Hindawi
International Journal of Aerospace Engineering
Volume 2019, Article ID 2124836, 12 pages
https://doi.org/10.1155/2019/2124836

Research Article

Reliability Model of the Fly-By-Wire System Based on Stochastic Petri Net

Zhong Lu,[1] **Zhiwen Zhang**,[1] **Lu Zhuang**,[1] **and Jia Zhou**[2]

[1]*College of Civil Aviation, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China*
[2]*Department of Aircraft Maintenance, China Eastern Airlines Jiangsu Limited, Nanjing 211113, China*

Correspondence should be addressed to Zhong Lu; luzhong@nuaa.edu.cn

Received 21 July 2019; Revised 11 September 2019; Accepted 8 October 2019; Published 12 November 2019

Academic Editor: Hikmat Asadov

The fly-by-wire system plays an important role in modern civil aircraft. As a typical safety-critical system, its reliability will affect the safety of aircraft significantly. In the paper, stochastic Petri nets are applied in the reliability modeling and analysis for the fly-by-wire system to represent its dynamic (time-dependent) failure behaviors. Stochastic Petri net-based reliability models are established for all kinds of architectures including series, parallel, *m*-out-of-*n*, warm standby, cold standby, and load-sharing architectures, which are commonly used in the fly-by-wire system. A Monte Carlo simulation method is proposed for the stochastic Petri net-based reliability models to generate system lifetime samples, and the system reliability parameters can be calculated in terms of the lifetime samples. Finally, a fly-by-wire system is used as a case study to illustrate the application and effectiveness of our proposed approaches. The results show that the error of the reliability value in a flight duration obtained by our Monte Carlo simulation method is less than 1×10^{-4} compared with the analytical equation.

1. Introduction

The flight-control system is a typical safety-critical system whose reliability will affect the safety of aircraft significantly. The failure or malfunction of the flight-control system will lead to an unsafe flight path or structural failure preventing continued safe flight and landing, which are considered as catastrophic top level failure conditions of the aircraft. In the modern transport category airplanes, fly-by-wire systems have been widely used to replace hydromechanical ones. By utilizing the fly-by-wire system, pilots' commands are converted to electronic signals transmitted by wires to flight-control computers, and control commands are calculated by flight-control computers based on control laws to determine the movements of the actuators at each control surface. Therefore, the mechanical circuits consisting of rods, cables, and pulleys are not required anymore, and the weight of the airplane can be reduced.

In order to improve the reliability of the fly-by-wire system, redundancy architectures including parallel, majority, standby, and load-sharing have been widely used in the design of the fly-by-wire system. As there are dynamic or state-dependent behaviors in the standby or load-sharing systems, the failure of the systems depends not only on the combinations of its component failures but also on the occurrence order of the component failures [1, 2]. At present, fault tree analysis (FTA), dependence diagram analysis (DDA), and Markov analysis (MA) are the most widely used tools for reliability modeling and safety analysis of airborne systems. FTA is a deductive failure analysis that focuses on one particular undesired event and provides a method for determining causes of this event. DDA, which is equivalent to reliability block diagram (RBD) in reliability engineering, provides an alternate pictorial representation of combinations of failures for the purpose of probability analysis. In MA, Markov chains are used to represent various system states and relationships among them. The states can be either operational or nonoperational. The transition rate from one state to another is a function of the failure rate or repair rate. The state probabilities are derived by solving a set of differential equations that are derived from the Markov chain [3, 4]. Among all the three methods, FTA and DDA are both static

tools; they cannot capture the state-dependent behavior of system failure mechanisms [5]. Although MA can cope with state-dependent behaviors, it will be faced with the infamous state space explosion problem when the system is large and complex. What is more, the solution of the differential equations for the Markov chains is a cumbersome work and MA can only deal with the system whose components are following exponential distributions [6].

As a tool for discrete event system simulation, Petri nets have been widely applied in reliability engineering since three decades ago. One of the important applications is focusing on system reliability modeling by using Petri nets instead of traditional reliability tools [7–11]. Hura and Atwood [7] presented a method to represent fault trees with Petri nets, which they thought can provide more insight into failure behaviors. Malhotra and Trivedi [8] studied to construct the reliability models by using stochastic Petri nets and stochastic reward nets, and different kinds of repair scenarios are also considered in these models. Liu and Chiou [9] used Petri nets to denote different kinds of logic operations, and a trapezoidal graph method is applied to account for failure scenarios. Schneeweiss [10] developed the Petri net models for many reliability scenarios, and maintenance cost and benefit are considered in their research. Volovoi [11] applied aging tokens in the Petri net-based reliability model, and the advantages of their method have been illustrated by comparing with classical reliability tools. Katsigiannis et al. [12] presented a new methodology for reliability modeling-based fluid stochastic Petri nets for small isolated power systems. Robidoux et al. [13] presented an algorithm that automatically converts the RBD model into a colored Petri net, and a case study is used to illustrate the effectiveness of the method. Wu et al. [14] established the reliability model for a solar array mechanical system by using fault tree and fuzzy reasoning Petri nets, and their method can be applied to find the fault mechanisms. Chu et al. [15] built a reliability model for the jet pipe servo valve by using generalized stochastic Petri nets, and the effectiveness of their method is illustrated by comparing with the Markov model. In recent years, the application of Petri nets has been extended to many other fields of reliability and safety engineering, which include the reliability analysis of integrated modular avionics [16], the reliability modeling of multimission phased mission system [17], the formal model-based safety analysis [18], and the dependability analysis of safety-critical real-time systems [19].

The Petri nets have displayed a powerful ability in reliability and safety modeling; thus, a stochastic Petri net-based reliability model will be proposed for the fly-by-wire systems in this study. The rest of this paper is structured as follows. In Section 2, a brief description of the stochastic Petri net is presented. In Section 3, the Petri net-based reliability models are constructed for both static and dynamic architectures including series, parallel, m-out-of-n, warm standby, cold standby, and load-sharing architectures, and the substitution transitions are applied in the hierarchical reliability model to simplify complex net structures. In Section 4, a Monte Carlo simulation method is proposed for the stochastic Petri net-based reliability models to generate system lifetime samples, and the system reliability parameters can be calculated in terms of the lifetime samples. In Section 5, a fly-by-wire system is used as a case study to illustrate the application and effectiveness of our proposed reliability approaches. In Section 6, concluding remarks are presented.

