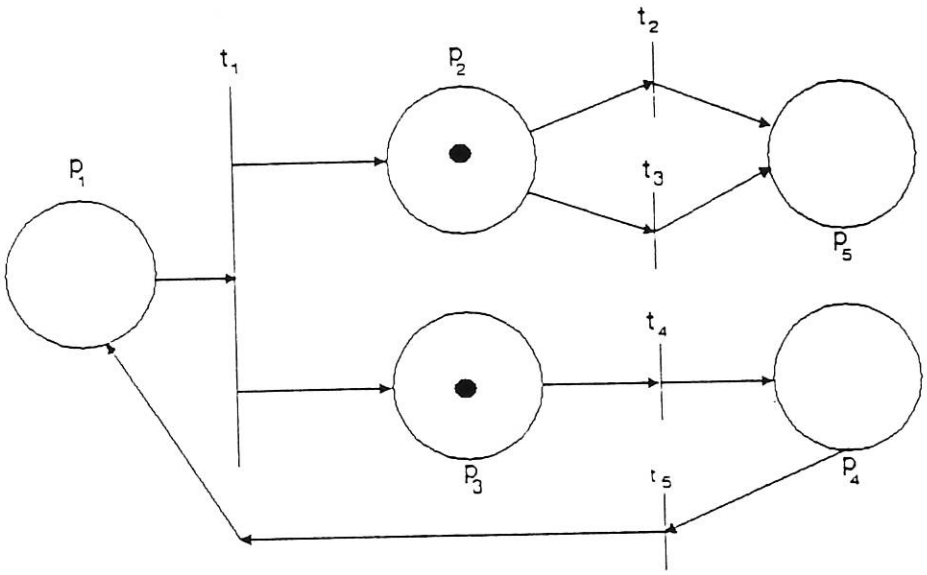

Figure 2: Petri net after the firing of transition t_1

3.3 Incident Matrix and State Space Equations

In analysing Pns one of the important problems is the determination of the reachability set. Reachability sets are obtained by either a matrix equation or by considering the firing sequences. The reachability concept along with the firing sequence has been presented through a state space equation proposed by Murata [5] which gives the next marking μ_{k+1} from its previous marking μ_k.

The incident matrix of a Pn A = $[a_{ij}]$ is a p $\times$ t matrix of integers where p is the number of places and t denotes the number of transitions in the net. The matrix entry a_{ij} is given to be equal to 1, -1, or 0 if transition i has an outgoing arc to place j, an incoming arc from place j or no arc between them, respectively.

A marking vector μ_k is the p $\times$ 1 column vector of non-negative integers whose elements p_j gives the number of tokens in place j. The elementary firing or control vector U_k is a t $\times$ 1 column vector containing one non-zero entry, 1, in the ith position indicating transition i fires at the kth firing.
The state space representation of a Pn is given by

$$\mu_{k+1} = \mu_k + A^T U_k \qquad k = 1, 2 \tag{1}$$

In the case of an existing firing sequence (U_1, U_2, ..., U_d) that transforms an initial

marking μ_0 to some marking say μ_d, then the state equation is

$$\mu_d = \mu_0 + A^T \sum_{k+1}^{d} U_k$$

which may be written as

$$\mu_d = \mu_0 + A^T \sum$$

or

$$\Delta\mu = A^T \sum \tag{2}$$

Where $\sum$ is a t $\times$ 1 column vector of non-negative integers called the firing-count vector and $\Delta\mu = \mu_d - \mu_0$.

