IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 20, 2020, accepted May 14, 2020, date of publication May 22, 2020, date of current version June 4, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2996643

A Hybrid Modular Approach for Dynamic Fault Tree Analysis

SOHAG KABIR[1], KOOROSH ASLANSEFAT[2], (Member, IEEE), IOANNIS SOROKOS[2], YIANNIS PAPADOPOULOS[2], AND SAVAS KONUR[1], (Member, IEEE)
[1]Department of Computer Science, University of Bradford, Bradford BD7 1DP, U.K.
[2]Department of Computer Science and Technology, University of Hull, Hull HU6 7RX, U.K.

Corresponding author: Koorosh Aslansefat (k.aslansefat-2018@hull.ac.uk)

This work was supported in part by the Dependability Engineering Innovation for Cyber Physical Systems (CPS) (DEIS) H2020 Project under Grant 732242, and in part by the LIVEBIO: Light-weight Verification for Synthetic Biology Project under Grant EPSRC EP/R043787/1.

ABSTRACT Over the years, several approaches have been developed for the quantitative analysis of dynamic fault trees (DFTs). These approaches have strong theoretical and mathematical foundations; however, they appear to suffer from the state-space explosion and high computational requirements, compromising their efficacy. Modularisation techniques have been developed to address these issues by identifying and quantifying static and dynamic modules of the fault tree separately by using binary decision diagrams and Markov models. Although these approaches appear effective in reducing computational effort and avoiding state-space explosion, the reliance of the Markov chain on exponentially distributed data of system components can limit their widespread industrial applications. In this paper, we propose a hybrid modularisation scheme where independent sub-trees of a DFT are identified and quantified in a hierarchical order. A hybrid framework with the combination of algebraic solution, Petri Nets, and Monte Carlo simulation is used to increase the efficiency of the solution. The proposed approach uses the advantages of each existing approach in the right place (independent module). We have experimented the proposed approach on five independent hypothetical and industrial examples in which the experiments show the capabilities of the proposed approach facing repeated basic events and non-exponential failure distributions. The proposed approach could provide an approximate solution to DFTs without unacceptable loss of accuracy. Moreover, the use of modularised or hierarchical Petri nets makes this approach more generally applicable by allowing quantitative evaluation of DFTs with a wide range of failure rate distributions for basic events of the tree.

INDEX TERMS Reliability analysis, fault tree analysis, dynamic fault trees, modularisation, petri nets.

I. INTRODUCTION

Safety-critical systems are widely used in many industries. Reliability engineering concentrates on assuring safety and reliability of such systems by identifying potential risks that may be caused by their failure and thereby determining necessary actions to reduce the likelihood of these risks. Research efforts have been made to develop reliability models to improve system safety and optimize system behaviour by taking into account system performance and components' failure probability [1], [2]. Modern engineering systems are getting increasingly complex and their behaviour is becoming more dynamic, leading to a variety of dynamic failure characteristics such as functional dependent events and priorities of failure events. Classical combinatorial fault trees [3] are unable to capture a system's dynamic failure behaviour. The Dynamic Fault Tree (DFT) was introduced by Dugan *et al.* [4] to model the dynamic failure behaviour of systems.

For the quantitative analysis of DFTs, they are typically converted to continuous-time Markov chain and then a set of ordinary differential equations representing the Markov chain are numerically solved [5]–[8]. The computational complexity of Markov model-based approaches increases exponentially with the increase in the number of system components as it causes an equivalent increase of the Markov chain states. Moreover, the application of Markov-chain-based approaches is under the assumption that the system failure

The associate editor coordinating the review of this manuscript and approving it for publication was Youqing Wang.

data is exponentially distributed. To overcome this limitation, other approaches such as Petri net-based approaches [9]–[12], Bayesian Network-based approaches [13]–[15], sequential binary decision diagrams (SBDD) [16], [17], Boolean logic Driven Markov Process [18], [19], Dynamic Reliability Block Diagrams [20], [21], stochastic methods [22], and a hybrid method with the combination of stochastic methods and simulation [23]–[25] have been proposed. These approaches can provide exact solutions, however, non-exact solutions to DFTs can be obtained via simulation approaches [26], [27]. The simulation requires more memory and takes much longer than analytical models to compute. The issues of state-space explosion and failure data distribution have been addressed in [28], [29] by formalizing an algebraic approach. This approach can synthesise the structure-function of any DFT. The computational effort required to find a closed-form solution to a DFT using this approach can be prohibitively expensive. Note that there are different tools developed to support the DFT analysis based on the concepts mentioned above. For instance, Galileo [30] and Altarica [31] support DFT analysis through the use of Markov chains, therefore, they inherit the issues associated with the Markov chain. At the same time, tools like DFTSim and MatCarloRe use Monte Carlo simulation as a mean to quantify DFTs, thus would require long computation time due to the use of Monte Carlo simulation. There are other tools, which have their strengths and weaknesses. A list of such other DFT analysis tools can be found in [32].

