Nuclear Engineering and Technology 55 (2023) 861–869

Contents lists available at ScienceDirect

Nuclear Engineering and Technology

journal homepage: www.elsevier.com/locate/net

Original Article

Performance measurement of safety-critical systems based on ordinary differential equations and Petri nets: A case study of nuclear power plant

Nand Kumar Jyotish [a, *], Lalit Kumar Singh [b], Chiranjeev Kumar [a]

[a] The Department of Computer Science & Engineering, Indian Institute of Technology (ISM), Dhanbad, Jharkhand, India
[b] The Department of Computer Science & Engineering, Indian Institute of Technology (BHU), Varanasi, India

ARTICLE INFO

Article history:
Received 28 February 2022
Received in revised form
16 November 2022
Accepted 21 November 2022
Available online 29 November 2022

Keywords:
Performance
Petri nets
Safety Critical Systems
Ordinary Differential Equations
Markov Chain

ABSTRACT

This article proposes a novel approach to measure the performance of Safety-Critical Systems (SCS). Such systems contain multiple processing nodes that communicate with each other is modeled by a Petri nets (PN). The paper uses the PN for the performance evaluation of SCS. A set of ordinary differential equations (ODEs) is derived from the Petri net model that represent the state of the system, and the solutions can be used to measure the system's performance. The proposed method can avoid the state space explosion problem and also introduces new metrics of performance, along with their measurement: deadlock, liveness, stability, boundedness, and steady state. The proposed technique is applied to Shutdown System (SDS) of Nuclear Power Plant (NPP). We obtained 99.887% accuracy of performance measurement, which proves the effectiveness of our approach.

1. Introduction

Safety-critical embedded systems could result in a loss of life, significant property damage, environmental damage, or loss of goal-oriented mission, if they fail. These are the computer-based systems (CBS) on which we rely on a daily basis. The most effective strategy to avoid these failures is to remove or minimize dangers early in the design and development phase, instead of later when the system becomes unmanageable complex. Such systems have grown in network connectivity and distribution, and thus become more perplexed. The growing complexity of the system may impact their overall performance. Thus, it is necessary to model such systems for performance measurement before its actual implementation.

The results of performance measurement using system model help to identify any potential bottlenecks to take design decisions. A model can be thought of as a conceptual abstraction of a particular system. In the past few decades, Researchers have increasingly relied on analytical tools to measure various performance metrics. Many of these analytical methods use the Petri net, which can explain the information flow of a system in a more meaningful way and compute several dependability attributes [1].

Many CBS are used to take important decision of to perform important measurement [2,3] and hence measuring their performance is essential. In this research work, a novel approach is devised to measure the performance of SCS that is based on PN. The design of a system with merely an informal definition is more difficult. Under such scenarios, the proposed method can aid in this situation. The modeled system can be analyzed by satisfying various PN properties such as reachability, reversibility, liveness, boundedness, deadlock absence, etc. That are performance indicators. The Other Performance indices, such as response time, throughput, dependability, system availability, etc., are also measurable.

The breakdown of the paper's structure is as follows. The following section summaries known techniques as well as their shortcomings. The preliminary concepts and definitions are covered in Section III. Section IV discusses the case study of the Shutdown System (SDS) and its PN model. Section V presents the proposed methodology for performance measurement. The validation of proposed methodology is discussed in Section VI. The paper comes to a close with the conclusion and future work in Section VII.

* Corresponding author.
E-mail addresses: nkjyotish.18dr0087@cse.iitism.ac.in (N.K. Jyotish), lalit.rs.cse@iitbhu.ac.in (L.K. Singh), chiranjeev@iitism.ac.in (C. Kumar).

https://doi.org/10.1016/j.net.2022.11.015

2. Related work

Singh et al. [4] used PN to present a framework for modeling and prediction of the performability of SCS. A SCS of NPP is used to demonstrate the technique. However, because the methodology depends solely on the TimeNET tool for calculation, it cannot properly consider the component interfaces. Further, the paper does not discuss any method to measure response time of the systems.

Su et al. [5] proposed a unified model-driven design framework to facilitate embedded control software development. Authors claim that the method can support the unification of models built by different modeling tools for high-efficiency simulation and high-quality code generation. However, no measurement method is explained to assess the performance of the code execution.

Singh and Rajput [6] employed PN to analyze the dependability of an SCS's shutdown system SDS. The suggested method takes advantage of the PNs modeling power by turning it to a Markov Chain for quantification. However, this work does not evaluate SCS performance. The authors in Refs. [7,8] surveyed various approaches for evaluating the performance of PN-based models of SCS, their limitations, tools utilized, and the performance measures employed.

Guo et al. [9] investigated state-of-the-art convolutional neural network (CNN) models and CNN-based applications. The method described a complete design flow for mapping CNN onto embedded field-programmable gate arrays (FPGA). However, the authors did not give any assurance that the mapped FPGA will meet the performance requirements.

Singh and Singh [10] proposed a method to measure the dependent failures of the components in a system, known as common cause failures. Although the authors consider the dependencies for risk and reliability measurement, however, such dependencies need to be considered for performance measurement of the system.

Rodríguez [11] transformed Unified Modeling Language profiles into Petri nets for software performance analysis based on the upper productive capacity. The authors employed the PeabraiN tool to determine the maximum through-put bounds using the iterative LPP algorithm. However, there is no method mentioned for assessing the performance.

Kumar et al. [12] measured SCS performance by using the timed Petri nets (TPN), and Markov Chain. The technique was tested on an NPP's shutdown system SDS. This method fails to capture dynamic interactions among components, thus making performance measurement accuracy questionable.

Xia et al. [13] evaluated the performance of Canada Deuterium Uranium (CANDU) reactor shutdown system SDS-1 using MATLAB/Simulink, signal processing system, and existing power management. The proposed methodology significantly improves trip response time in comparison to the present system. However, the validity of considering all the functional requirements is not ensured.

Rhee et al. [14] developed a 3-D Computational Fluid Dynamics (CFD) model to analyze the performance of the SDS CANDU reactor's liquid poison injection process. The authors conducted a series of studies to construct a restricted validation CFD model. However, the experiment does not provide a detailed dispersion profile of the liquid poison's distribution in the moderator CANDU reactor.

By leveraging observations of safety parameters in an NPP's SDS, Rankin and Jiang [15] suggested a Kalman filter-based technique for developing predictive SDS and predicting the attainment of trip set-points. When compared to traditional SDSs, the given prognostic SDS significantly brings down time-to-trip. However, the method is just a preliminary step toward developing a potentially beneficial plan to improve the performance of ceremonious SDS.