4 Modeling of Logic Operations Using Petri Nets

The AND and OR logic operations can be represented using the state space equation 2. The Pn model of an AND gate is illustrated in figure 3 while figure 4 shows the Pn model of an OR gate. From figure 3, p_1 and p_2 are the input places to transition t while p_3 is the output place. The incident matrix A^T is obtained as

$$A^T = \begin{matrix} \\ p_1 \\ p_2 \\ p_3 \end{matrix} \begin{matrix} t \\ \left[\begin{matrix} -1 \\ -1 \\ 1 \end{matrix}\right] \end{matrix}$$

For the firing of t, both p_1 and p_2 should have at least one token, so the initial marking is

$$\mu_0 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

From figure 3, firing t will deposit a token in p_3, hence the final marking

$$\mu_1 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Thus, if μ_0 and A^T are known, the state space can be used to find another state

$$\mu_1 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

As shown in figure 4, p_1 and p_2 are the input places to t_1 and t_2 respectively while p_3 is the output place. The OR gate is realized through the state space equation under any of the three conditions; t_1 fires alone, t_2 fires alone or both t_1 and t_2 fire. However, in the last condition the output place p_3 can have two tokens after the firing of t_1 and t_2 but for simplicity, the presence of two tokens in p_3 will only indicate the presence of data in p_3. The execution of the OR operation is found to be the incident matrix A^T of the OR gate

$$A^T = \begin{array}{c} \\ p_1 \\ p_2 \\ p_3 \end{array} \begin{array}{c} \begin{array}{cc} t_1 & t_2 \end{array} \\ \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 1 \end{bmatrix} \end{array}$$

Taking the initial marking as

$$\mu_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

Then the final marking can be obtained using the state equation as

$$\mu_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

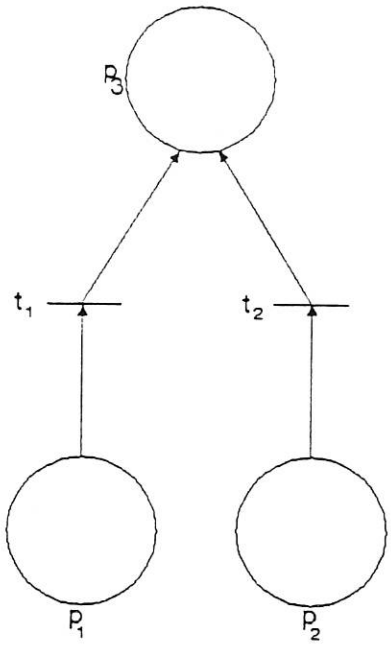

Figure 4: The OR gate model

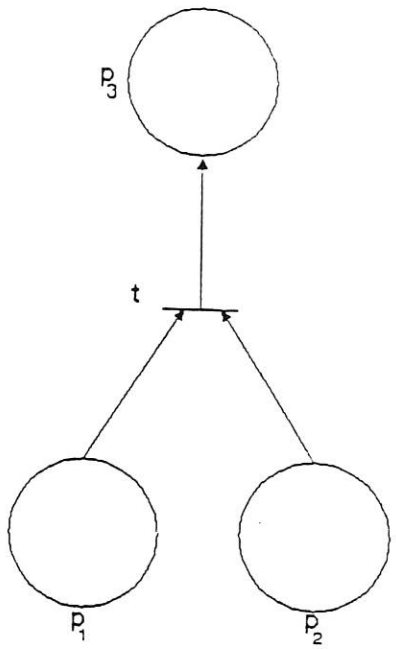

Figure 3: The AND gate model

5 Petri Nets for Fault Diagnosis and Alarm Analysis

A Pn is used to model the fault trees for the different accidents and alarms that might occur in the LOFT reactor model [14]. Using the reachability concept and the state space equation the modelled Pn can be used to assist the nuclear reactor operator in diagnosing any faults or accidents and in identifying any alarms that might happen in the reactor.

The initial marking of the input places for the net are the symptoms for the accidents or alarms that could occur in the reactor. The symptoms are found through monitoring the state variables of the simulated LOFT model for any deviation from the normal operating limits and by asking the operator questions concerning the availability of some information about the whole power plant.

The input place is marked with a token if the symptom has occurred, other wise the input place is considered empty referring to the normal operation situation. The state space equation for the Pn is applied to obtain the final marking of the Pn and the function at the last place in the net.