A. RELATED WORK AND MOTIVATION

To address the issue of high computational effort involved in solving large fault trees, modularisation (a.k.a. hierarchical) approaches have been developed and used with great effectiveness. The early application of modularised techniques to solve fault trees can be traced back to the 1990s [33], [34]. DIFtree [35], a modularisation technique for DFT analysis, follows the divide-and-conquer strategy to solve the DFTs by dividing the system-level DFTs into independent static and dynamic sub-trees. The static and dynamic sub-trees are then solved using Binary Decision Diagrams (BDDs) [36] and Markov chains, respectively. Finally, these smaller solutions to the sub-trees are combined to solve the whole DFT. In DIFtree, independent sub-trees, which have no shared input, were identified using the algorithm proposed by Dutuit and Rauzy [37]. The same authors have further formalised and operationalised the modular FTA approach in [38].

A similar solution to DFTs based on Rauzy's linear time modularisation algorithm [37] can be found in [39]. Later, Manian *et al.* [40] extended the DIFtree approach to allow modelling different lifetime distributions for the system components with the help of Monte Carlo simulation. A major drawback of modularisation approaches is that it is difficult to perform a sensitivity analysis of the eliminated basic events once the state space of the Markov model has been reduced. Moreover, in these approaches, if the module's top-level gate is dynamic then further modularisation is not performed. To address these issues, Huang and Chang [41] proposed an approach which can further modularise a dynamic module if an independent module exists within it. The approach is also capable of performing sensitivity analysis even after the elimination of basic events through modularisation. In [42], a modular approach was proposed by showing that further modularisation of a DFT is possible in a set of cases. A Weibull-distribution-based modularisation scheme was proposed in [43] where both analytical and simulation techniques were used to solve DFTs. Table 1 shows a comparison between different features of the existing modularisation-based DFT analysis approaches. The table outlines the previous approaches with their capabilities and limitations.

In the literature, modularisation techniques have been proven to be highly effective in improving the computing performance of DFT quantification processes. However, there exist a few issues that require further research. For instance, it can be seen from Table 1 that most of the existing modularisation approaches use Markov chains to solve dynamic modules. As Markov chains are only applicable given an exponentially distributed failure rate, the use of Markov chains limits the application of these approaches to a particular class of DFTs. Therefore, it is beneficial to utilise other DFT solution approaches in a modularisation scheme, which can alleviate the above limitation, thus making the scheme capable of solving more general types of DFTs. Moreover, in most existing modularisation schemes, dynamic modules are not decomposed further even when they contain independent modules within them. Furthermore, most of these approaches are not capable of performing sensitivity/criticality analysis of basic events due to modularisation.

At this point, the contribution of the method proposed in this publication and its improvement over previous approaches can be stated. This paper seeks to address the issues highlighted previously by proposing a modularisation scheme, which can provide all the features as mentioned in Table 1. Like the existing approaches, firstly, the proposed approach identifies the independent static and dynamic modules in a DFT. Afterwards, the static modules are solved using algebraic formulas and the dynamic modules are solved using Petri nets (PN) [44], the widespread use of which in safety and reliability analysis is reported in [45]. In the literature, the readers can find many extensions of PNs that can model both exponentially and non-exponentially distributed transition rates. For instance, the use of Weibull distribution in PN was shown in [46], [47]. In addition to the Weibull distribution, the use of other types of distributions such as normal and lognormal distribution was shown in [48], [49]. A detailed description of the different types of PNs is out of the scope of this paper. However, interested readers can find more information about different kinds of Petri nets in [44], [50]. The use of PNs for the evaluation of dynamic modules can support allocating different distributions for failure rates. Moreover, due to the state-space explosion problem, while it is infeasible to create Markov states for the behaviour

TABLE 1. Comparison between the existing approaches in temrs of their features.

Approaches	Models used	F1	F2	F3	F4
Pullum and Dugan [34]	Markov chain and BDD	Yes	No	No	No
Dugan et al. [35], [38]	Markov chain and BDD	Yes	No	No	No
Anand and Somani [39]	Markov chain and BDD	Yes	No	No	No
Manian *et al.* [40]	Markov chain, BDD, and Monte Carlo simulation	Yes	Yes	No	No
Huang and Chang [41]	Markov chain and Algebraic	Yes	No	Yes	Yes
Yevkin [42]	Markov chain and Algebraic	Yes	No	Yes	No
Chiacchio *et al.* [43]	Markov chain, Algebraic, and Monte Carlo simulation	Yes	Yes	No	No
Proposed appraoch	PN and algebraic solution	Yes	Yes	Yes	Yes

F1: Exponential distribution; **F2:** Non-exponential distribution; **F3:**Modularisation in dynmaic sub-trees; **F4:** Improtance measures

of small-scale systems, PNs can be used for formal and compact presentation of the behaviour of large-scale systems [49]. Moreover, the proposed approach allows sensitivity analysis to be performed even after modularisation. The effectiveness of the approach is illustrated by applying it to five different DFTs. The results show that the approach finds an approximate solution to DFTs without losing unacceptable accuracy. The contributions of this paper can be summarised as:

- Proposal of a modularised solution to DFTs that can reduce the computational complexity and increase the efficiency of the existing DFT quantification approaches.
- Enabling the integration of multiple solutions such as algebraic, Petri Nets and their reachability tree and Monte Carlo simulation in a single place, thus allowing to take advantages of their strong features to solve a wide range of DFTs. For example, consider a DFT with two large independent modules that can be solved very fast and simple with the algebraic solution and one small module that has non-exponential and shared basic events that can be easily solved with Monte Carlo solution. In the proposed approach, the best solution for each independent module is detected and applied to increase the efficiency and use the advantages of all existing methods.
- Introduction of a modified version of Birnbaum importance measure as part of the proposed hybrid approach to determine the criticality of basic events.
- The capabilities and accuracy of the proposed method are illustrated and compared through using different well-known hypothetical and industrial case studies facing issues such as repeated basic events and non-exponential failure distributions.