2. Definitions Related to the Stochastic Petri Net-Based Reliability Model

2.1. Definitions of the Stochastic Petri Net-Based Reliability Model. According to the definition of the stochastic Petri net [20, 21], the stochastic Petri net-based reliability model can be defined as a 7-tuple $\sum = (\mathbf{P}, \mathbf{T}, \mathbf{F}, K, W, M_0, \mathbf{\Lambda})$ such that

(i) $\mathbf{P} = \{p_1, p_2, \cdots, p_n\}$ is the set of places which are used to denote the states of components or systems

(ii) $\mathbf{T} = \{t_1, t_2, \cdots, t_m\}$ is the set of transitions including timed transitions and immediate transitions; the timed transitions are used to denote the failure process of each component and the immediate transitions are used to denote the relationship among different states

(iii) $\mathbf{F} \subseteq (\mathbf{P} \times \mathbf{T}) \cup (\mathbf{T} \times \mathbf{P})$ is the set of arcs which build the links between places and transitions

(iv) $K : \mathbf{P} \longrightarrow \{1, 2, 3, \cdots\}$ is the capacity function which indicates the number of states for a specific component or system

(v) $W : \mathbf{F} \longrightarrow \{1, 2, 3, \cdots\}$ is the weight function of arcs; the weights contain the imformation about how a transition can be fired and how the state can be changed after the transition has been fired

(vi) $M : \mathbf{P} \longrightarrow \{0, 1, 2, \cdots\}$ is the marking of the net, $\forall p \in \mathbf{P} : M(p) \leq K(p)$, and M_0 is the initial marking; markings reflect the states of all components and the system simultaneously

(vii) $\mathbf{\Lambda} = \{\lambda_1, \lambda_2, \cdots, \lambda_m\}$ is the set of firing rates. The firing rate of the timed transition is the failure rate of the corresponding component, while the firing rate of the immediate transition is an infinite value that means the transition will be fired instantaneously

2.2. Enabled and Fired of Transitions. $\forall x \in \mathbf{P} \cup \mathbf{T}$, $\bullet x$ is called the *preset* of x and $x\bullet$ is called the *postset* of x such that

$$\begin{aligned} \bullet x &= \{y \mid (y \in \mathbf{P} \cup \mathbf{T}) \cap ((y, x) \in F)\}, \\ x\bullet &= \{y \mid (y \in \mathbf{P} \cup \mathbf{T}) \cap ((x, y) \in F)\}. \end{aligned} \tag{1}$$

A transition $t \in \mathbf{T}$ is *enabled* if and only if

$$\begin{cases} \forall p \in \bullet t : & M(p) \geq W(p, t), \\ \forall p \in t\bullet - \bullet t : & M(p) + W(t, p) \leq K(p), \\ \forall p \in t\bullet \cap \bullet t : & M(p) + W(t, p) - W(p, t) \leq K(p). \end{cases} \tag{2}$$

Equation (2) illustrates the prerequisite of the component or system state changing.

When a transition is enabled, it does not imply that it will be immediately fired. Among all the enabled transitions, only the transition that has the minimum firing time will be fired. If several transitions have identical minimum firing time, one of them will be selected randomly to be fired. After the transition t is *fired*, the markings of the net will be changed according to the following rule:

$$\forall p \in \mathbf{P} : M'(p) = \begin{cases} M(p) - W(p,t), & p \in \bullet t\text{-}t\bullet, \\ M(p) + W(p,t), & p \in t\bullet\text{-}\bullet t, \\ M(p) + W(t,p) - W(p,t), & p \in \bullet t \cap t\bullet, \\ M(p), & \text{otherwise.} \end{cases} \tag{3}$$

Equation (3) illustrates how the states will change after the transitions have been fired.

3. Petri Net-Based Reliability Model of Typical Architectures for the Fly-By-Wire Systems

A fly-by-wire system usually consists of three subsystems, which are the sensor subsystem, the flight-control-computer subsystem, and the servo-control subsystem. The sensor subsystem usually has a majority architecture, such as *m-out-of-n architecture*, which means the subsystem will be failed at least $m+1$ of the total n sensors or transducers are failed. The flight-control-computer subsystem usually applies standby architecture, especially the warm standby or hot standby (parallel) architectures. Hence, the uninterruptible command calculation and monitoring can still be provided when part of the redundant channels in the computer is failed. The servo-controls usually have a load-sharing architecture. When one of the redundant actuators is failed, the failure rate of other actuators will increase. To the overall flight-control system, it can be treated as a series system composed of sensing units, flight-control computers, and servo-controls. In this section, the reliability models of all the abovementioned architectures will be presented based on the stochastic Petri net.

3.1. Petri Net-Based Reliability Model of Static Architectures. Failures of the static system are completely decided by the combinations of its component failures. The reliability model of a static system can be expressed by either a dependence diagram (RBD) with series, parallel, and m-out-of-n architectures, or a fault tree with AND, OR, and Voting gates.

In this study, a unified Petri net-based reliability model is proposed to express all static architectures including the series, parallel (hot standby), and *m-out-of-n architectures*. To a system composed of n components, the unified Petri net-based reliability model is shown in Figure 1.

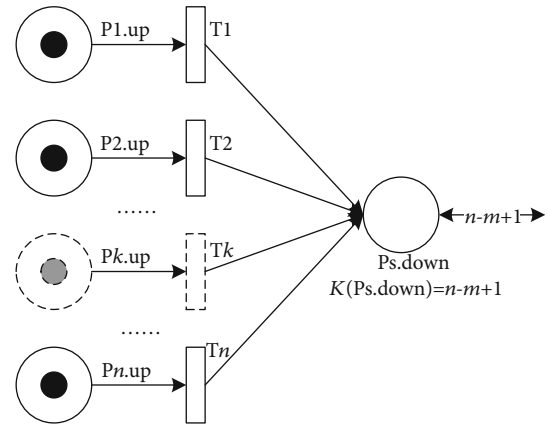

Figure 1: The model of the static architecture.

In Figure 1, the place Pk.up$(k = 1, 2, \cdots, n)$ denotes the operating state of the ith component, and their capacity functions are all equal to 1, namely,

$$K(\mathrm{P}k.\mathrm{up}) = 1. \tag{4}$$

When there is a token in Pk.up, the component k is in the operating state; otherwise, the component k is in the failed state. The place Ps.down denotes the failed state of the system, and its capacity function is decided by the system architecture. To the series architecture, the capacity function of Ps.down is equal to 1, and the system will be failed when there is one token in Ps.down. To the parallel architecture, the capacity function of Ps.down is equal to n, and the system will be failed when there are n tokens in Ps.down. To the *m-out-of-n* architecture, the capacity function of Ps.down is equal to $n - m + 1$, and the system will be failed when there are $n - m + 1$ tokens in Ps.down. The capacity function of the three types of static architectures can be expressed as follows:

$$K(\mathrm{Ps.down}) = \begin{cases} 1, & \text{series}, \\ n, & \text{parallel(hot standby)}, \\ n - m + 1, & m\text{-out-of-}n : \mathrm{G}. \end{cases} \tag{5}$$

The timed transition T$k(i = 1, 2, \cdots, n)$ denotes the failure of the kth component, and its firing time equals the time to failure of the kth component.

The weight of the arc originating from Ps.down is equal to $n - m + 1$, and the weights of all other arcs are equal to 1.