The accident is defined to occur if the final marking indicates the occurrence of a token in the output place while normal operation is assumed if the last place is empty.

The Pn model of the fault tree for the excessive feedwater heat removal accident [14] shown in figure 6 is illustrated in figure 5 and is obtained by replacing the AND and OR gates by their equivalent Pn model shown in figure 3 and figure 4 respectively. This type of accident is selected as an example to illustrate and discuss the Pn approach in diagnosing the accidents. From figure 5, p_1 and p_2 are the input places for t_1 and t_2 respectively and they are referred to a further decrease in feedwater temperature and a further increase in feedwater flow symptoms. While p_3 and p_4 are referred to a further decrease in average temperature coolant and further increase in reactor power symptoms, both of them with p_5 are considered as input places for t_3 while p_6 is the output place for the Pn.

For the accident to occur, a token should be deposited in p_6 through firing t_3 where t_3 is enabled if p_3, p_4 and p_5 have at least one token. A token is deposited in p_5 through firing either t_1 or t_2 where t_1 and t_2 are fired if a token is found in p_1 and p_2 respectively. Generally, firing t_3 can be achieved through three conditions:

1. Token deposits in p_1, p_3 and p_4
2. Token deposits in p_2, p_3 and p_4
3. Token deposits in p_1, p_2, p_3 and p_4

Places p_1, p_2, p_3 and p_4 are marked with tokens if the referred symptoms for the described accident have occurred. Thus, the first and second condition can achieve the same result as the third condition and can represent the third condition, so the Pn illustrated in figure 5 can be modified into a small model consisting of one transition, three input places and one output place as illustrated in figure 7. The Pn model shown in figure 7 is used twice, first with the input set given in the first condition and then with the input set given in the second condition, and in both cases the output place is monitored for any token. The accident is diagnosed if the output place is marked with a token in any case while if no token is found in the output place then this accident has not occurred in the reactor.

The incident matrix for the state space Pn model is given as

$$A^T = \begin{array}{c} \\ p_1 \\ p_2 \\ p_3 \\ p_4 \end{array} \begin{array}{c} t \\ \left[\begin{array}{c} -1 \\ -1 \\ -1 \\ 1 \end{array}\right] \end{array}$$

The initial marking μ_0 is given to be either the first condition marking or the second condition marking. If tokens are assumed to deposit in p_1, p_2 and p_3 places then the initial marking is defined as

$$\mu_0 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Applying the state space equation 2, the final marking is found to be

$$\mu_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 \\ -1 \\ -1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

From the final marking vector it is possible to recognize a token in the output place (p_4), indicating the occurrence of the accident.