II. DYNAMIC FAULT TREE ANALYSIS

The DFT extends the capability of static fault trees (SFTs) by introducing dynamic gates like the Priority AND (PAND), Priority OR (POR), Functional dependency (FDEP), SPARE, and SEQ to model time-dependent failure behaviour of systems. Fig. 1 shows the graphical symbols of the commonly used DFT gates. Detailed information about the definitions and functional behaviour of these gates can be found

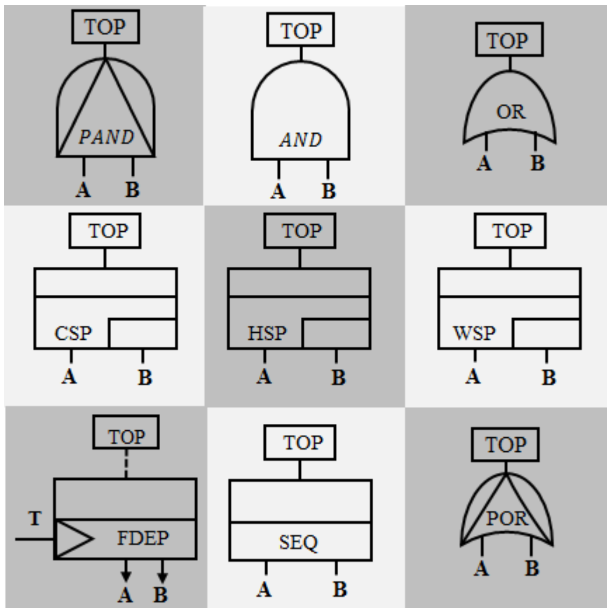


FIGURE 1. Commonly used logic gates in DFT.

in [4], [51], [52]. Similar to SFTs, a DFT analysis follows a top-down procedure, starting from the undesired system-level top event (TE), which represents the system failure condition. The TE is decomposed into a combination of intermediate events. The intermediate events are further decomposed using Boolean and dynamic logic gates down to the specification of the lowest-level event causes, named Basic Events (BEs).

DFTs can be analysed both qualitatively and quantitatively. Through qualitative analysis, minimal cut sequences (MCSQs) can be obtained from DFTs. These MCSQs show how different sequences of events can cause a system failure. On the other hand, the quantitative analysis focuses on calculating system failure probability and other reliability indices based on the failure data of the DFT's basic events. As mentioned earlier, there are several approaches available for the quantitative analysis of DFTs. As the approach proposed in this paper uses the algebraic solution for the static gates and PNs for the dynamic gates, we concentrate on these only.

AND and OR are the commonly used Boolean gates in DFTs. If the probability of a basic event (BE) i at time t is

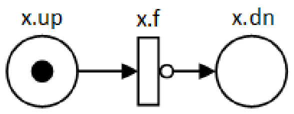


FIGURE 2. PN model of the failure behaviour of a non-repairable component.

given as $Pr\{BE_i\}(t)$ and an AND gate contains n statistically independent BEs, then the probability of that AND can be calculated as:

$$Pr\{E_1 \wedge E_2 \wedge \ldots \wedge E_n\}(t) = \prod_{i=1}^{n} Pr\{E_i\}(t) \qquad (1)$$

Similarly, an OR gate with n BEs as inputs can be evaluated as:

$$Pr\{E_1 \vee E_2 \vee \ldots \vee E_n\}(t) = 1 - \prod_{i=1}^{n} \left(1 - Pr\{E_i\}(t)\right) \qquad (2)$$

PNs have been used to evaluate fault trees. For example, in [53]–[55], classical fault trees have been converted into PNs for reliability analysis. Furthermore, PN-based DFT quantification methods have been proposed in [9], [56], [57]. In these approaches, DFTs are transformed into PNs. In the transformation process, each DFT node is translated to a particular sub-PN with a 'place' indicating the status of the DFT node. Places are therefore used to indicate the state of the system, while timed transitions, symbolised by white rectangles, represent random faults and immediate transitions, symbolised by black rectangles, indicate the propagation of failures. The firing rate of a timed transition is characterised by the failure rate of the component it is representing. For instance, the PN model of the failure behaviour of a non-repairable component is shown in Fig. 2. In this model, the places *'x.up'* and *'x.dn'* represent the functional and non-functional state of the component x, respectively. At the beginning of system operation, the place *'x.up'* contains a token (the black dot symbolises it), meaning the component is fully operational (i.e., operating as a new component). The timed transition *'x.f'* is characterized by the time to failure distribution of the component. Note that depending on the failure behaviour of a component, i.e., exponentially on non-exponentially distributed time to failure, a transition can be modelled accordingly. For instance, if the component has an exponentially distributed failure rate λ, then the probability of the transition *x.f* firing at time t is $1 - e^{-\lambda t}$. On the firing of the transition *x.f* the place *x.dn* will get a token, which will mark the occurrence of the basic event, i.e., failure of the corresponding component. PN models of the DFT's logic gates are shown in Fig. 3 and further details on how these transformations are made can be found in [9], [51].