3.2. Petri Net-Based Reliability Model of Standby Architectures. A standby system consists of an active component and one or more standby ones. A sensing and switching mechanism is used to detect failures of the active component and activate the standby ones immediately when a failure of the active one occurs. There are three types of standby architectures including hot standby, warm standby, and cold standby. The hot standby is just the parallel architecture,

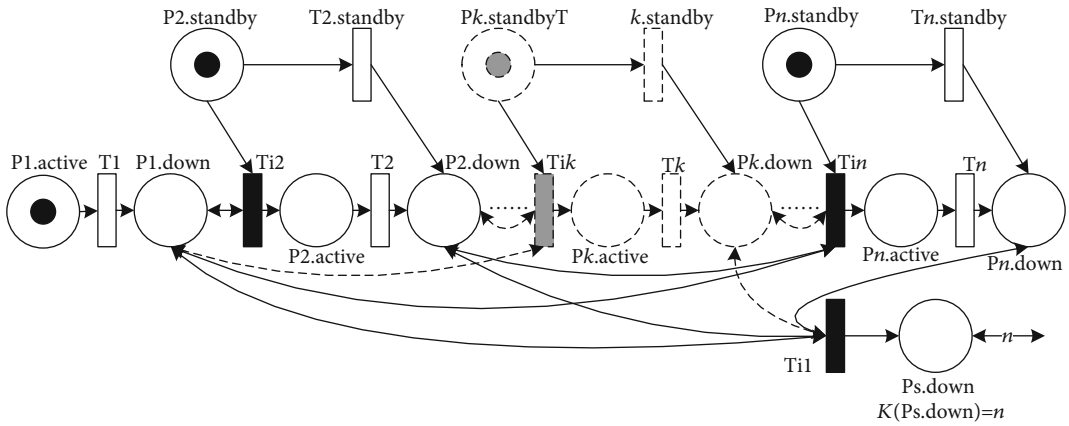

Figure 2: The model of the warm standby system.

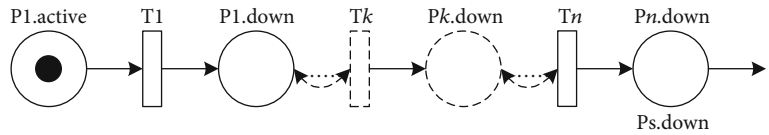

Figure 3: The model of the cold standby system.

and the standby components have the same failure rate as if it was operating in the system. The standby component cannot fail in a cold standby system which is usually assumed for spare or shelf items, whereas the standby component has a lower hazard rate than the operating component in a warm standby system and this is usually a realistic assumption [22]. Additionally, the flight-control system does not have individual sensing and switching mechanisms, whose function will be fulfilled by the monitor module in each channel of the flight-control computers.

The stochastic Petri net-based reliability model of a warm standby system composed of n components is expressed by Figure 2.

In Figure 2, Pk.active, Pk.down, and Pk.standby$(k = 1, 2, \cdots, n)$ denote the active state, the failed state, and the standby state of the kth component, respectively. Their capacity function is equal to 1. Hence,

$$\begin{cases} K(\text{P}k.\text{active}) = 1, \\ K(\text{P}k.\text{down}) = 1, \\ K(\text{P}k.\text{standby}) = 1. \end{cases} \tag{6}$$

When there is a token in Pk.active, Pk.down, or Pk.standby, it means the component k is in the active, failed, or standby state, respectively. The place Ps.down denotes the failed state of the system, and its capacity function is equal to n, namely,

$$K(\text{Ps.down}) = n. \tag{7}$$

The system will be failed when there are n tokens in Ps.down.

The timed transition T$k(k = 1, 2, \cdots, n)$ denotes the failure of the kth component in its active state, and its firing time is also equal to the time to failure of the kth component. The time transition Tk.standby$(k = 2, 3, \cdots, 4)$ denotes the failure of the kth component in the standby state, and its firing time is equal to the time to failure of the kth component in the standby state. Transitions Ti1 to Tin are immediate transitions. Each Tik connects all Pl.down$(l < k)$ by a dual-arrow arc, which means the failures of component 1 to component $k - 1$ are the prerequisite to enable transition Tik, and place Pl.down$(l < k)$ will still hold a token after Tik have been fired.

The weight of the arc originating from Ps.down is equal to n, and the weights of all other arcs are equal to 1. The arcs connecting places Pk.down or Ps.down with transitions are bidirectional. It means the number of tokens in these places will not change after the pertinent transitions have been fired. As the state of a failed component or system will not change in the flight duration.

The stochastic Petri net-based reliability model of a cold standby system composed of n components is given in Figure 3, which is simplified based on Figure 2. In the cold standby system, the place Pn.down is just the place Ps.down.

3.3. Petri Net-Based Reliability of Load-Sharing Architectures. In a load-sharing architecture, there is a dependency between the components. If one component fails, the failure rate of the other components increases as the result of the additional load placed on it. The servo-control with a load-sharing architecture usually has two or three redundant components. The Petri net-based reliability model of a load-sharing architecture with three components can be expressed by Figure 4.

In Figure 4, Pi.up$(i = 1, 2, 3)$ denotes the operating state of the ith component when all the three components are

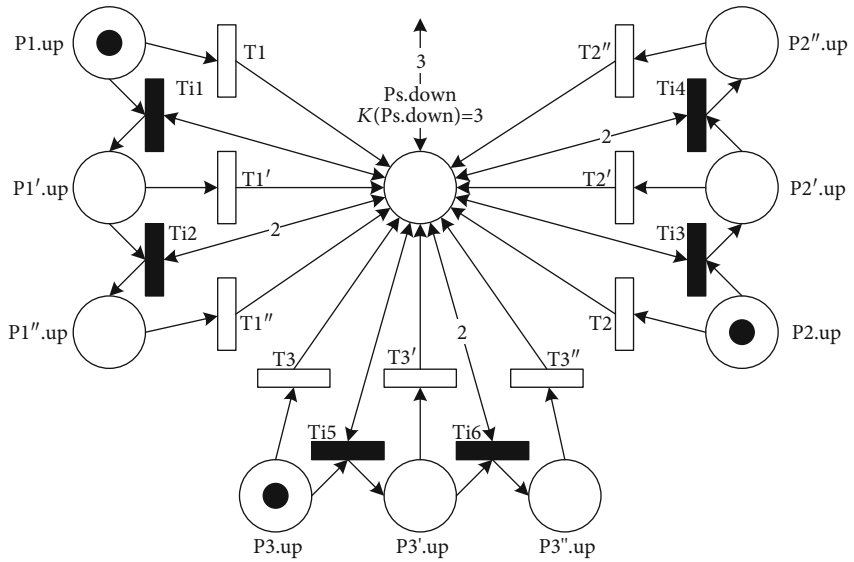

Figure 4: The model of a load-sharing architecture with three components.

operating. $Pi'.up(i = 1, 2, 3)$ denotes the operating state of the ith component when one of the other two components is failed. $Pi''.up(i = 1, 2, 3)$ denotes the operating state of the ith component when the other two components are failed. And the capacity functions of Pi.up, Pi'.up, and Pi''.up are equal to 1; we have

$$\begin{cases} K(\mathrm{P}i.\mathrm{up}) = 1, \\ K\left(\mathrm{P}i''.\mathrm{up}\right) = 1, \\ K\left(\mathrm{P}i''.\mathrm{up}\right) = 1. \end{cases} \tag{8}$$