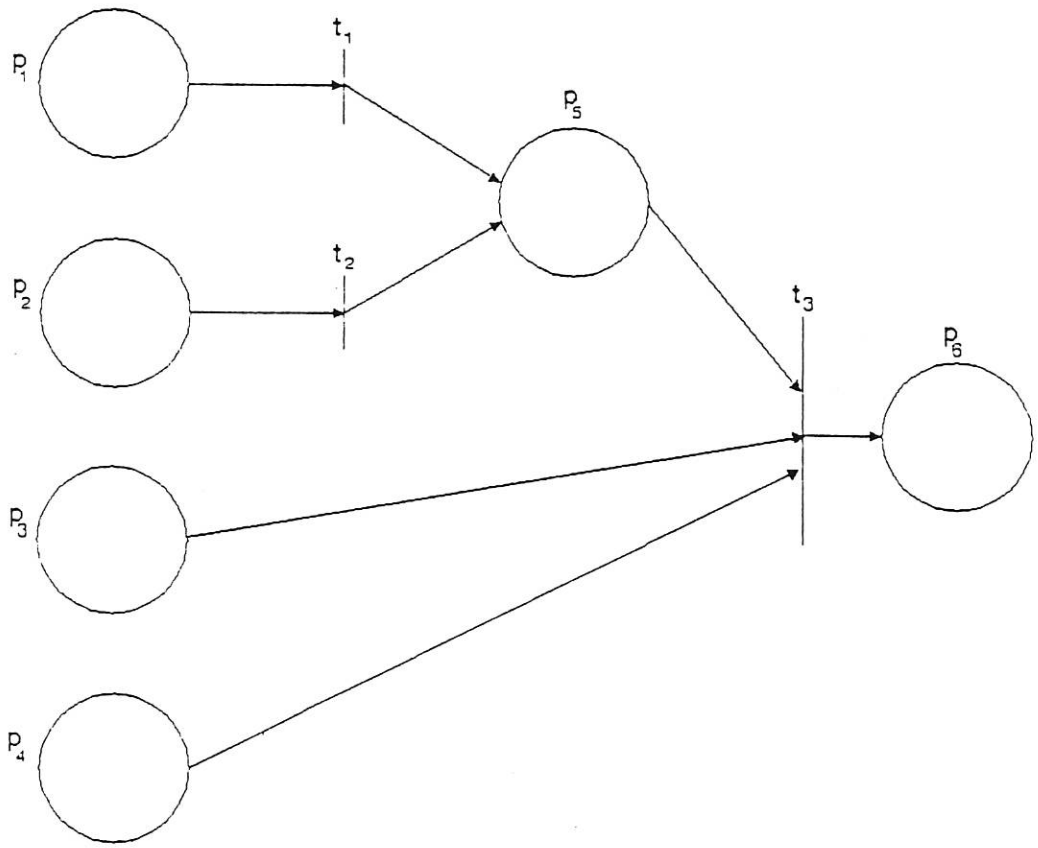

Figure 5: Petri net model for excessive feedwater heat removal

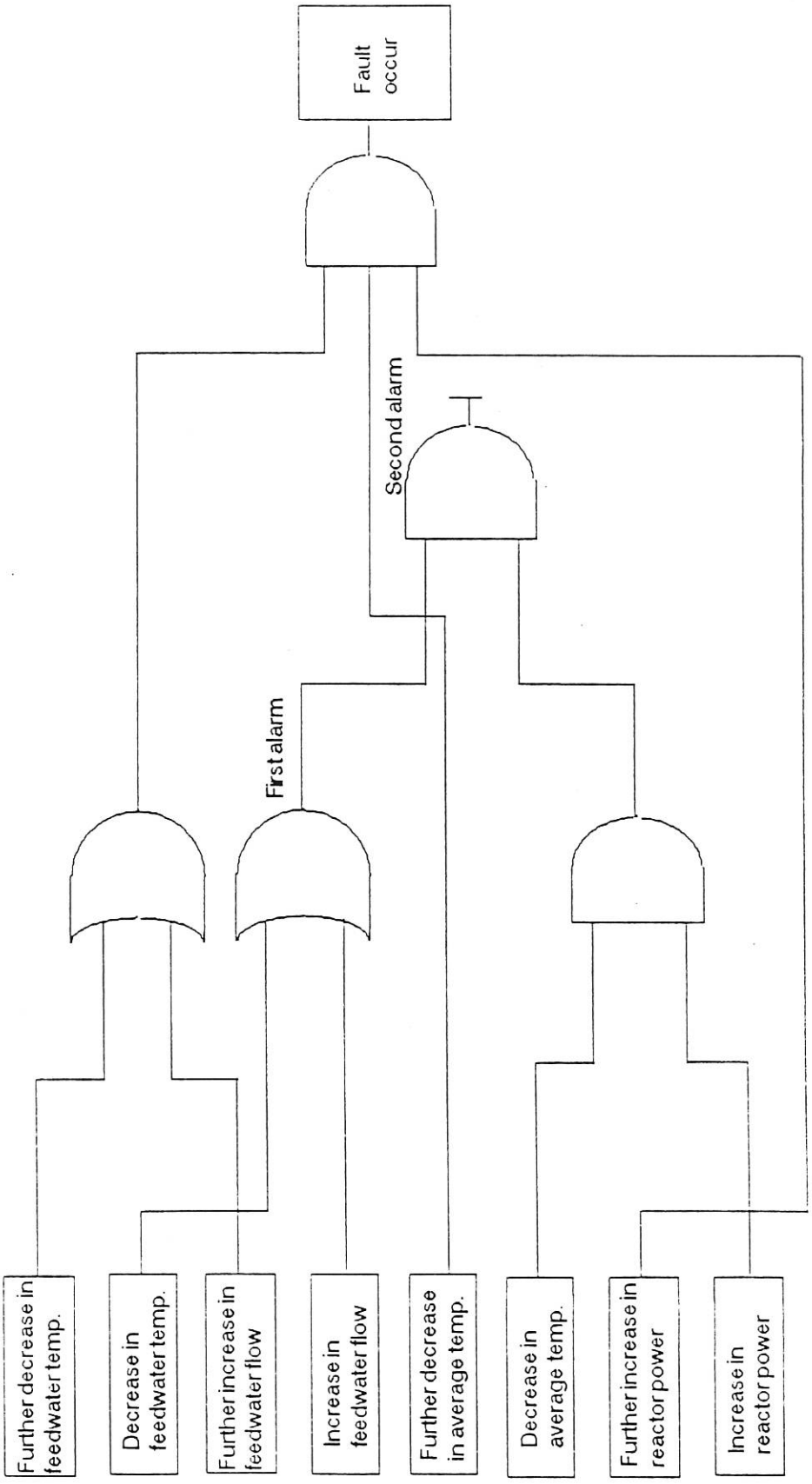

Figure 6: Fault tree for excessive feedwater heat removal

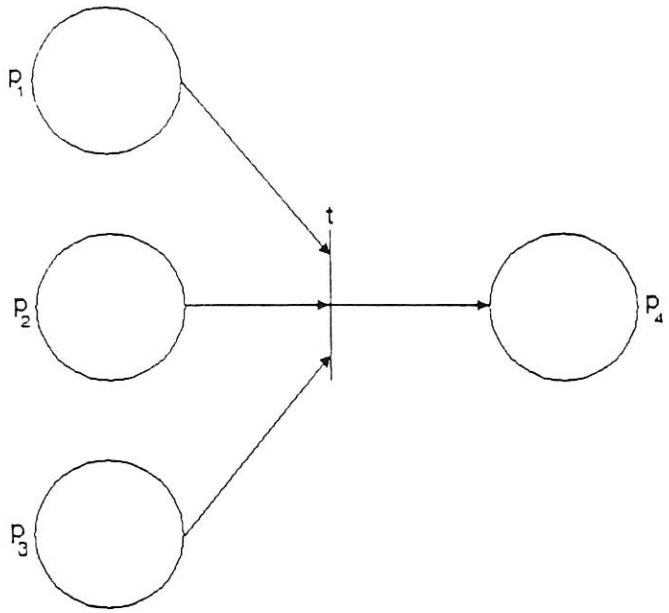

Figure 7: Petri net with three input places

6 Petri Net Performance

A computer package has been written in C language and run on a Sun workstation. The system is intended to simulate the LOFT reactor model, monitor the state variables of the LOFT model for any increase or decrease from the normal operating limit, interrogate the system using questions concerning the occurrence of some symptoms, arranging and assigning the initial marking for the Pns, calculate the final marking of the Pns through the use of the state space equation, and finally diagnose any accidents and identify any alarms in the reactor by analyzing the final marking condition. The system is designed to perform these procedures every 2.5 sec.

A hypothetical accident is assumed to occur in the LOFT model to test and examine the ability of Pns to diagnose the accident. The accident is a blockage in the feedwater piping which is initiated at a time greater than 3 sec by a drop in feedwater flow from 57.64 (lbm/sec) to 0.0 (lbm/sec) and by an increase in the feedwater pump pressure from 942 to 1000 (psia).