Chen and Li [16] use sparse autoencoder and artificial neural network for multisensory feature fusion to perform the fault diagnosis of the bearing and also to improve the reliability of fault diagnosis. However, performance is also a health indicator, which is not considered in this work.

3. Preliminary concepts and definitions

A Petri net is a directed, weighted, and bipartite graph containing two different types of nodes: places (shown by circles) and transitions (depicted by bars or boxes). The Positive weight-labeled directed arcs connect these places and transitions. Any of the places may hold any number of tokens. The black dots inside the places denote tokens held by that respective place. Formally, a PN is described as a 5-tuple $\Psi = \{P, T, \alpha, \beta, M_0\}$, where $P = \{p_1, p_2, p_3, ..., p_m\}$ is a non-empty finite set of places which describe the state of a system, $T = \{t_1, t_2, ..., t_n\}$ is a non-empty finite set of transitions which help in changing the state of the system, α: $(P \times T) \rightarrow N$ is the pre-incidence function defining directed arcs from place to transition, β: $(T \times P) \rightarrow N$ is the post-incidence function defining directed arcs from transition to place. Here, N refers set of natural numbers. $M_0 \ : P \rightarrow \{0, 1, 2, . . .\}$ is the initial marking, i.e., a m-vector whose element representing the token present in each of the m places of net. Also, $P \cap T = \phi$and $P \cup T \neq \phi$ [1].

The PN model's token movement delineates the system's dynamic behavior, represented by a change in token distribution among the places. The necessary condition to change the token distribution is that at least one model transition must be in the enabled state. When every input place(s) p of transition t is having a minimum number of tokens equal to the weight of the arc (p, t), then the transition t is said to be in the enabled state. The enabled transition can fire. When transition t fires, it takes tokens from each of their input place(s) p based on the weight of the arc(s) (p, t) and adds them in their every output places.

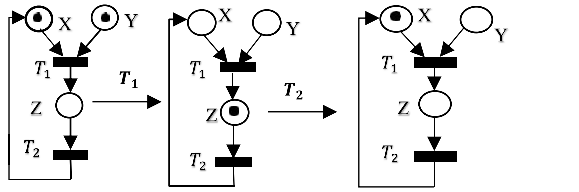
Fig. 1 depicts the working of a PN-modeled system in which X, Y, and Z represent the places, whereas T_1 and T_2 are the model's transitions.

The performance of the systems depends on its reliability and safety. Therefore, while assessing performance, we must analyze factors that might endanger SCS's reliability and safety. These factors include Deadlock, Boundedness, Liveness, Reachability, Stability, & Reversibility [1,2,6,17].

As long as a marked trap exist in the siphon, there's no danger of potential deadlock in any siphon and therefore PN is deadlock-free and live. The bounded property assures the absence of overflow at any place of the PN. The token count at any place of a bounded PN never surpasses a finite integer l for any marking reachable from initial marking and PN is safe in all cases for $l = 1$. If the boundedness property is satisfied for every possible firing sequence, then PN becomes stable, and it is called steady if the following conditions are met [1,7].

$$(\Delta M(t)) \, / \, \Delta t = 0, \text{where} \Delta t = t - t_0 \tag{1}$$

Reachability is the key criteria for studying the dynamic aspects of system. A firing sequence in a Petri net leads to a marking sequence. If a sequence of firing transforms marking M_1 to another



a. Initial Marking b. Marking after T_1 fires c. Marking after T_2 fires

Fig. 1. Petri net execution.

marking M_n, then M_n is said to be reachable from M_1. A reversible PN always allow returning to the initial marking or a home state [1]. These properties must exist for the Petri net-based modeled system. The PN model's steady-state probability distribution is computed after creating an equivalent Markov Chain from its reachability graph and solving the following linear system.

$$\begin{cases} \Pi \times Q = 0 \\ \sum_{i=0}^{n} \pi_i = 1 \end{cases} \tag{2}$$

and

$$q_{ij} = -\sum_{j=1}^{j=n, j\neq i} q_{ij} \tag{3}$$

where, $\Pi = (\pi_1, \pi_2, \pi_3, \bullet\bullet\bullet, \pi_n)$ is the steady-state probability and π_i denotes the probability of being in state S_i. $Q = [q_{ij}]$ is the transition rate matrix such that $(i \neq j)$ and q_{ij} indicates transition rate for S_i to S_j [6]. For no transition, $q_{ij} = 0$.

A process net $PRN = (P \cup \{p_s, p_e\}, T, F, M_0)$ is a consistent, strongly connected, conservative, and live Petri net. Where, P & T denote set of internal places, and transitions respectively, p_s is a start place with $| {}^\circ p_s | = 0$, and $|p_s {}^\circ| = 1$; p_e is an end place with $| {}^\circ p_e | = 1$, and $|p_e {}^\circ| = 0$; $F(P \times T)\cup(T \times P)$ denotes a collection of arcs connecting places and transitions. Here, ${}^\circ p_s$ and $p_s {}^\circ$ are a set of input and output transitions of p_s. Similarly, ${}^\circ p_e$ and $p_e {}^\circ$ can be defined. A process net becomes closed process net if $p_s = p_e$. The term "strongly connectedness" refers to the fact that when p_s is removed, the resulting net becomes acyclic. It means that there is a directed path between any pair of nodes of the net. Consistency is described as a presence of firing sequence from M_0 to M_0 such that each transition fires at least once. The closed process net has strong reversibility properties, which means we can always return to M_0 from any other marking $M \in \mathrm{R}(M_0)$ upon firing of transitions [18,19].

Continuous Petri net (CPN) is a relaxation strategy, to prevent exponentially growing reachable marking resulting from increased PN size. Formally, it is defined as a 3-tuple $CPN = \{PN^M, M_0, R\}$, where $\{PN^M, M_0\}$ is a marked message passing (MP) net [19]. *CPN* consists of a set of closed process nets, along with various synchronous and asynchronous mechanisms. In Synchronous message passing mechanism, the sending action wait until it receives an acknowledgment, whereas, asynchronous message passing scheme allow sender can transmit information without waiting for acknowledgment and without waiting for message to reach at destined place. The PN representation of these mechanisms are shown in Fig. 2. Fig. 2(a) represents the asynchronous message passing scheme, and Fig. 2(b) is used for synchronous message passing mechanism. $R: T \rightarrow (0, +\infty), R\ (t_i) = r_i\ (\mathrm{i} = 1,2, \ldots, \mathrm{m})$ is a function which assigns a firing rate r_i to t_i. If all the input places of a CPN transition have nonzero markings, then that transition is said to be enabled.