III. A MODULAR APPROACH FOR DFT ANALYSIS

The flowchart of the proposed modular approach is shown in Fig. 4. Similar to the existing modularisation techniques, the proposed approach takes the system level DFTs as input and then identifies the independent sub-trees. Independent sub-trees are those which do not share any input among them. These sub-trees are then further classified into static and dynamic fault trees. Static sub-trees are solved using an algebraic solution by utilising equations (1) and (2). Separating static parts and solving them with algebraic solution reduces the computation time and complexity. On the other hand, unlike the existing modularisation approaches, dynamic sub-trees and trees with shared inputs are solved using the PN-based approach. In case of BEs having exponential failure distribution, the PN model will be converted to a reachability graph and can be solved through Markov theorem. If BEs obey non-exponential failure distribution, the PN-model will be simulated with a number of iterations and the result will be gained through Monte Carlo theorem. It should be noted that the proposed solution solves the DFT hierarchically and layer by layer in each separated module. Afterwards, these independent solutions are combined to achieve the solution of the system-level DFT.

A. MODULE IDENTIFICATION

Several algorithms have been developed for the identification of modules in fault trees [37], [58]–[63]. Among these, the algorithm proposed by Dutuit and Rauzy [37] is the simplest and most efficient one. This is a highly efficient linear time algorithm [38].

The basic idea of this algorithm is as follows: "*Let v be an internal event and t_1 and t_2 respectively the first and second dates of visits of v in a depth-first left most traversal of the fault tree. Then v is a module iff none of its descendents is visited before t_1 and after t_2 during the traversal*" [37]. We used this algorithm to identify the independent sub-trees (modules) in the original fault tree. To compute and facilitate the integration of the sub-tree solutions, we used two flags: *TypeofNode* to indicate whether the independent sub-tree is static or dynamic, and *Independent_Child* to indicate whether the current independent sub-tree contains other independent sub-trees. The latter flag helps further modularisation of sub-trees if they contain independent sub-trees.

B. RELIABILITY QUANTIFICATION THROUGH INTEGRATION

Once the independent sub-trees are identified, a recursive approach is used to solve independent sub-trees because an independent sub-tree may contain other independent sub-trees. The solutions of sub-trees at different levels are combined in the recursion process to obtain the probability of the top event. As static and dynamic gates can be combined in different ways to form different tree structures, we consider these distinct scenarios and provide ways of addressing each.

The first scenario is shown in Fig. 5. In this case, the top event (TE) is a static gate and input to this gate is the output of another static sub-tree. To solve this tree, at first, the static sub-tree is solved using an algebraic formula to obtain the probability of the sub-tree. This probability is used directly as an input to the parent tree and the parent tree is then solved using an algebraic formula to calculate the TE probability.