The fault tree for this accident is shown in figure 8, where the symptoms of the accident include a decrease in steam generator water level and feedwater flow, increase in feedwater discharge pressure and reactor coolant average temperature and the feedwater pump contact breaker is open. This accident is modelled using

a Pn with five input places shown in figure 9 and for the accident to occur the five input places should have at least one token, referring to the occurrence of the five symptoms, thus enabling the transition to fire and deposit a token in the net output place to assure the occurrence of the accident.

Table 1 illustrates the results of the system where at time 0 and 2.5 sec before the occurrence of the accident, the system indicates the normal operation situation in the reactor. At time 5 sec the first alarm for blockage in the feedwater is indicated by the system, thus the second alarm for the blockage in feedwater piping is announced at time 7.5 sec. In addition to the above two alarms, the system warns the user of the occurrence of the first and second alarms for the partial loss of flow due to blockage in the fuel assembly at time equal to 12.5 sec. The system reaches the final conclusion and indicates the occurrence of the failure in the system due to the blockage in the feedwater piping at time equal to 15 sec.

7 Conclusion

It has been shown that the logic tree for any fault or accident can be modelled by Pns, and by using the reachability concepts and the state space equation the Pn model can be analysed in a simple way. Analysing the model will assist the nuclear plant operator in diagnosing any faults or alarms that might occur in the reactor. The results of the present study are very encouraging, and the system can be extended to model different accidents or alarms that could occur.