Let p_k^1 and p_k^2 are the input places of transition t_i with their respective markings m_k^1 and m_k^2. Suppose the transition t_i fires at time τ during a period $\Delta\tau$, then

$$\forall p_k \in t_i : m_k(\tau + \Delta\tau) = m_k(\tau) - v_i(\tau)\Delta\tau. \tag{4}$$

$$\forall p_k \in t_i^\circ : m_k(\tau + \Delta\tau) = m_k(\tau) + v_i(\tau)\Delta\tau. \tag{5}$$

where, v_i is the instantaneous firing speed of transition t_i and equals the maximum firing speed given by $v_i = r_i \times \min\{m_k^1, m_k^2\}$ [19]. Based on these semantics, a collection of ODEs of a model are developed.

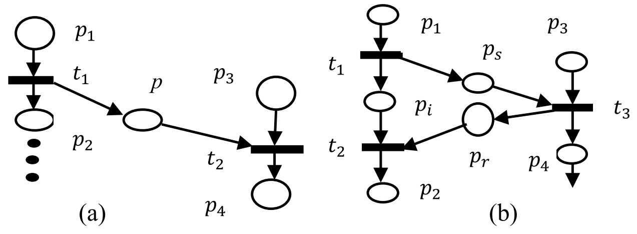


Fig. 2. Petri net representation for (a) asynchronous and (b) synchronous message passing mechanism.

4. Case study: SDS and its PN model

4.1. SDS

To achieve shutdown criteria of NPP, certain essential factors known as trip parameters must be monitored at all times. There are two types of trip parameters: absolute and conditional. The use of absolute trip parameters is correct at any power level of the reactor, while the conditional parameters are valid only where they are equal to or higher than 2% of the full power of the reactor. SDS triggers in auto mode when any of parameters deviated from its normal range [6]. Fig. 3 is the simplified schematic diagram of SDS.

SDS is a CBS type that incorporates various components, such as sensors, actuators, digital I/O cards, relay output modules, software for data processing, graphical user interface, etc. The liquid poison is injected into the calandria via a 2-out-of-3 trip circuit employing control valves. There are 4 fast acting valves (FAVs) between the helium tank and helium header that serve the poison tanks. Because they operate on the principle of air closure and spring opening mechanism, the FAVs ensure that they open reliably and on-demand.

The valves used are in air-to-close style, which means that if instrument air is lost, the valves will fail open, resulting in a reactor shutdown. The poison tanks, cylindrical in shape, are fixed to the reactor vault's exterior fence [6]. The nozzle connects every poison tank, allowing the poison to be pumped into the moderator. Every poison tank contains a ball that floats on top of the poison. When the injection begins, helium pressure delivers poison into the calandria, and the poison ball is driven into the lower seat of the poison tank. The ball takes position at the poison tank exit in the bottom, preventing helium gas from over-pressurizing the calandria.

4.2. PN model of SDS

The SDS's failure will result in exponential increase in the power and the deign parameters will exceed its range that may jeopardize the integrity of mechanical components by which the radioactivity may get exposed to the public. The SDS incorporates many components, including sensors, logic, actuators, and a specific human-machine interface for achieving its objective. Each FAV line has two vent valves; both are typically open to relieve pressure in that line, if any, and prevent an erroneous poison injection. Fig. 4 shows Petri net model of SDS and is explained as follows. Tables 1a and 1b, respectively, contain descriptions about transitions and places of Fig. 4.

A token is in place m_1 represents that one or more of the trip parameters is/are deviated from their design limits. A token is in m_2 represents the creation of logic condition, a token in m_4 represents the hold state of logic condition. A relay is energized to close the vent valves, which is represented by a token in m_5. The poison is injected into the moderator when the FAV is opened, which is represented by a token in place m_{10}. For improved reliability, a duplicate information about FAV state, from redundant sensor is

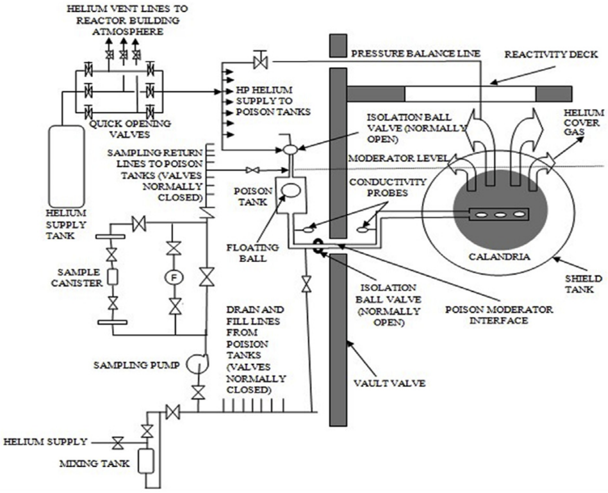

HELIUM VENT LINES TO REACTOR BUILDING ATMOSPHERE
PRESSURE BALANCE LINE
REACTIVITY DECK
HP HELIUM SUPPLY TO POISON TANKS
QUICK OPENING VALVES
ISOLATION BALL VALVE (NORMALLY OPEN)
HELIUM COVER GAS
SAMPLING RETURN LINES TO POISON TANKS (VALVES NORMALLY CLOSED)
MODERATOR LEVEL
CONDUCTIVITY PROBES
POISON TANK
HELIUM SUPPLY TANK
FLOATING BALL
CALANDRIA
SAMPLE CANISTER
F
SHIELD TANK
ISOLATION BALL VALVE (NORMALLY OPEN)
POISON MODERATOR INTERFACE
DRAIN AND FILL LINES FROM POISION TANKS (VALVES NORMALLY CLOSED)
SAMPLING PUMP
VAULT VALVE
HELIUM SUPPLY
MIXING TANK

Fig. 3. Simplified diagram of SDS Liquid Poison Injection System.

monitored, which is represented by a token in m_8 place. The place m_{13} is included to prioritize t_6 over t_2 when they are race conditions. This will ensure the opening of FAV; in case any security threat reports false information (closed state) about the FAV state. The place m_{14} has been used to show strong reversibility properties of closed process nets.

As shown in Fig. 4., our model consists of following two closed process nets: (a) First set of Closed process net is $\{m_1, m_2, m_3, m_4\}$, and (b) second set of closed process net is made up of $\{m_5, m_6, m_{13}, m_3, m_{11}, m_7, m_8, m_{12}, m_9, m_{10}, m_{14}, m_1\}$. These two closed process nets communicate with each other via asynchronous message passing mechanism.

5. Proposed methodology for performance measurement

Apart from the evaluation of performance measure i.e., mean latency time, it is essential to verify some of the other metrics as well, which are the indicators of performance. For example, deadlock may lead to delay in process execution or even hold the state of system for infinite time. Liveness refers a set of properties that require a system to make progress even though its concurrently executing components. The system must also be stable, steady and must be bounded. The computation of these metrics is shown below.