Ps.down denotes the failed state of the load-sharing system. The load-sharing system will be failed if and only if all the three components are failed; therefore, there will be three tokens in Ps.down when the system is failed. The capacity function of Ps.down is equal to 3. Hence, we have

$$K(\mathrm{Ps.down}) = 3. \tag{9}$$

$\mathrm{T}i(i = 1, 2, 3)$ denotes the failure of the ith component when all the three components are operating. $\mathrm{T}i'(i = 1, 2, 3)$ denotes the failure of the ith component when one of the other two components is failed. $\mathrm{T}i''(i = 1, 2, 3)$ denotes the failure of the ith component when the other two components are failed. The firing times of Ti, $\mathrm{T}i'$, and $\mathrm{T}i''$ are equal to the time to failure of the ith component in the case that all the three components are operating, one of the other two components is failed, and the other two components are failed, respectively. Ti1 to Ti6 are all immediate transitions.

The weight of the arc originating from Ps.down is equal to 3; the weights of the arcs between Ti2 and Ps.down, Ti4 and Ps.down, and Ti6 and Ps.down are all equal to 2; and the weights of all other arcs are equal to 1.

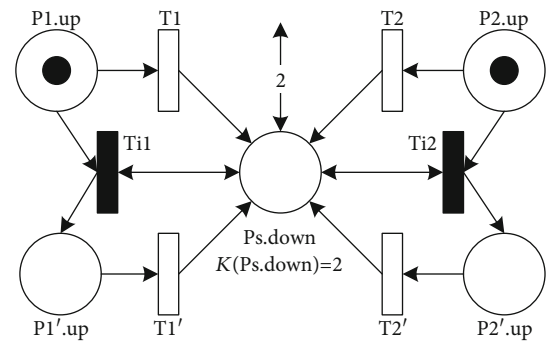

Figure 5: The model of a load-sharing architecture with two components.

To a load-sharing architecture with two components, the Petri net-based reliability model can be simplified. Figure 5 illustrates the model of a load-sharing architecture with two components.

3.4. Petri Net-Based Hierarchical Reliability Model. Creating the Petri net-based reliability model of a large system can be a cumbersome task. A Petri net-based hierarchical reliability model is proposed to simplify the reliability modeling of large systems in this section, and the state space explosion problem can be avoided to some extent.

In our model, a transition is used to represent an entire piece of net architecture or a branch of the Petri net. Such a transition is a substitution transition. Thus, a large Petri net can be simplified by replacing its small pieces of net or branches by substitution transitions. In this study, we use a square to denote the substitution transition. When the substitution transition is used to represent the pieces of net or branch architectures, two immediate transitions

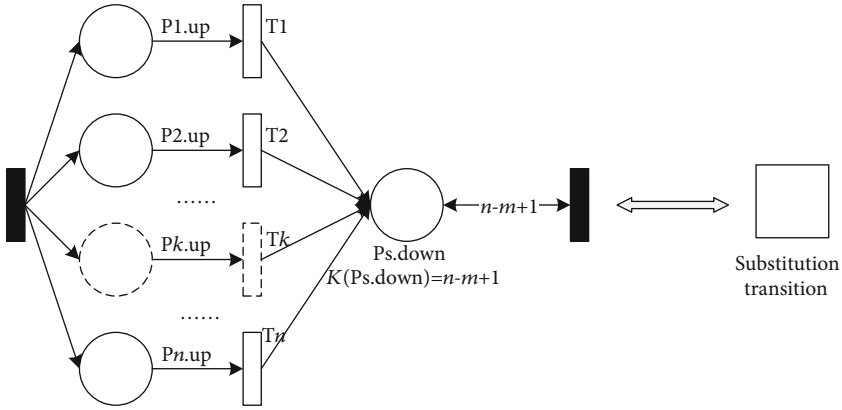

Figure 6: The static architecture represented by the substitution transition.

should be added. One is connected to the Ps.down place, and the other is connected to the places that have tokens in the initial markings. Take the static architecture given in Figure 1 as an example; the corresponding piece of the net architecture represented by the substitution transitions is shown in Figure 6.

4. Stochastic Petri Net-Based Monte Carlo Simulation for Reliability Evaluation

A Monte Carlo simulation method is proposed for the stochastic Petri net-based reliability model, and the lifetime samples can be obtained via the Monte Carlo simulation. In this way, the reliability parameters can be calculated in terms of the lifetime samples.

4.1. Procedure of Stochastic Petri Net-Based Monte Carlo Simulation. The input of our Monte Carlo simulation procedure includes as follows:

(1) The input incidence matrix $\mathbf{W}^- = [W(p_i, t_j)]$

(2) The output incidence matrix $\mathbf{W}^+ = [W(t_j, p_i)]$

(3) The initial marking $\mathbf{M}_0$

(4) The capacity function of each place $K(\bullet)$

(5) The set of firing rate $\mathbf{\Lambda} = \{\lambda_1, \lambda_2, \cdots, \lambda_m\}$ or the firing time distribution of each transition

(6) The maximum number of Monte Carlo simulations $N_{\max}$

The flowchart of the Monte Carlo simulation procedure is given in Figure 7. We can get one sample of the system lifetime from one simulation.

The detailed procedure of the Monte Carlo simulation is as follows.

Step 1. Let $N = 1$ and start the Nth simulation.

Step 2. Variable initialization. Let $\mathbf{M}_{\text{current}} = \mathbf{M}_0$, $\pi_{\text{current}} = 0$, and $\pi_j = 0$ for all j $(j = 1, 2, \cdots, m)$. $\mathbf{M}_{\text{current}}$ is the current marking of the model, π_{current} is the current time, and π_j is the firing time of the transition t_j.

Step 3. Decide whether the Nth simulation can terminate. The Nth simulation will terminate when $M(\text{Ps.down}) = K(\text{Ps.down})$, namely, the number of tokens in Ps.down reaches the value of its capacity function. If $M(\text{Ps.down}) = K(\text{Ps.down})$, the simulation will go to Step 10; otherwise, the simulation will go to Step 4.

Step 4. Determine the enabled transitions. A Boolean variable E_j is used to denote whether the transition t_j is enabled or not. When E_j equals 1, t_j is enabled; otherwise, t_j is not enabled. We let $E_j = 0$ for all $(j = 1, 2, \cdots, m)$. We can determine which transitions are enabled by Equation (2). If t_j is enabled, let $E_j = 1$.

Step 5. Update the firing time for all transitions. For each nonenabled transition $(E_j = 0)$, let its firing time equal 0 $(\pi_j = 0)$. For each enabled transition $(E_j = 1)$, if its original firing time is 0 $(\pi_j = 0)$, a random variable will be generated as its new firing time according to its firing rate λ_j; otherwise, its firing time will not be changed. Additionally, the random variable of the firing time that does not follow exponential distribution (the firing rate is not a constant value) will be generated in terms of their firing time distributions. And the random variable of the substitution transitions' firing time can be obtained via the Monte Carlo simulation of the corresponding piece of net architecture.