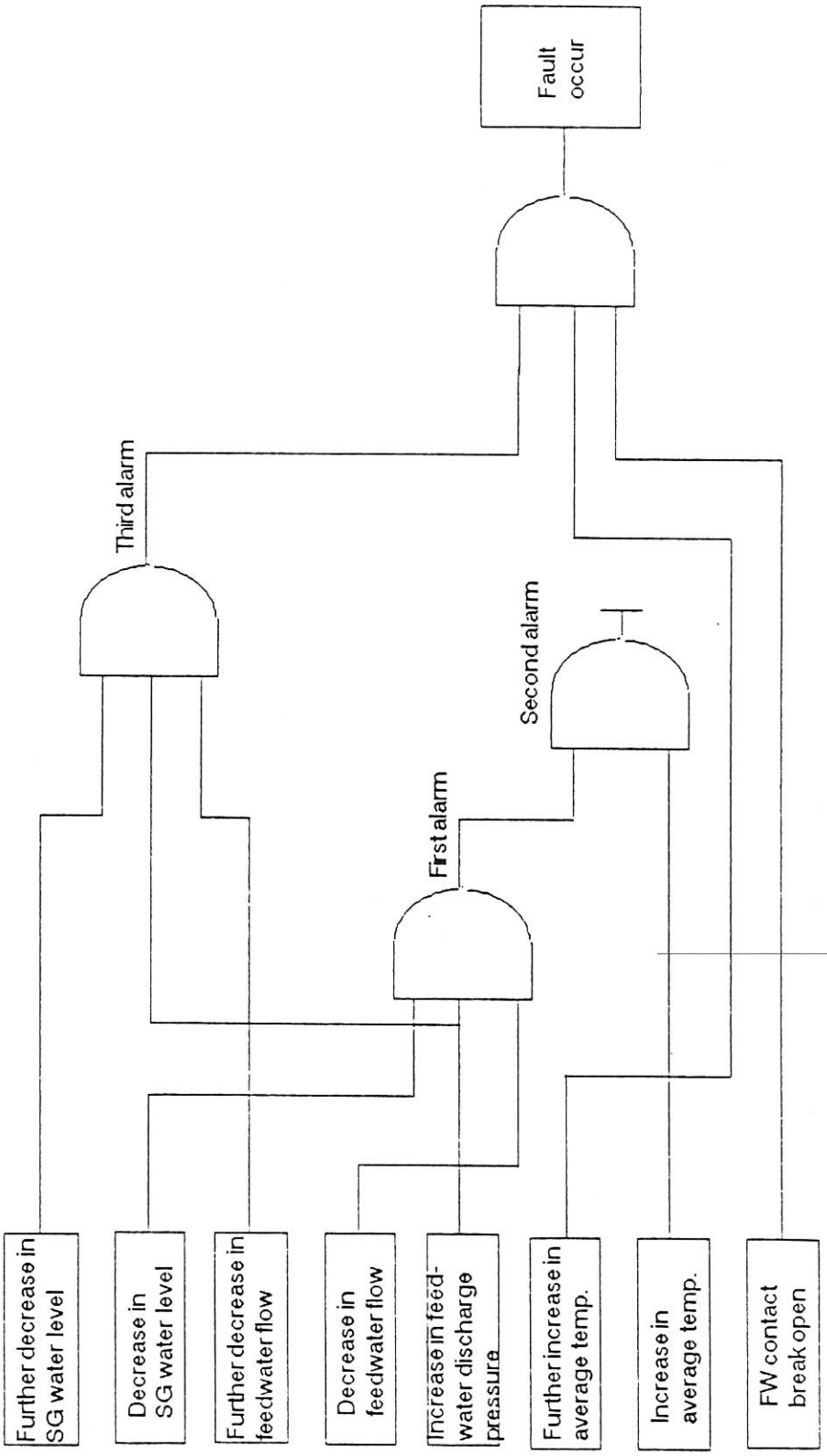

Figure 8: Fault tree for blockage in feedwater piping

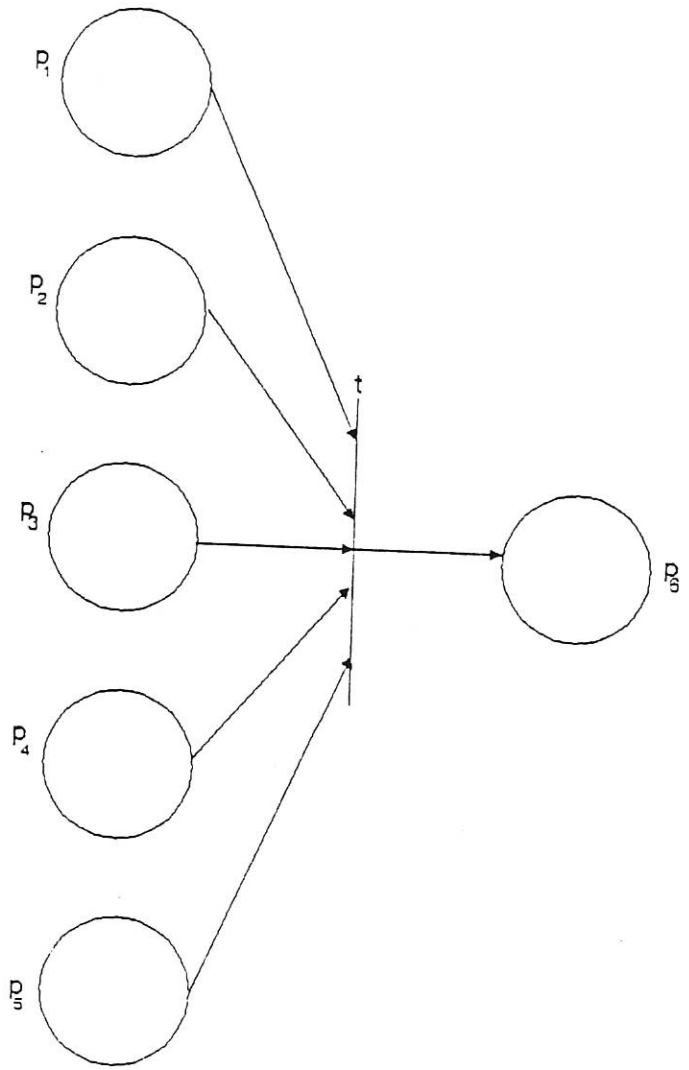

Figure 9: Petri net with five input places

Time	Faults and alarms
0.0	I can't identify any accident in the reactor
2.5	I can't identify any accident in the reactor
5.0	First alarm for blockage in feedwater piping I can't identify any accident in the reactor
7.5	First alarm for blockage in feedwater piping Second alarm for blockage in feedwater piping I can't identify any accident in the reactor
10.0	First alarm for blockage in feedwater piping Second alarm for blockage in feedwater piping I can't identify any accident in the reactor
12.5	First alarm for blockage in feedwater piping Second alarm for blockage in feedwater piping First alarm for partial loss of flow due to blockage in a fuel assembly Second alarm for partial loss of flow due to blockage in a fuel assembly I can't identify any accident in the reactor
15.0	First alarm for blockage in feedwater piping Second alarm for blockage in feedwater piping Third alarm for blockage in feedwater piping First alarm for partial loss of flow due to blockage in a fuel assembly Second alarm for partial loss of flow due to blockage in a fuel assembly Failure due to blockage in feedwater piping
17.5	First alarm for blockage in feedwater piping Second alarm for blockage in feedwater piping Third alarm for blockage in feedwater piping First alarm for partial loss of flow due to blockage in a fuel assembly Second alarm for partial loss of flow due to blockage in a fuel assembly Failure due to blockage in feedwater piping

Table 1: System conclusion with respect to time

References

[1] Hura, G. (1982). *Petri Net as a Modeling Tool.* Microelectron. and Reliab. (GB), Vol. 22, No. 3, pp. 433-439.