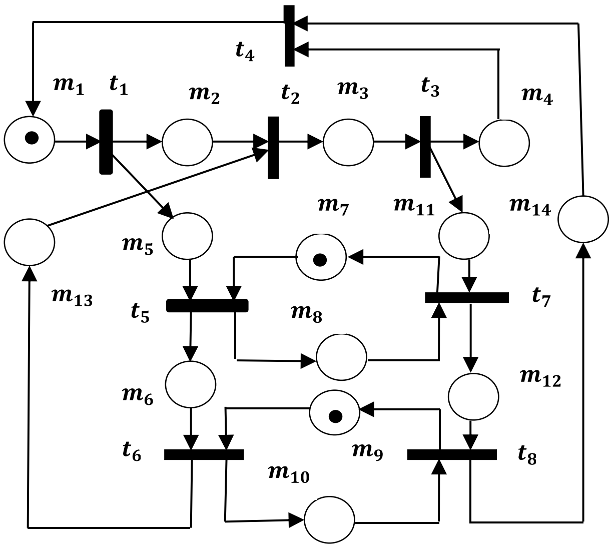


Fig. 4. Petri net model of Poison Injection System of SDS.

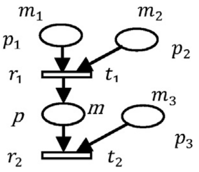


Fig. 5. Two places to Two places model.

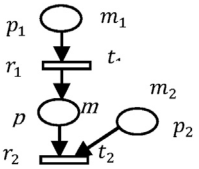


Fig. 6. One place to two places model.

1) Deadlock and Liveness Analysis:

The modeling of SDS was carried out using a TPN, as shown in Fig. 4. We explained deadlock and liveness analysis using siphons and traps in Section III. We run the TimeNET tool to calculate the number of siphons and traps present in the SDS PN model. It has 12 minimal siphons and 12 marked traps. The siphons are: $S_1 = \{m_6, m_{13}, m_3, m_{11}, m_7\}, S_2 = \{m_3, m_{11}, m_{12}, m_{14}, m_1, m_5, m_6, m_{13}\}, S_3 = \{m_1, m_5, m_6, m_{13}, m_3, m_4\}, S_4 = \{m_1, m_2, m_3, m_4\}, S_5 = \{m_1, m_2, m_3, m_{11}, m_{12}, m_{14}\}, S_6 = \{m_3, m_{11}, m_{12}, m_9, m_{13}\}, S_7 = \{m_3, m_4, m_1, m_5, m_8, m_{12}, m_9, m_{13}\}, S_8 = \{m_1, m_2, m_3, m_{11}, m_7, m_6, m_{10}, m_{14}\}, S_9 = \{m_7, m_8\}, S_{10} = \{m_5, m_8, m_{12}, m_{14}, m_1\}, S_{11} = \{m_9, m_{10}\}, S_{12} = \{m_5, m_6, m_{10}, m_{14}, m_1\}$.

The traps are: $T_1 = \{m_1, m_5, m_8, m_{12}, m_{14}\}, T_2 = \{m_1, m_2, m_3, m_4\}, T_3 = \{m_1, m_5, m_6, m_{13}, m_3, m_{11}, m_{12}, m_{14}\}, T_4 = \{m_1, m_2, m_3, m_{11}, m_{12}, m_{14}\}, T_5 = \{m_3, m_{11}, m_7, m_6, m_{13}\}, T_6 = \{m_3, m_{11}, m_{12}, m_9, m_{13}\}, T_7 = \{m_1, m_5, m_8, m_{12}, m_9, m_{13}, m_3, m_4\}, T_8 = \{m_7, m_8\}, T_9 = \{m\ 1, m_5, m_6, m_{10}, m_{14}\}, T_{10} = \{m_9, m_{10}\}, T_{11} = \{m_3, m_4, m_1, m_5, m_6, m_{13}\}$, and $T_{12} = \{m_1, m_2, m_3, m_{11}, m_7, m_6, m_{10}, m_{14}\}$. We can observe that S_1, S_2, S_3, S_4, S_5, S_6, S_7, S_9, S_{10}, S_{11}, and S_{12} are also marked trap but S_8 do not contain any trap. It means our PN model is deadlock-free. Also, the Petri net satisfies the liveness criteria as mentioned in Section III since it has no potential deadlock.

2) Stability, Boundedness and Steady State Analysis:

From SDS PN-model, shown in Fig. 4, each place contains either zero or one token for each marking, which is reachable from the initial marking M_0, i.e., $M_0 \leq 1$. It concludes that the system is stable. Additionally, because the model is one-bounded, it indicates that it is safe. We can also see that $\Delta M / \Delta t = 0$. As a result of equation (1), the system is steady also. Thus, from the analysis of Petri net model of SDS, it satisfies all of the performance metrics.

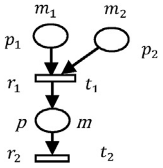


Fig. 7. Two places to one place model.

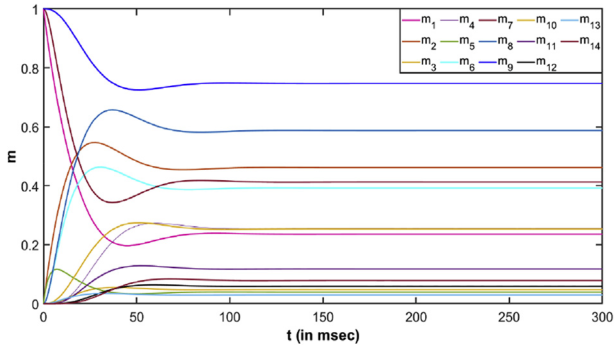


Fig. 8. The solutions (state measures) of the Petri nets model of SDS.

Table 1a
SDS-2 process transitions.

Transitions	Description
t_1	Sends signal for creating LC and to energize the relays for closing the vent valves
t_2	Trigger signal for holding LC at the created state
t_3	Trigger signal for restoring LC and de-energize
t_4	Resend signal for opening the FAV if it fails to open
t_5	Trigger for closing the vent valves
t_6	Trigger for opening all FAV
t_7	Trigger for opening the vent valves
t_8	Trigger for closing all FAV

3) Performance Analysis:

The success criteria of SDS is that it should be able to inject poison into the nuclear core within 1 s [6]. The proposed framework for performance analysis consists of three steps, explained as follows.