Step 6. Determine the fired transition. Among all the enabled transitions, the transition that has the minimum firing time $(\pi_{\min})$ will be fired. If several transitions have identical minimum firing time, we can select one of them randomly to be fired. A Boolean variable F_j is used to denote whether the transition t_j is enabled or not. If F_j equals 1, t_j is enabled; otherwise, t_j is not enabled.

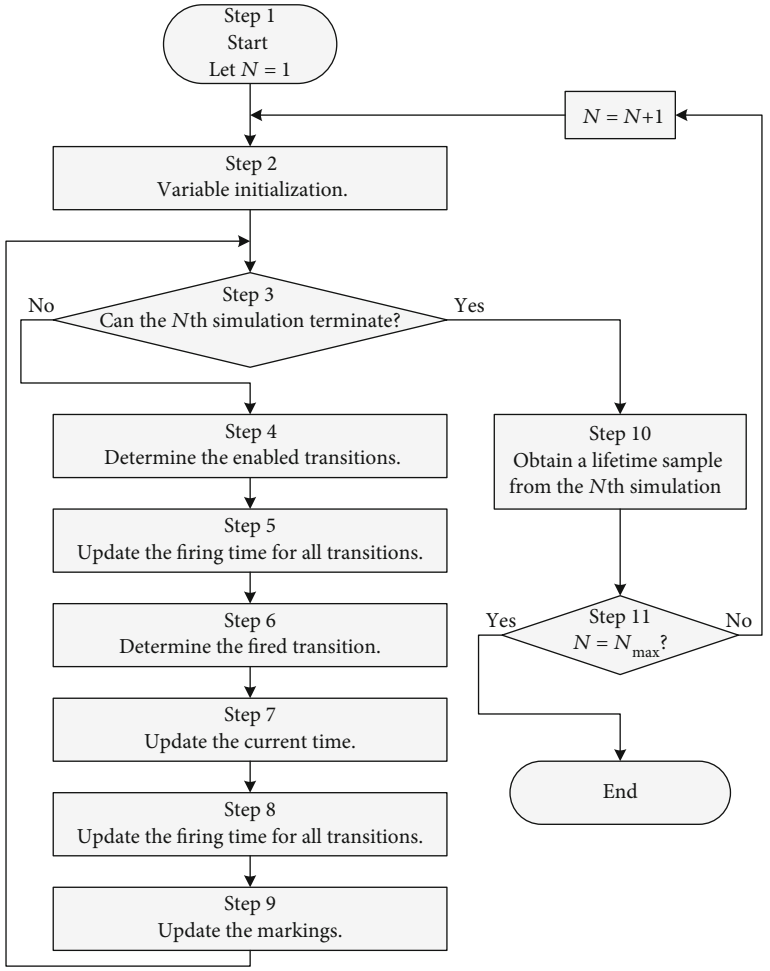

Figure 7: The Monte Carlo simulation procedure of the stochastic Petri net.

Step 7. Update the current time. The current time π_{current} will be updated by $\pi_{\text{current}} + \pi_{\min}$, i.e., let $\pi_{\text{current}} = \pi_{\text{current}} + \pi_{\min}$.

Step 8. Update the firing time for all enabled transitions. The firing time of the enabled transition t_j (π_j) will be updated by $\pi_j - \pi_{\min}$, i.e., let $\pi_j = \pi_j - \pi_{\min}$.

Step 9. Update the markings. The markings will be updated by the state function of the Petri net, i.e., let $\mathbf{M} = \mathbf{M} + \mathbf{C} \times \mathbf{F}$. The jth element of $\mathbf{F}$ is the Boolean variable F_j. And go back to Step 2.

Step 10. Obtain a sample of system lifetime. The current time π_{current} will be a sample of the system lifetime.

Step 11. Decide whether all simulations have been finished. If $N = N_{\max}$, it means all simulations have been finished, and the procedure will end; otherwise, we let $N = N + 1$, and the next simulation will start (go back to Step 2).

4.2. Reliability Parameter Calculation Based on Lifetime Samples. Let $s_1, s_2, \cdots, s_{N_{\max}}$ denote the $N_{\max}$ ordered lifetime samples, i.e., $s_1 \le s_2 \le \cdots \le s_{N_{\max}}$; the estimate of reliability can be expressed as follows:

$$\hat{R}(s_i) = \frac{N_{\max} - i}{N_{\max}}. \tag{10}$$

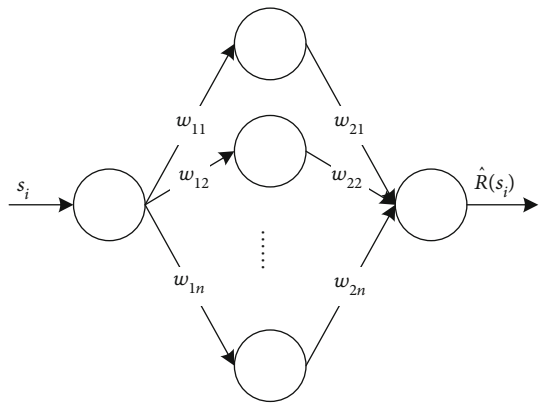

Figure 8: The BP neural network structure used for reliability regression.

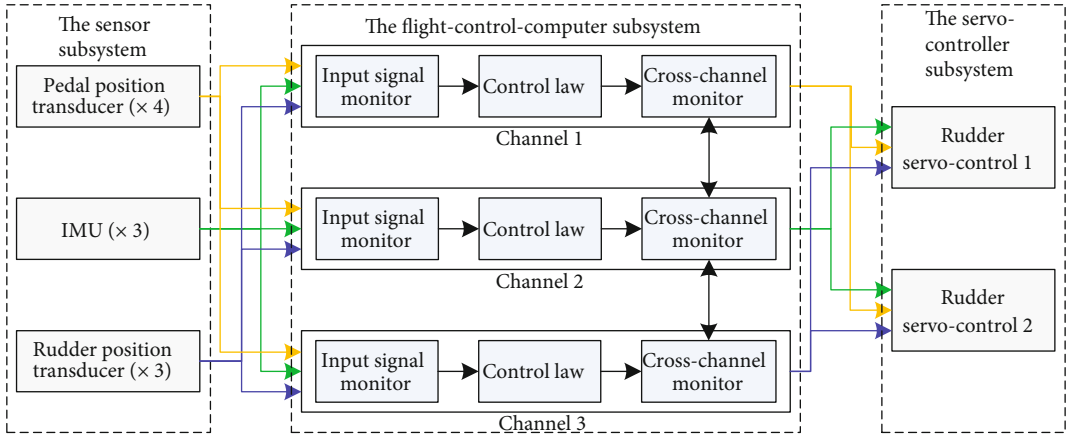

Figure 9: The architecture of the rudder control system.