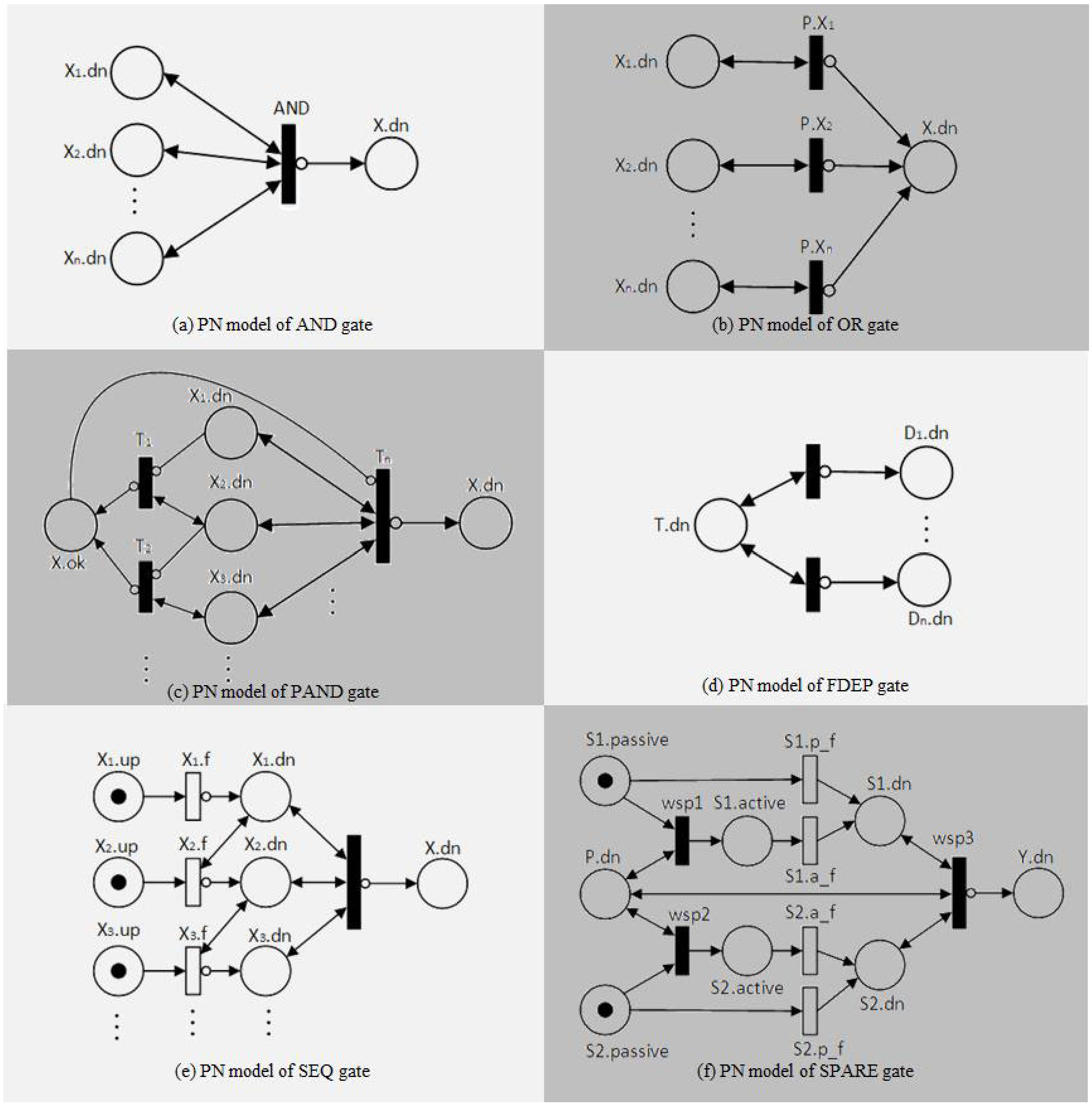


FIGURE 3. PN models of Boolean and dynamic gates.

Fig. 6 shows the second scenario, where the TE of the trees is a dynamic gate and the sub-trees are static gates. To solve these trees, firstly, the static sub-trees are solved using mathematical formulas to obtain their probability. As probability values cannot be used directly to quantify dynamic gates, we obtain the failure rate from the probability value. After that, the static sub-tree is replaced by a single node and a PN model of the dynamic gate is created for evaluation. The PN model can be evaluated in many different ways to obtain the unreliability of the dynamic module. For instance, in the PN model, if all the timed transitions are exponentially distributed, then the PN model can be evaluated by evaluating an underlying Markov model. On the other hand, if the PN model contains non-exponentially distributed timed transitions, then simulation like Monte Carlo simulation can be used for evaluation. In this paper, we evaluated PNs containing exponentially distributed transitions by converting them to reachability graph and then solved it via Markov theorem. On the other hand, we used Monte Carlo simulation to evaluate PNs having non-exponentially distributed timed transitions. More details about this process are provided in section III-C.