Step 3.1: Formulation of Ordinary Differential Equation System: A collection of ODEs of a PN model are developed based on equations (4) and (5), and semantics, discussed in Section III. These ODE help in computing the marking. Let m_i and m are the markings of places p_i and p respectively and r_i denotes the firing rate of transition t_i. We consider the following cases in the formulation of ODE system:

Case A: Two *places to two places model:* As Fig. 5 shows, place p is getting markings from both the place p_1 and place p_2, while it sends some marking along with place p_3. That is, p_1 and p_2 are the input places for t_1, while p and p_3 are the input places for t_2.

Table 1b
SDS process places.

Places	Description
m_1	Deviation of trip parameters
m_2	Logic Conditions (LC) gets created
m_3	Hold LC
m_4	Restore LC
m_5	Relay energized to close the vent valves
m_6	Vent valves get closed
m_7	Redundant information of FAV in the closing state
m_8	Redundant information of FAV in the opening state
m_9	FAV close
m_{10}	FAV open
m_{11}	Relay de-energized to open the vent valves
m_{12}	Open the vent valves
m_{13}	Ensures the precedence of t_6 over t_2
m_{14}	Ensures reversibility properties

If each transition fires, then the marking m for a time increment $\Delta\tau$ is written as,

$$m(\tau+\Delta\tau)=m(\tau)+r_1\cdot\min\{m_1(\tau),m_2(\tau)\}\Delta\tau$$

$$-r_2\cdot\min\{m(\tau),m_3(\tau)\}\Delta\tau$$

$$\Rightarrow\frac{m(\tau+\Delta\tau)-m(\tau)}{\Delta\tau}=r_1\cdot\min\left\{m_1(\tau),m_2(\tau)\right\}-r_2\min$$

$$\left\{m(\tau),m_3(\tau)\right\}$$

Let $\Delta\tau\rightarrow 0$, then we get the following ODE

$$m'(\tau)=r_1.\min\{m_1(\tau),m_2(\tau)\}-r_2.\min\{m(\tau),m_3(\tau)\} \tag{6}$$

Case B: *One place to two places model:* As Fig. 6 shows, place p gets marking from p_1, and it produces some marking with the help of p_2. If every transition fires, then for a time interval $\Delta\tau$, the marking m can be represented as,

$$m(\tau+\Delta\tau)=m(\tau)+r_1m_1(\tau)\Delta\tau-r_2\cdot\min\{m(\tau),m_2(\tau)\}\Delta\tau$$

$$\Rightarrow\frac{m(\tau+\Delta\tau)-m(\tau)}{\Delta\tau}=r_1m_1(\tau)-r_2.\min\{m(\tau),m_2(\tau)\}$$

Let $\Delta\tau\rightarrow 0$, then we obtain a below ODE

$$m'(\tau)=r_1m_1(\tau)-r_2.\min\{m(\tau),m_2(\tau)\} \tag{7}$$

Case C: *Two places to one place model:* As Fig. 7 shows, place p obtains marking from p_1 and p_2, and it produces a marking for another place. Then, we derive the below differential equation:

$$m'(\tau)=r_1.\min\{m_1(\tau),m_2(\tau)\}-r_2.m(\tau). \tag{8}$$

Assume that it is possible to achieve the firing constants r_i in advance for every activity modeled by a transition. The ODEs system of the PN model given in Fig. 4 is shown in (9).

$$\begin{aligned}
m_1' &= r_4\min\{m_4,m_{14}\}-r_1m_1\\
m_2' &= r_1m_1-r_2\min\{m_2,m_{13}\}\\
m_3' &= r_2\min\{m_2,m_{13}\}-r_3m_3\\
m_4' &= r_3m_3-r_4\min\{m_4,m_{14}\}\\
m_5' &= r_1m_1-r_5\min\{m_5,m_7\}\\
m_6' &= r_5\min\{m_5,m_7\}-r_6\min\{m_6,m_9\}\\
m_7' &= r_7\min\{m_8,m_{11}\}-r_5\min\{m_5,m_7\}\\
m_8' &= r_5\min\{m_5,m_7\}-r_7\min\{m_8,m_{11}\}\\
m_9' &= r_8\min\{m_{10},m_{12}\}-r_6\min\{m_6,m_9\}\\
m_{10}' &= r_6\min\{m_6,m_9\}-r_8\min\{m_{10},m_{12}\}\\
m_{11}' &= r_3m_3-r_7\min\{m_8,m_{11}\}
\end{aligned} \tag{9}$$

$$m'_{12} = r_7 \min\{m_8, m_{11}\} - r_8 \min\{m_{10}, m_{12}\}$$

$$m'_{13} = r_6 \min\{m_6, m_9\} - r_2 \min\{m_2, m_{13}\}$$

$$m'_{14} = r_8 \min\{m_{10}, m_{12}\} - r_4 \min\{m_4, m_{14}\}$$

The initial values for the ODE system are: $m_1(0) = m_7(0) = m_9(0) = 1$, and all others are 0. Where, m_i is marking of the respective place and r_i is the firing rate assigned to t_i.

Step 3.2: Solution of ODEs Using MATLAB: The above ODEs system can be solved using MATLAB function “ode45”. The ode45 function is a fourth or fifth order of Runge-Kutta method. For the ODEs system (9) of the SDS Petri net model, with the simulation data, we have $r_1 = 0.05$, $r_2 = 0.40$, $r_3 = 0.25$, $r_4 = 0.15$, $r_5 = 0.30$, $r_6 = 0.03$, $r_7 = 0.10$, and $r_8 = 0.20$. Using this method, we get the results as illustrated in Fig. 8. When t > 133.6481 msec, every result approaches a unique fixed value: m_1 (t) ≈ 0.2355, m_2 (t) ≈ 0.4621, m_3 (t) ≈ 0.0472, m_4 (t) ≈ 0.2552, m_5 (t) ≈ 0.0393, m_6 (t) ≈ 0.3933, m_7 (t) ≈ 0.4321, m_8 (t) ≈ 0.5899, m_9 (t) ≈ 0.7488, m_{10} (t) ≈ 0.2535, m_{11} (t) ≈ 0.1179, m_{12} (t) ≈ 0.0590, m_{13} (t) ≈ 0.0295, and m_{14} (t) ≈ 0.0784. The ODEs solution is used to find the system's delay.

Step 3.3: Evaluation of Performance measures using ODE Solution and Queuing theory: Based on the ODE solution of step 3.2, we can now evaluate the performance measures (delay time) of SCS using queuing theory, as evaluated below.