In this way, we can get a corresponding reliability value $\widehat{R}(s_i)$ for each lifetime sample s_i. By treating the $N_{\max}$ pairs of s_i and $\widehat{R}(s_i)$ as training samples, we can get the regression curve of reliability function by using the backpropagation (BP) neural network. The method we used here is a kind of nonparametric regression; we do not make the assumption that the lifetime follows a specific distribution such as exponential distribution, log-normal distribution, or Weibull distribution. This is reasonable that the lifetime of a complex system may not follow a specific distribution.

The BP neural network we used has three layers, which are the input layer, the hidden layer, and the output layer. Both the input layer and the output layer only have one unit (neuron); the input unit denotes the lifetime sample s_i and the output unit denotes the corresponding estimate of reliability function $\widehat{R}(s_i)$. The neural network we used is shown in Figure 8.

According to the structure of the BP neural network, the reliability function can be expressed as follows:

$$R(t) = \frac{1}{1 + \exp\left[-\sum_{i=1}^{n}\left((w_{2i})/(1 + \exp\left(-w_{1i}t\right))\right)\right]}, \quad (11)$$

where w_{1i} and w_{2i} are weights, and we can get the optimal values of weights by backpropagation algorithm.

5. Case Study

In the case study, a rudder control system of a commercial aircraft is used to illustrate the application and effectiveness of our proposed approach.

5.1. System Description. The rudder control system also consists of the sensor subsystem, the flight-control-computer subsystem, and the servo-control subsystem. Figure 9 shows the architecture of the rudder control system.

The sensor subsystem consists of the pedal position transducers, the inertial measurement units (IMUs), and the rudder position transducers. The pedal position transducer subsystem has a 2-out-of-4 architecture, and both the IMU subsystem and the rudder position transducer subsystem have a 2-out-of-3 architecture, namely, the triple modular redundancy (TMR).

The flight-control-computer subsystem has three independent channels. Each channel has an input signal monitor, a set of control laws, and a cross-channel monitor. The three channels can become any one of these three channels: the command channel, the standby channel, and the monitor channel. The command channel transmits its command output signals to the servo-control subsystem. It also has a channel monitoring function to find and isolate failures in the standby and monitor channels. The standby channel transmits test data only, and it becomes the command channel and transmits its command output signals to the servo-control subsystem when a failure causes the command channel to shut down. It also performs monitoring functions to find and isolate failures in the command and monitor channels. The monitor channel also transmits test data only. Its command output signals are used internally to find failures in the command and standby channels. The monitor channel becomes the standby channel when the command channel shuts down. The reliability model of the flight-control-computer subsystem has the characteristics of both the standby and the 2-out-of-3 architectures. The command channel is the active state; both the standby and monitor channels are in the warm standby state. The subsystem will be failed if and only if at least two of the three channels are failed.

The hydraulic actuation is achieved by the servo-control subsystem composed of two electrohydraulic servo-controls for the rudder. Each servo-control has two kinds of operation modes, which are the active mode (state) and the damping mode. Both the servo-controls are in the active mode initially, and they have identical failure rate in this situation. When one of the servo-controls is failed, the failed servo-control will be in the damping mode and the other one will be in the active mode. And the failure rate of the servo-control in the active mode will increase.

5.2. Petri Net-Based System Reliability Model. The Petri net-based reliability model of the pedal position transducer subsystem can be expressed as Figure 10(a), and both the

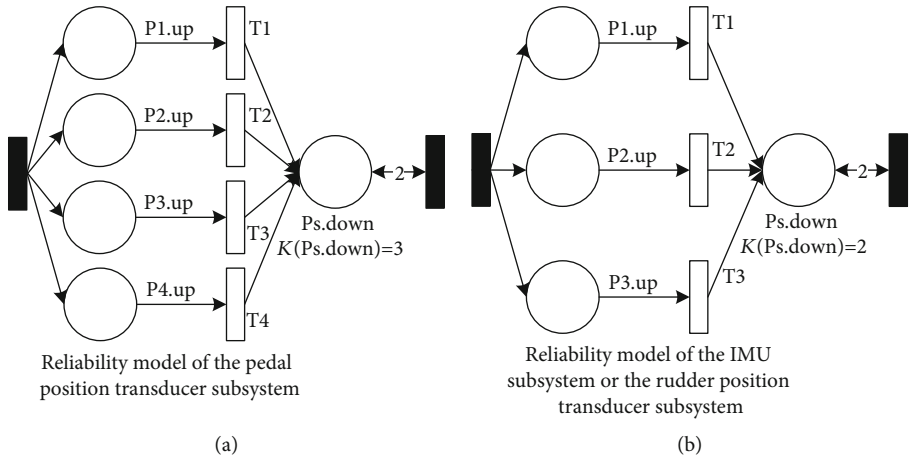

Figure 10: Petri net-based reliability model of the sensor subsystem.

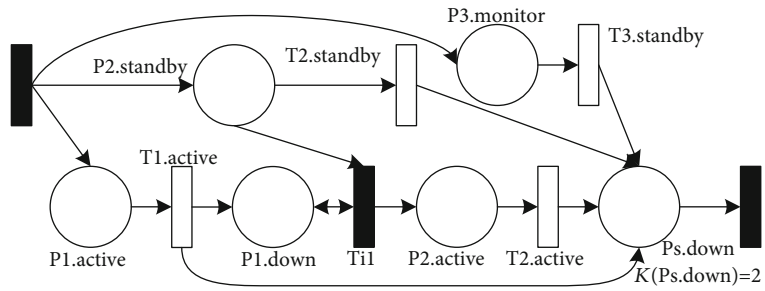

Figure 11: Petri net-based reliability model of the flight-control-computer subsystem.

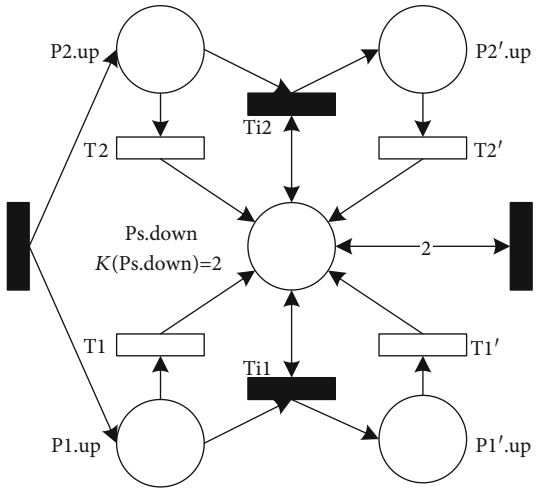

Figure 12: Petri net-based reliability model of the servo-control subsystem.