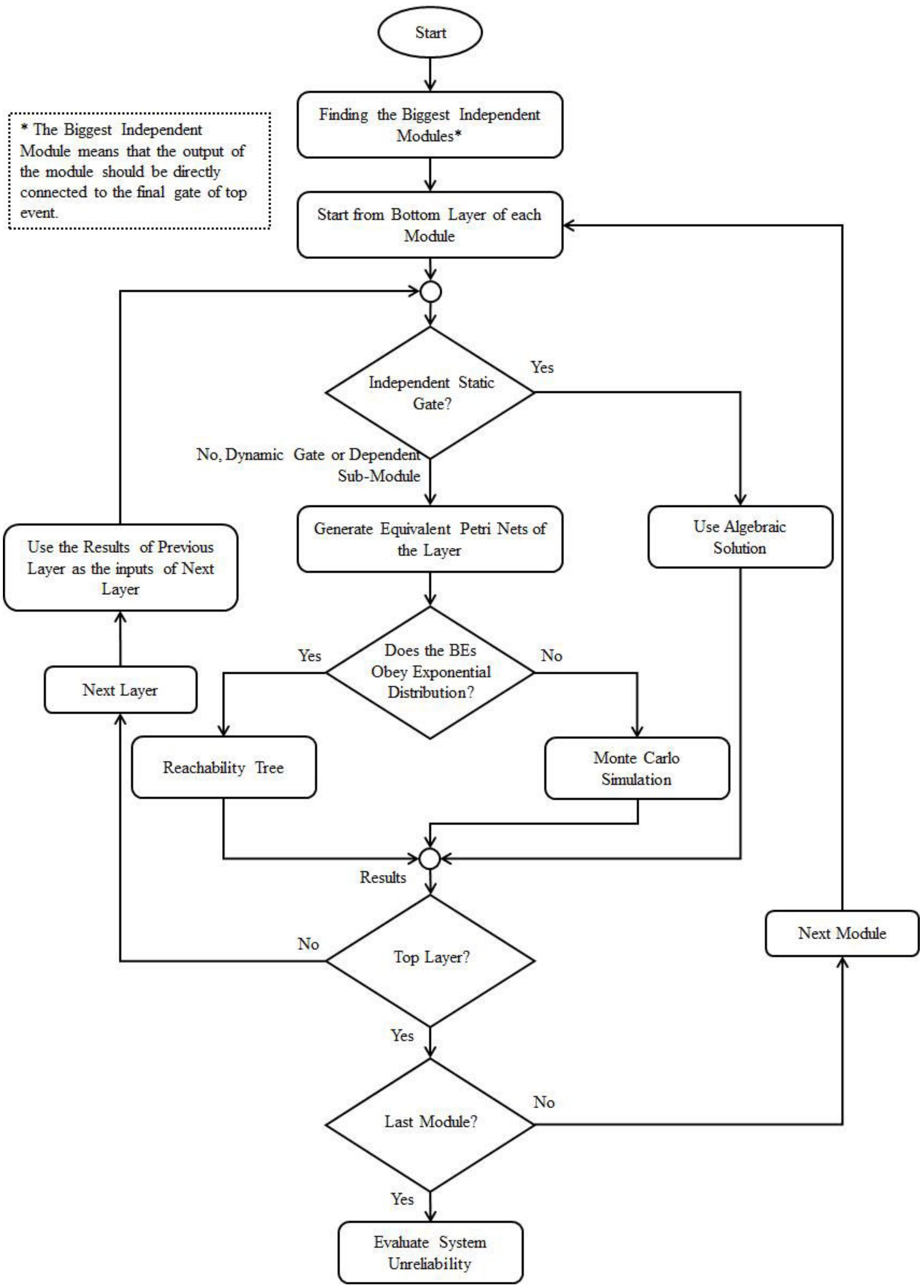

Start
Finding the Biggest Independent Modules*
* The Biggest Independent Module means that the output of the module should be directly connected to the final gate of top event.
Start from Bottom Layer of each Module
Independent Static Gate?
Yes
No, Dynamic Gate or Dependent Sub-Module
Generate Equivalent Petri Nets of the Layer
Use Algebraic Solution
Use the Results of Previous Layer as the inputs of Next Layer
Does the BEs Obey Exponential Distribution?
Yes
No
Next Layer
Reachability Tree
Monte Carlo Simulation
Results
No
Top Layer?
Next Module
Yes
No
Last Module?
Yes
Evaluate System Unreliability

FIGURE 4. Flowchart of the proposed hybrid modularised approach.

The third scenario (see Fig. 7) is the opposite of scenario 2. In this case, the TE of the trees is a static gate and the sub-tree is a dynamic tree. Therefore, to solve this tree, we first solve the dynamic sub-tree using the PN-based method to obtain the probability of the dynamic tree. Subsequently, a single node is used to replace the sub-tree and the probability value

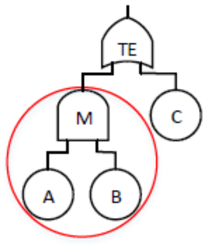


FIGURE 5. Hierarchically arranged static trees.

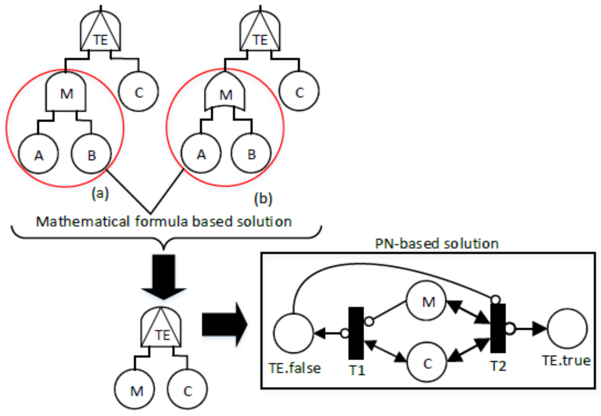


FIGURE 6. DFT having dynamic gate as top event with an independent static sub-tree.

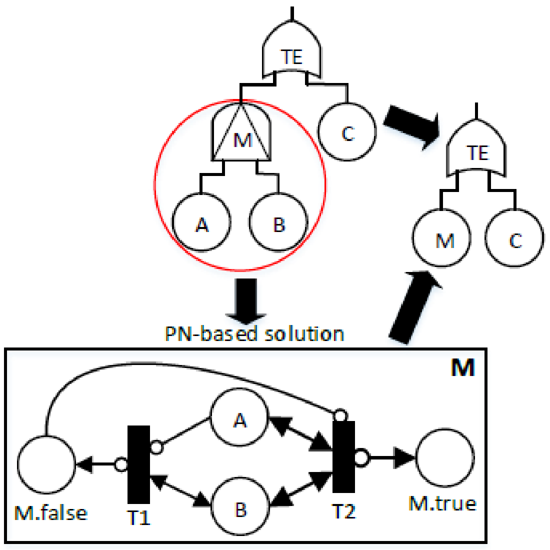


FIGURE 7. DFT having static gate as top event with an independent dynamic sub-tree.

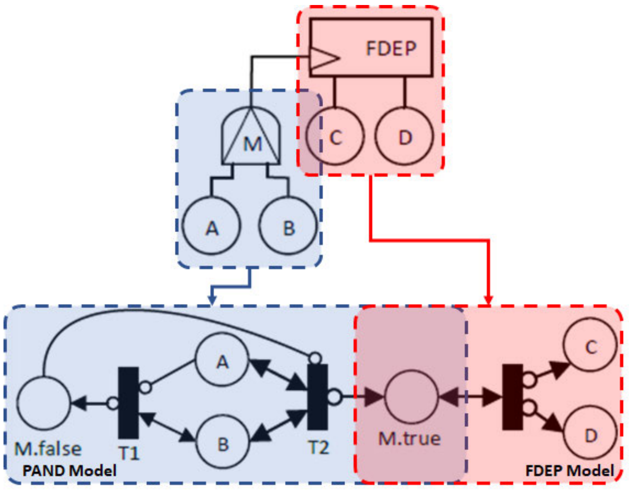


FIGURE 8. DFT with cascaded dynamic gates.