Mean Latency Time: It is defined as the delay time to inject the poison into calandria of the SDS. i.e., for the closed process net based system, it is the delay time spent in a process, from the start of SDS until the finish when the poison is completely injected into the system. The mean latency of a subsystem is computed while the system is present at the steady-state. Based on the queueing theory and *Little's law*, the mean latency time can be computed as:

$$W = L/\lambda \tag{10}$$

where L indicates the average token count present in the system, λ is the mean token arrival rate in the system, and W represents the mean latency time of subsystem. Because, the ODE solutions indicate the average marking of each place while the system is in steady-state, therefore L can be calculated as, $L = \sum_{l \in M} m_l$. Where M represents the set of places that model either other component of SDS waiting for the token so that they can perform their task or their token request in the process. Thus, in steady state, the mean delay time is defined as the task's queue length divided by the mean number of markings entering the subsystem in unit time.

In our PN model as shown in Fig. 4, after the initiation of poison injection process at the place m_1, all the remaining places from other closed process nets $\{m_5, m_6, m_{13,} m_3, m_{11}, m_7, m_8, m_{12}, m_9, m_{10}, m_{14,} m_1\}$ are waiting for tokens so that they can perform their intended task. Therefore,

$$\begin{aligned} L = {} & m_5 + m_6 + m_{13} + m_3 + m_{11} + m_7 + m_8 + m_{12} + m_9 \\ & + m_{10} + m_{14} + m_1 \\ & = 3.0244 \end{aligned} \tag{11}$$

The mean token arrival rate λ is computed as,

$$\lambda = r_1 m_1 = (0.05 \times 0.2355) = 0.011772 \tag{12}$$

Thus, mean delay time using ODE solution is

$$W_{ODE} = L/\lambda = \frac{3.0244}{0.011772} = 256.91\ msec \tag{13}$$

From Fig. 9, we find that when t > 133.6486 msec, the mean latency time approaches a fixed value i.e., 256.91 msec. Hence, the average delay of the SDS system is computed as 0.25691 s.

5.1. Complexity analysis of performance measurement

The complexity analysis lies in the solution of ODEs. In the framework of performance analysis, we employed *Runge–Kutta* method to solve a family of ODEs. This method is better than Newton's method if the accuracy is less than 0.000001. We know that the Newton's method has complexity *O(m n^3)*. Here, *m* indicates number of iterations, whereas *n* is the number of variables. *m* is generally *O(n)* and never exceeds *O(n^2)*. As a result, the *Runge–Kutta* method's complexity is around *O(n^4)* and never surpasses *O(n^5)*. Thus, computing the state measures of an ODE model requires a maximum of *O(n^5)*, where *n* denotes number of equations and $n \le |P|$. In our model, $n = 14$ i.e., system have 14 places (as shown in Figs. 4), and 14 ordinary differential equations (as shown in equation (9)). The polynomial time complexity *O(n^5)* of the state measure of ODE model demonstrates that the proposed strategy is capable of avoiding the state explosion problem, which is generally experienced by the traditional Markov chain based approaches.

6. Validation of proposed methodology

An effective method for performance assessment is proposed recently, by Kumar et al. [12]. The authors claim that the proposed method is very effective and gives the performance estimates with an accuracy of more than 99% and demonstrated the approach on a case study of NPP. To prove the effectiveness of our proposed approach, we carried out two steps: (i) we compute the performance of our case study using the recent method proposed in Ref. [12] and compare the results with the real data to find the accuracy of this method. (ii) we compute the performance using our proposed ODE method and compare the results with the real data to find the accuracy of our ODE method. Thereafter, both accuracies are compared to find the method that gives higher accuracy.

6.1. Performance validation with [12]: it involves following seven steps

a) *Step 1:* Petri net model creation

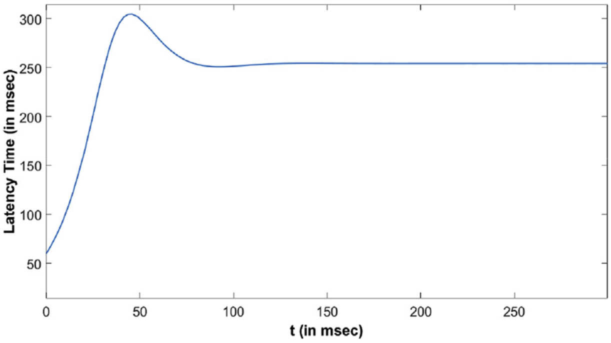


Fig. 9. The mean latency time of the poison injection process of SDS.

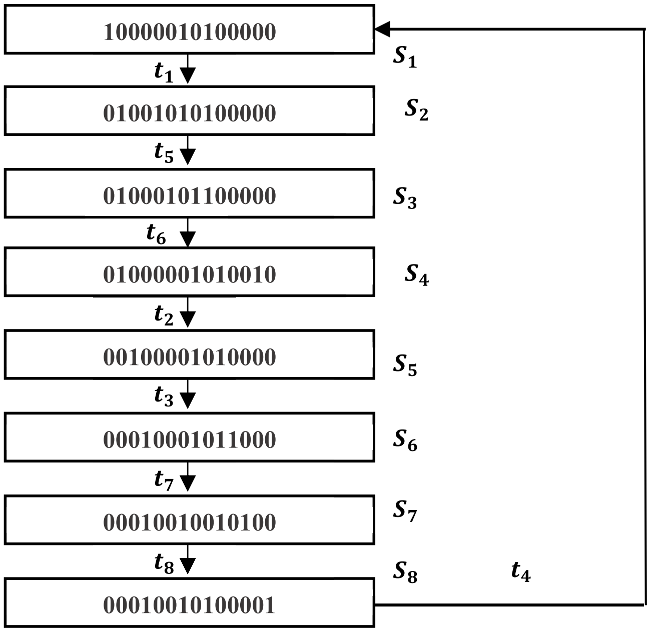


Fig. 10. Reachability Graph for the Petri net model of SDS.

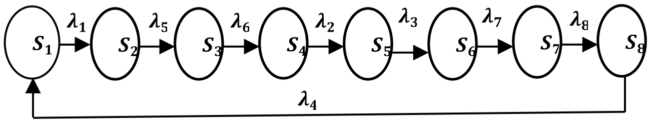


Fig. 11. Markov chain for the Petri net model of SDS.

We create Petri net model of SDS using the TimeNET tool, as shown in Fig. 4.

b) *Step 2:* Model Parameter Assignment

In this step, the delay of each transition is input into the model as per specification, expert's elicitation, and experiences from similar projects. The model was run using TimeNET tool to measure the transition firing rates as indicated in Table 2. λ_i denotes the firing rate of transition t_i (where $i = 1, 2, \ldots, 8$).