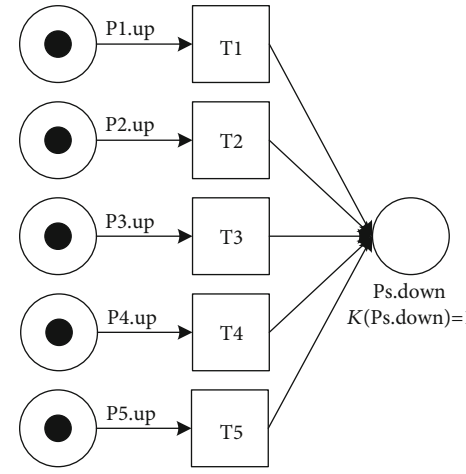

Figure 13: The Petri net-based reliability model of the rudder control system.

IMU subsystem and the rudder position transducer subsystem can be expressed as Figure 10(b).

The Petri net-based reliability model of the flight-control-computer subsystem is shown in Figure 11.

The Petri net-based reliability model of the servo-control subsystem is shown in Figure 12.

The Petri net-based reliability model of the rudder control system is given in Figure 13.

In Figure 13, the place P1.up to P5.up denote the operating state of the pedal position transducer subsystem, the IMU subsystem, the rudder position transducer subsystem, the flight-control-computer subsystem, and the servo-control subsystem, respectively. T1 to T5 are all substitution

Table 1: Component failure rates of the rudder control system.

Pedal position transducer	IMU	Rudder position transducer	Flight-control-computer channel	Servo-control
$\lambda_P = 10^{-6}$/h	$\lambda_I = 10^{-7}$/h	$\lambda_R = 10^{-7}$/h	$\lambda_F = 2 \times 10^{-7}$/h $\lambda_F^- = 0.6 \times 10^{-7}$/h	$\lambda_S = 10^{-6}$/h $\lambda_S^+ = 1.5 \times 10^{-6}$/h

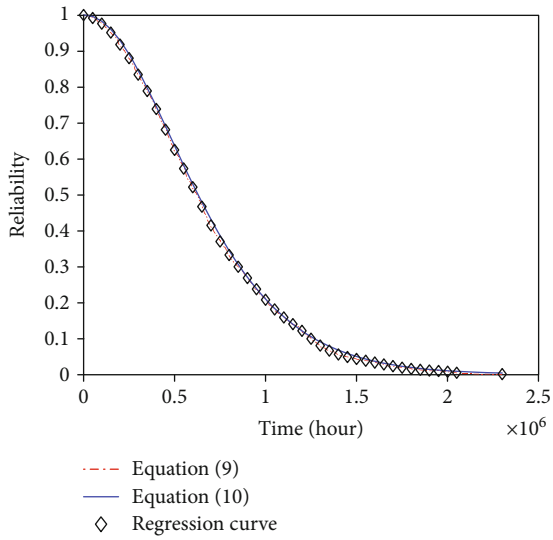

Figure 14: The reliability curves obtained by different methods.

transitions which are used to denote the failure of the above-mentioned subsystems.

5.3. Results and Discussion. The component failure rates of the rudder control system are given in Table 1.

In Table 1, λ_P, λ_I, λ_R, λ_F, and λ_S are the failure rates in the active state of pedal position transducers, IMUs, rudder position transducers, flight-control-computer channels, and servo-controls, respectively. λ_F^- is the failure rate of the flight-control-computer channels operating as the standby or monitor channels. $\lambda_S^+ = 1.5 \times 10^{-6}$ is the increased failure rate the sever-control when the other one is failed. We let $N_{max} = 1000$, and 1000 lifetime samples of the system can be obtained. The reliability curve calculated by Equation (10) is shown in Figure 14. The analytical reliability function and the regression curve of the reliability function obtained by the BP neural network are also shown in Figure 13. The BP neural network used in this case study has ten neurons in the hidden layer.

The analytical expression of the system reliability function can be expressed as follows:

$$R(t) = R_P(t)R_I(t)R_R(t)R_F(t)R_S(t), \tag{12}$$

where $R_P(t)$, $R_I(t)$, $R_R(t)$, $R_F(t)$, and $R_S(t)$ are the reliability function of the pedal position transducer subsystem, the IMU subsystem, the rudder position transducer subsystem, the flight-control-computer subsystem, and the servo-control subsystem, respectively.

$R_P(t)$, $R_I(t)$, and $R_R(t)$ can be calculated by RBD, and $R_F(t)$ and $R_S(t)$ can be calculated by the Markov process as all failure rates are constant values. They can be expressed as follows:

$$\begin{aligned}
R_P(t) &= \sum_{i=2}^{4} C_4^i \left(e^{-\lambda_P t}\right)^i \left(1 - e^{-\lambda_P t}\right)^{4-i}, \\
R_I(t) &= \sum_{i=2}^{3} C_3^i \left(e^{-\lambda_I t}\right)^i \left(1 - e^{-\lambda_I t}\right)^{3-i}, \\
R_R(t) &= \sum_{i=2}^{3} C_3^i \left(e^{-\lambda_R t}\right)^i \left(1 - e^{-\lambda_R t}\right)^{3-i}, \\
R_F(t) &= \left(\frac{\lambda_F + 2\lambda_F^-}{\lambda_F^-}\right) e^{-(\lambda_F + \lambda_F^-)t} - \left(\frac{\lambda_F + \lambda_F^-}{\lambda_F^-}\right) e^{-(\lambda_F + 2\lambda_F^-)t}, \\
R_S(t) &= e^{-2\lambda_S t} + \frac{2\lambda_S}{2\lambda_S - \lambda_S^+}\left(e^{-\lambda_S^+ t} - e^{-2\lambda_S t}\right),
\end{aligned} \tag{13}$$

where C_n^m is the combination formula which can be expressed as follows:

$$C_n^m = \frac{n!}{m!(n-m)!}. \tag{14}$$

Figure 15 shows the error between the analytical reliability function and the regression reliability function. And the maximum value of the error is 0.0174 when the lifetime is 723154 hours.

To the system of a civil aircraft, we usually care about the reliability in a flight duration which varies from a few hours to a dozen hours. We set the flight duration as 15 hours arbitrarily; the results obtained by both the analytical method and our Monte Carlo simulation-based regression method show that the reliability of the fly-by-wire system in a flight duration is greater than 99.99%.

Figure 16 shows the error between the analytical reliability function and the regression reliability function in the flight duration. The maximum error is 8.28×10^{-5} and the minimum error is 8.10×10^{-5} which illustrate that the error in a flight duration is negligible, although the error is greater than 0.01 in the whole life as shown in Figure 14.

6. Conclusion

In this study, stochastic Petri net-based reliability models are established for series, parallel, *m*-out-of-*n*, warm standby, cold standby, and load-sharing architectures, which

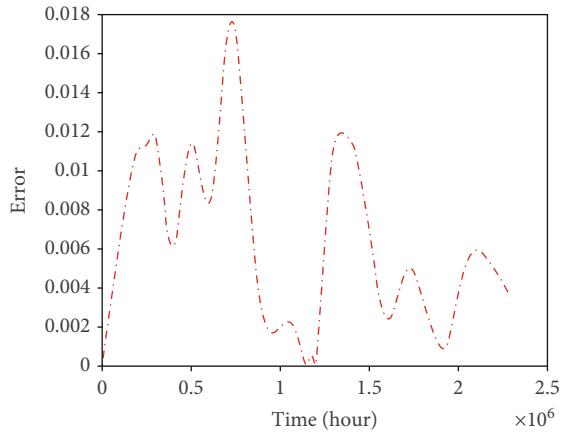

Figure 15: The error between the analytical function and the regression function.