[2] Hura, G. (1987). *Petri Nets.* IEEE Potentials, Vol. 6, No. 3, pp. 18-20.

[3] Hura, G. (1987). *Petri Net Applications.* IEEE Potentials, Vol. 6, No. 3, pp. 25-28.

[4] Hura, G. and Atwood, J. (1988). *The Use of Petri Nets to Analyze Coherent Fault Trees.* IEEE Transactions on Reliability, Vol. 37, No. 5, pp. 469-473.

[5] Murata, T. (1977). *State Equation, Controllability, and Maximal Matchings of Petri Nets.* IEEE Transactions on Automatic Control, Vol. 22, No. 3, pp. 412-416.

[6] Murata, T. (1984). *Petri Nets and Their Applications an Introduction, in, Management and Office Information Systems, Ed. Chang, S.* Plenum Publishing.

[7] Murata, T. (1987). *Petri Nets, in, Systems and Control Encyclopedia, Theory, Technology, Applications., Ed. Singh, G., Vol. 6.* Pergamon.

[8] Agerwala, T. (1979). *Putting Petri Nets to Work.* IEEE Computer, Vol. 12, No. 12, pp. 85-94.

[9] Johnsonbaugh, R. and Murata, T. (1982). *Petri Nets and Marked Graphs-Mathematical Models of Concurrent Compution.* Am. Math. Mon., Vol. 89, No. 8, pp. 552-556.

[10] Peterson, J. (1977). *Petri Nets.* Computing Surveys, Vol. 9, No. 3, pp. 223-252.

[11] Peterson, J. (1981). *Petri Net Theory and The Modeling of Systems.* Prentice-Hall, Inc.

[12] Tylee, J. (1980). *Low Order Model of the Loss Of Fluid Test (LOFT) Reactor Plant for Use in Kalman Filter Based Optimal Simulation.* Proceedings of the 4th Power Plant Dynamics, Control and Testing Symposium, Gatlinburg, pp. 1-31.

[13] Tylee, J. (1980). *Low Order Model of the Loss Of Fluid Test (LOFT) Reactor Plant for Use in Kalman Filter Based Optimal Estimators.* EGG-2006, EG & G Idaho Falls, Idaho.

[14] Jalel, N. (1990). *Fault Diagnosis and Accident Analysis in Nuclear Reactors.* PhD Thesis to be submit to the Department of Control Engineering, University of Sheffield.