The reachability graph determines the system's boundary conditions, which may indicate the number of possible states during the system's operational life. The total number of possible markings shows the entire number of states that a system can go through. The total eight states are possible for the SDS PN model, as specified by (14). From the PN-model depicted in Fig. 4, the corresponding reachability graph is constructed [1,6] and presented in Fig. 10.

d) *Step 4:* Markov Chain Creation

The reachability graph of the PN model is used to generate the Markov chain [1,6]. Fig. 11 illustrates the Markov chain for a TPN model of Fig. 4.

e) *Step 5:* Steady-State Marking Probability Calculation

Equations (2) and (3) can be used to calculate the steady-state marking probabilities. The transition rate matrix Q is shown in (14). The resulting equation is shown in (15). The steady-state marking probabilities are calculated using (15) and the transition's firing rate values of Table 2. These values are shown in Table 3.

$$Q = \left[\begin{array}{ccccccccc} & S_1 & S_2 & S_3 & S_4 & S_5 & S_6 & S_7 & S_8 \\ S_1 & q_{11} & \lambda_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ S_2 & 0 & q_{22} & \lambda_5 & 0 & 0 & 0 & 0 & 0 \\ S_3 & 0 & 0 & q_{33} & \lambda_6 & 0 & 0 & 0 & 0 \\ S_4 & 0 & 0 & 0 & q_{44} & \lambda_2 & 0 & 0 & 0 \\ S_5 & 0 & 0 & 0 & 0 & q_{55} & \lambda_3 & 0 & 0 \\ S_6 & 0 & 0 & 0 & 0 & 0 & q_{66} & \lambda_7 & 0 \\ S_7 & 0 & 0 & 0 & 0 & 0 & 0 & q_{77} & \lambda_8 \\ S_8 & \lambda_4 & 0 & 0 & 0 & 0 & 0 & 0 & q_{88} \end{array}\right] \quad (14)$$

$$\begin{aligned}\pi_1 \bullet \lambda_1 &= \pi_8 \bullet \lambda_4, \pi_2 \bullet \lambda_5 = \pi_1 \bullet \lambda_1, \pi_3 \bullet \lambda_6 = \pi_2 \bullet \lambda_5, \pi_4 \bullet \lambda_2 \\ &= \pi_3 \bullet \lambda_6, \pi_5 \bullet \lambda_3 = \pi_4 \bullet \lambda_2, \pi_6 \bullet \lambda_7 = \pi_5 \bullet \lambda_3, \pi_7 \bullet \lambda_8 \\ &= \pi_6 \bullet \lambda_7, \pi_8 \bullet \lambda_4 = \pi_7 \bullet \lambda_8\end{aligned} \quad (15)$$

f) Step 6:

Steady-State Token Probability Density Function Calculation: It calculates the likelihood of a specific amounts of token being present at a particular place in the steady-state. These values are shown in Table 4 for the presence of a single token at each place.

g) Step 7:

Use Queuing Theory for the Delay Measurement.

The mean latency of a subsystem while the system is present at the steady-state is computed using *Little's law.* It is defined as: D = Mean tokens arrival rate in the system, S = Mean latency of subsystem, and V = mean token count.

Then, using *Little's law*, $V = DS$ (16)

The value of V is obtained after summing of all the steady-state probability density values obtained from Table 4.

$$\therefore V = 3.9705 \quad (17)$$

Initially, there is one token present in the positions m_1, m_7, and m_9. Therefore, the mean token arrival rate can be found by multiplying the values of these places' steady-state probability density to their respective transition rates, and then they are added. *i.e.,*

$$D = (P(m_1) \bullet \lambda_1) + (P(m_7) \bullet \lambda_5) + (P(m_9) \bullet \lambda_6)$$

$$\therefore D = 0.015592 \quad (18)$$

$$\therefore S = \frac{V}{D} = \frac{3.9705}{0.015592} = 254.6498 \quad (19)$$

So, mean delay time by approach of [12] is

$$W_{[12]} = 254.65\ msec \quad (20)$$

Table 2
Transition's firing rate (in per msec).
c) *Step 3:* Reachability Graph Creation.

λ_1	λ_2	λ_3	λ_4
0.01256	0.820	0.532	0.650
λ_5	λ_6	λ_7	λ_8
0.148	0.020	0.0185	0.150

Table 3
Steady-state marking probabilities.

π_1	π_2	π_3	π_4
$1.4988\times e^{-15}$	$1.7347\times e^{-18}$	$2.2204\times e^{-16}$	0.0191
π_5	π_6	π_7	π_8
0.0295	0.8470	0.1045	$2.8961\times e^{-15}$

Table 4
Steady-state token probability density values.

$P(1\ Token\ in\ m_1) = P(m_1) = \pi_1$
$P(1\ Token\ in\ m_2) = P(m_2) = \pi_2 + \pi_3 + \pi_4$
$P(1\ Token\ in\ m_3) = P(m_3) = \pi_5$
$P(1\ Token\ in\ m_4) = P(m_4) = \pi_6 + \pi_7 + \pi_8$
$P(1\ Token\ in\ m_5) = P(m_5) = \pi_2$
$P(1\ Token\ in\ m_6) = P(m_6) = \pi_3$
$P(1\ Token\ in\ m_7) = P(m_7) = \pi_1 + \pi_2 + \pi_7 + \pi_8$
$P(1\ Token\ in\ m_8) = P(m_8) = \pi_3 + \pi_4 + \pi_5 + \pi_6$
$P(1\ Token\ in\ m_9) = P(m_9) = \pi_1 + \pi_2 + \pi_3$
$P(1\ Token\ in\ m_{10}) = P(m_{10}) = \pi_4 + \pi_5 + \pi_6 + \pi_7$
$P(1\ Token\ in\ m_{11}) = P(m_{11}) = \pi_6$
$P(1\ Token\ in\ m_{12}) = P(m_{12}) = \pi_7$
$P(1\ Token\ in\ m_{13}) = P(m_{13}) = \pi_4$
$P(1\ Token\ in\ m_{14}) = P(m_{14}) = \pi_8$

Table 5
Firing rate in communication network of SDS.

Transition	Throughput Rate (firing/sec)
Send, Send Ack	9.375
Msg Drop, Ack Drop CRC Ok, Ack Ok	3.91
	74.22
Timeout	1.000

It means that, on average, a single token is in use for about 254.65 msec of time in the system. Therefore, the modeled SDS Petri net injects the poison to trip the reactor in 0.25465 s in an emergency event. It depicts the SDS system's average delay.

6.2. Performance validation on real-time data

An SDS system is expected to inject poison into the calandria of the nuclear reactor if any of the trip parameters deviate from their intended values. As soon as the token is deposited in m_1, the poison injection procedure begins in accordance with the Petri net model, as shown in Fig. 4. However, prior to the poison injection process, adequate communication occurs between the various components of the SDS. The communication between transitions requires reading a message, sending a message, and sending/receiving acknowledgment, each of which has an exponentially distributed execution time. If a sent message is lost in transit, or the sender does not receive an acknowledgment within a time limit then there is a need to send the message again. The cyclic redundancy check computation is also performed during communication. The trip values conveyed to the SDS system are denoted by a token in the place m_1 having a poison rate of μ. Thus, the SDS system's actual throughput is μ(1 - ρ). Here, ρ denotes the probability that there is no token in place m_1 implies that the subsystem is too busy to take new messages.