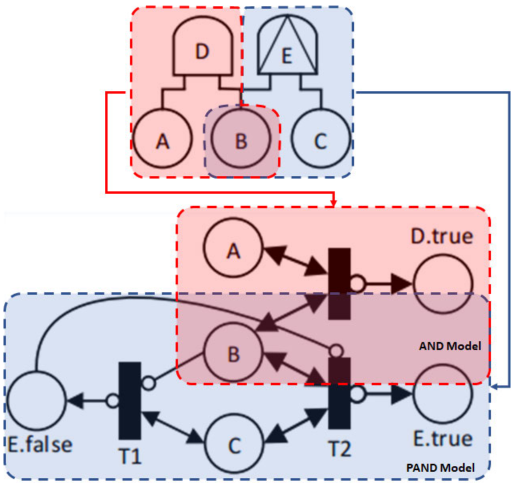


FIGURE 9. DFT with shared events.

of this node is used directly as an input to the evaluation of the parent tree. As the parent tree is static, it can be evaluated algebraically using equations (1) or (2).

In the fourth scenario, as can be seen from Fig. 8, two dynamic gates are arranged hierarchically, i.e., the output of one dynamic gate is an input to another dynamic gate. We solve this DFT by converting it to a PN model directly. However, it is possible to solve them one at a time using the PN model, i.e., solving the child tree first and then the parent tree. Since a PN model is needed to address both parent and child, we use a single PN model to solve the whole tree in one go. Fig. 9 shows a case where an input is shared between two logic gates, thus making them dependent. In such cases, we use the PN approach to solve the tree.

C. REACHABILITY SOLUTION

To demonstrate the quantitative solution of the proposed method, a simple DFT consisting of a PAND and a POR gate with three basic events is considered as shown in Fig. 10.

The DFT of Fig. 10 can be converted to an equivalent PN as illustrated in Fig. 11. It is assumed that the PN models of all gates in DFT are bounded. Therefore, from the PN model of each gate, a reachability graph can be obtained. By removing immediate transitions the reachability graph will be converted to a Markov process. Interested readers are referred to [64] to find more information about how this can be done.

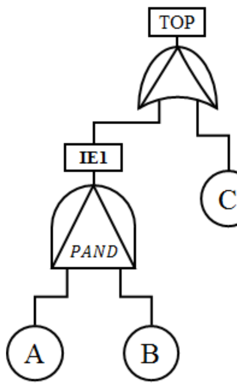


FIGURE 10. A Simple DFT with a PNAD and a POR gate.

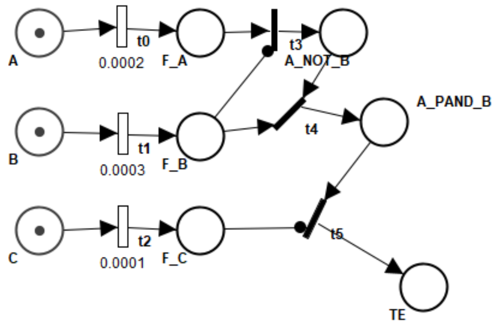
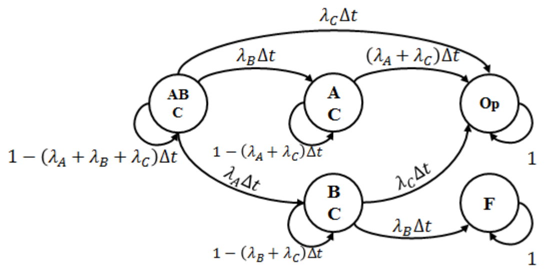


FIGURE 11. PN model for a simple DFT of Figure 10.


$\lambda_C\Delta t$
$\lambda_B\Delta t$
$(\lambda_A+\lambda_C)\Delta t$
AB C
A C
Op
B C
F
$1-(\lambda_A+\lambda_B+\lambda_C)\Delta t$
$1-(\lambda_A+\lambda_C)\Delta t$
$\lambda_A\Delta t$
$\lambda_C\Delta t$
$\lambda_B\Delta t$
$1-(\lambda_B+\lambda_C)\Delta t$
1
1
FIGURE 12. Markov process model for a simple DFT of Figure 10.

For the PN model presented in Fig. 11, a Markov process of Fig. 12 can be achieved. Note that as the Markov process of Fig. 12 is obtained by optimising the original reachability graph of the PN model of Fig. 11, it would not be possible to find a one-to-one correspondence between the two models. However, in Fig.12, λ_A, λ_B, and λ_C correspond to the failure rates of events A, B, and C, respectively, which are denoted by the timed transitions (white rectangles) with values 0.0002, 0.0003, and 0.0001, respectively in Fig. 11.