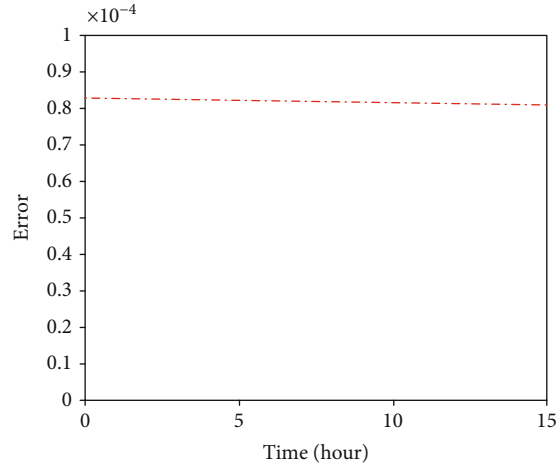

Figure 16: The error between the two methods in a flight duration.

are commonly used in the fly-by-wire system. And a reliability evaluation approach is proposed by using Monte Carlo simulation in terms of the stochastic Petri net-based reliability models.

Compared with the traditional reliability modeling tools such as RBD (DDA), FTA, and MA, our proposed approaches have the following advantages:

(1) The time-dependent failure characteristics can be expressed by stochastic Petri net-based reliability models, and the model can be simplified by using the substitution transitions to construct a hierarchical reliability model. In this way, the state space explosion of the Markov model can be reduced to some extent

(2) Our stochastic Petri net-based Monte Carlo simulation is suitable for all kinds of lifetime distributions; the limits of the Markov model that all component lifetimes should follow exponential distributions can be overcome

(3) By using our proposed Monte Carlo simulation method, the cumbersome work of solving the differential equations for the Markov chains can be avoided. Meanwhile, the error of the reliability value in a flight duration obtained by our simulation method is less than 1×10^{-4} that is negligible

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors wish to appreciate the support from the National Natural Science Foundation of China (U1733124) and the Aeronautical Science Foundation of China (20180252002).

References

[1] S. Distefano and A. Puliafito, "Dependability evaluation with dynamic reliability block diagrams and dynamic fault trees," *IEEE Transactions on Dependable and Secure Computing*, vol. 6, no. 1, pp. 4–17, 2009.

[2] G. Merle, J. M. Roussel, and J. J. Lesage, "Algebraic determination of the structure function of dynamic fault trees," *Reliability Engineering & System Safety*, vol. 96, no. 2, pp. 267–277, 2011.

[3] SAE International S-18 Committee, *ARP4754A: Guidelines for Development of Civil Aircraft and Systems*, Society of Automotive Engineers, Warrendale, PA, USA, 2010.

[4] SAE International S-18 Committee, *ARP4761, Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne System and Equipment*, Society of Automotive Engineers, Warrendale, PA, USA, 1996.

[5] K. Durga Rao, V. Gopika, V. V. S. Sanyasi Rao, H. S. Kushwaha, A. K. Verma, and A. Srividya, "Dynamic fault tree analysis using Monte Carlo simulation in probabilistic safety assessment," *Reliability Engineering & System Safety*, vol. 94, no. 4, pp. 872–883, 2009.

[6] Z. Lu, J. Zhou, and X. Li, "Monte Carlo simulation based time limited dispatch analysis with the constraint of dispatch reliability for electronic engine control systems," *Aerospace Science and Technology*, vol. 72, no. 1, pp. 397–408, 2018.

[7] G. S. Hura and J. W. Atwood, "The use of Petri nets to analyze coherent fault trees," *IEEE Transactions on Reliability*, vol. 37, no. 5, pp. 469–474, 1988.

[8] M. Malhotra and K. S. Trivedi, "Dependability modeling using Petri-nets," *IEEE Transactions on Reliability*, vol. 44, no. 3, pp. 428–440, 1995.

[9] T. S. Liu and S. B. Chiou, "The application of Petri nets to failure analysis," *Reliability Engineering & System Safety*, vol. 57, no. 2, pp. 129–142, 1997.

[10] W. G. Schneeweiss, "Tutorial: Petri nets as a graphical description medium for many reliability scenarios," *IEEE Transactions on Reliability*, vol. 50, no. 2, pp. 159–164, 2001.

[11] V. Volovoi, "Modeling of system reliability Petri nets with aging tokens," *Reliability Engineering & System Safety*, vol. 84, no. 2, pp. 149–161, 2004.

[12] Y. A. Katsigiannis, P. S. Georgilakis, and G. J. Tsinarakis, "A novel colored fluid stochastic Petri net simulation model for reliability evaluation of wind/PV/diesel small isolated power systems," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 40, no. 6, pp. 1296–1309, 2010.

[13] R. Robidoux, H. Xu, L. Xing, and M. C. Zhou, "Automated modeling of dynamic reliability block diagrams using colored Petri nets," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 40, no. 2, pp. 337–351, 2010.

[14] J. Wu, S. Yan, and L. Xie, "Reliability analysis method of a solar array by using fault tree analysis and fuzzy reasoning Petri net," *Acta Astronautica*, vol. 69, no. 11-12, pp. 960–968, 2011.

[15] Y. Chu, Z. Yuan, and J. Chen, "Research on dynamic reliability of a jet pipe servo valve based on generalized stochastic Petri nets," *International Journal of Aerospace Engineering*, vol. 2015, Article ID 171642, 8 pages, 2015.

[16] W. Yun-Sheng, L. Hang, and H. Xuan, "The stochastic Petri net based reliability analysis for software partition integrated modular avionics," *IEEE Aerospace and Electronic Systems Magazine*, vol. 30, no. 4, pp. 30–37, 2015.

[17] X. Wu and X. Wu, "Mission reliability modeling and evaluation of multi-mission phased mission system based on an extended object-oriented Petri net," *Eksploatacja i Niezawodnosc - Maintenance and Reliability*, vol. 19, no. 2, pp. 244–253, 2017.

[18] D. Wu and W. Zheng, "Formal model-based quantitative safety analysis using timed coloured Petri nets," *Reliability Engineering & System Safety*, vol. 176, no. 8, pp. 62–79, 2018.

[19] L. K. Singh and H. Rajput, "Dependability analysis of safety critical real-time systems by using Petri nets," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 2, pp. 415–426, 2018.

[20] R. David and H. Alla, *Discrete, Continuous, and Hybrid Petri Nets*, Springer, Berlin, Germany, Second edition, 2010.

[21] P. Z. Louchka, *Time and Petri Net*, Springer, Berlin, Germany, 2013.

[22] K. C. Kapur and M. Pecht, *Reliability Engineering*, John Wiley & Sons, Inc, Hoboken, NJ, USA, 2014.