In our scenario, the SDS communication network's baud rate is 9600 with a 5% error rate and a packet size of 128B. Then, we conduct a performance analysis of our system using the transition firing rates given in Table 5. The mean latency time can be calculated when the system is congested or there is a loss of packet acknowledgment or is on-hold. In this situation, we use Little's law $N = \mu T$, to calculate the latency, where, μ is the throughput rate. Using the values mentioned above and throughput values of Table 5, the mean latency time for the poison injection in the SDS is 0.2572 s.

So, mean delay time using Little's law is

$$W_{LL} = 257.2\ msec \tag{21}$$

Comparing equations (13) and (21), the accuracy of our proposed approach for performance assessment using ODE can be computed by:

$$error\% = \frac{|W_{LL} - W_{ODE}|}{W_{LL}} \times 100\%$$

$$= \frac{|257.2 - 256.91|}{257.2} \times 100 = 0.11275\%$$

$$\therefore Accuracy = (100 - 0.11275)\% = 99.887\% \tag{22}$$

Similarly, the accuracy using approach mentioned in Ref. [12] for the performance assessment can be computed as 99.008%, after comparing (20) and (21). Hence, the accuracy of the performance measurement using our approach is remarkable. The deviation in the accuracy of our approach is less compare to some other approach on a real-time data of NPP. This proves the effectiveness of our approach and hence can be used in all SCS.

7. Conclusions & future work

This research paper aims to measure the performance of the SCS using an ODE and timed Petri net. We introduced some important metrics of performance, which is essential to be verified in case of SCS such as deadlock, stability, steady state, etc. The proposed approach is illustrated on a SCS of NPP. The suggested technique can address the constraints and limits of existing methods, as stated in Section II. The presented methodology involves modeling of SDS using Petri net and then converting the model into a series of ODEs system for the performance measurement. The mechanism explained here calculates the time required for the successful poison injection to trip the NPP by the SDS. The MATLAB simulation result help in the evaluation of the outcome. The obtained average accuracy of our method is 99.887%. It is to be noted that a major issue in developing Petri net model is state space explosion problem when the number of states of a system are more, which may occur in large scale systems. The proposed ODE and PN based solution are capable to deal with this limitation. The proposed technique has not been validated for non-exponential failures, which will be considered in our future work. We intend to expand our work in the future to improve the proposed technique for other classes of concurrent systems. We shall also try to integrate several dependability measures that influence performance of SCS.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Tadao Murata, Petri nets: properties, analysis and applications, Proc. IEEE 77 (1989) 541–580, 4.
[2] Jana Siebert, Dario Petri, Michele Fedrizzi, From measurement to decision:

sensitivity of decision outcome to input and model uncertainties, IEEE Trans. Instrum. Meas. 68 (2018) 3100–3108, 9.

[3] Gaowei Xu, et al., Online fault diagnosis method based on transfer convolutional neural networks, IEEE Trans. Instrum. Meas. 69 (2019) 509–520, 2.

[4] Lalit Kumar Singh, Gopika Vinod, A.K. Tripathi, Design verification of instrumentation and control systems of nuclear power plants, IEEE Trans. Nucl. Sci. 61 (2014) 921–930, 2.

[5] Zhuo Su, et al., MDD: A Unified Model-Driven Design Framework for Embedded Control Software, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 41 (10) (2021) 3252–3265, https://doi.org/10.1109/TCAD.2021.3132564. In this issue.

[6] Lalit Kumar Singh, Hitesh Rajput, Dependability analysis of safety critical real-time systems by using Petri nets, IEEE Trans. Control Syst. Technol. 26 (2017) 415–426, 2.

[7] Nand Kumar Jyotish, Lalit Kumar Singh, Chiranjeev Kumar, A state-of-the-art review on performance measurement petri net models for safety critical systems of NPP, Ann. Nucl. Energy 165 (2022), 108635.

[8] Pooja Singh, Lalit Kumar Singh, Reliability and safety engineering for safety critical systems: an interview study with industry practitioners, IEEE Trans. Reliab. 70 (2021) 643–653, 2.

[9] Kaiyuan Guo, et al., Angel-eye: a complete design flow for mapping CNN onto embedded FPGA, IEEE Trans. Comput. Aided Des. Integrated Circ. Syst. 37 (2017) 35–47, 1.

[10] Pooja Singh, Lalit Kumar Singh, Modeling and measuring common cause failures in measurement of reliability of nuclear power plant systems, IEEE Trans. Instrum. Meas. 70 (2021) 1–8.

[11] Ricardo J. Rodríguez, A Petri net tool for software performance estimation based on upper throughput bounds, Autom. Software Eng. 24 (2017) 73–99, 1.

[12] Pramod Kumar, Lalit Kumar Singh, Chiranjeev Kumar, Performance evaluation of safety-critical systems of nuclear power plant systems, Nucl. Eng. Technol. 52 (2020) 560–567, 3.

[13] Lingzhi Xia, et al., Performance evaluation of a new signal processing system design to improve CANDU SDS1 trip response during large break LOCA events, J. Nucl. Sci. Technol. 53 (2016) 1513–1520, 10.

[14] Bo Wook Rhee, et al., A three-dimensional CFD model for a performance verification of the liquid poison injection system of a CANDU-6 reactor, Nucl. Technol. 159 (2007) 158–166, 2.

[15] Drew J. Rankin, Jiang Jin, Predictive trip detection for nuclear power plants, IEEE Trans. Nucl. Sci. 63 (2016) 2352–2362, 4.

[16] Zhuyun Chen, Weihua Li, Multisensor feature fusion for bearing fault diagnosis using sparse autoencoder and deep belief network, IEEE Trans. Instrum. Meas. 66 (2017) 1693–1702, 7.

[17] Nancy G. Leveson, Janice L. Stolzy, Safety analysis using Petri nets, IEEE Trans. Software Eng. 3 (1987) 386–397.

[18] MuDer Jeng, Xiaolan Xie, MaoYu Peng, Process nets with resources for manufacturing modeling and their analysis, IEEE Trans. Robot. Autom. 18 (2002) 875–889, 6.

[19] Zuohua Ding, Yuan Zhou, MengChu Zhou, A polynomial algorithm to performance analysis of concurrent systems via Petri nets and ordinary differential equations, IEEE Trans. Autom. Sci. Eng. 12 (2013) 295–308, 1.