For the Markov process of Fig. 12, the equations can be formed as eq.(3).

$$P(t+\Delta t) = MP(t) \tag{3}$$

where P is the "states vector" denoted by eq.(4) and M is the discrete state transition matrix denoted by eq.(5).

$$P(t) = \left[P_1(t), P_2(t), P_3(t), P_{op}(t), P_F(t)\right] \tag{4}$$

$$M = \begin{bmatrix} 1-\xi\Delta t & 0 & 0 & 0 & 0 \\ \lambda_A\Delta t & 1-\varphi\Delta t & 0 & 0 & 0 \\ \lambda_B\Delta t & 0 & 1-\psi\Delta t & 0 & 0 \\ 0 & \lambda_B\Delta t & \lambda_A\Delta t & 1 & 0 \\ \lambda_C\Delta t & \lambda_C\Delta t & \lambda_C\Delta t & 0 & 1 \end{bmatrix} \tag{5}$$

where $\xi = (\lambda_A+\lambda_B+\lambda_C)$, $\psi = (\lambda_A+\lambda_C)$, *and* $\varphi = (\lambda_B+\lambda_C)$. Eq. (3) can be recursively solved if the initial probability vector $P(0)$ is known. The result at times $n\Delta t$ is given by eq.(6).

$$P(n\Delta t) = M^n P(0) \tag{6}$$

Eq. (6) in its continuous form is written as (7).

$$\dot{P}(t) = AP(0) \tag{7}$$

where A is the continuous Markov transition matrix in the form of (8).

$$A = \frac{\partial M}{\partial \Delta t} = \begin{bmatrix} -\lambda_A-\lambda_B-\lambda_C & 0 & 0 & 0 & 0 \\ \lambda_A & -\lambda_B-\lambda_C & 0 & 0 & 0 \\ \lambda_B & 0 & -\lambda_A-\lambda_C & 0 & 0 \\ 0 & \lambda_B & \lambda_A & 0 & 0 \\ \lambda_C & \lambda_C & \lambda_C & 0 & 0 \end{bmatrix} \tag{8}$$

Solving (7) gives the probability of system states at any time t and the unreliability of the system can be calculated through the probability of failed state.

$$U(t) = P_F(t) = \frac{\lambda_B\left(\lambda_A+\lambda_B e^{-(\lambda_A+\lambda_B+\lambda_C)t}+\lambda_C e^{-(\lambda_A+\lambda_B+\lambda_C)t}\right)}{(\lambda_B+\lambda_C)(\lambda_A+\lambda_B+\lambda_C)} - \frac{\lambda_B(\lambda_A+\lambda_B+\lambda_C)e^{-(\lambda_B+\lambda_C)t}}{(\lambda_B+\lambda_C)(\lambda_A+\lambda_B+\lambda_C)} \tag{9}$$

In the case of having non-exponential failure distribution, the proposed approach will use the combined Monte Carlo Simulation and PN. Consider N is the number of total iterations in which the Petri Net model can be simulated. The time to failure can be calculated for each timed arc transition in the model based on its probability distribution. For example, in the case of having exponential failure distribution, the time for arc transition can be calculated through an exponential distribution.

$$P_A = e^{-\lambda_A t} \rightarrow t_A = -ln\frac{(1-rand)}{\lambda_A} \tag{10}$$

where rand is the uniformly generated random number and λ_A is the failure rate of event A. The unreliability of the system can be calculated by dividing the number of time that a token reaches the place denoting the TE by the total number of iterations. For Weibull distribution, it is also possible to use inverse distribution. In MATLAB "wblinv" can be used.

D. CRITICALITY ANALYSIS

In FTA, criticality analysis plays an important role by identifying the critical events causing the top event of a fault tree. Criticality is measured in terms of the relative contributions of the events to the occurrence of the TE. Different approaches such as Fussel-Vesely importance measures, Birnbaum importance measures (BIM), and Risk Reduction Worth (RRW) are available to perform the criticality analysis [65]. For illustration, in this paper, we show how BIM can be used for identifying critical basic events using our proposed approach. Note that other approaches can also be used for this purpose.

The BIM of an event is calculated by taking the difference between the conditional TE probability given the occurrence or absence of that event. The event's occurrence and absence are represented by setting the basic event's probability as 1 and 0, respectively. Mathematically, the BIM of an event can be expressed as:

$$BIM(BE_i) = Pr(TE|Pr(BE_i = 1)) - Pr(TE|Pr(BE_i = 0)) \tag{11}$$

where $Pr(TE|Pr(BE_i = 1))$ and $Pr(TE|Pr(BE_i = 0))$ is the probability of the TE given that the probability of BE_i is 1 and 0, respectively.

Most of the existing modularisation approaches are not able to perform criticality analysis. This is because when modularisation is performed to replace a sub-tree using a single event, the basic events involved in the sub-tree are eliminated, thus are absent from the further analysis. For this reason, it is not possible to set the probability of non-existent events. The methodology proposed in this paper provides a way to perform criticality analysis event after modularisation.

Consider that we want to find the criticality of an event '*e*', which is part of an independent tree '*tr*'. In the general case, when we replace the sub-tree with a single event, the existence of event '*e*' will be lost. To facilitate the criticality analysis, when evaluating the sub-tree, firstly, we set event *e's* probability to the pre-specified value (either 0 or 1). After that, the sub-tree is evaluated and the probability is calculated. As a result, the calculated new node's probability reflects the effect of the change in event *e's* probability. In this way, the effect of the change in the probability of a basic event is calculated despite the basic event itself becoming non-existent.

IV. NUMERICAL EXAMPLES AND EVALUATION

In this section, five different DFTs are evaluated using the proposed method to illustrate its effectiveness. The first example of an abstract temporal fault tree (TFT) is shown in Fig. 13 and the failure rates and probabilities of the BEs of the TFT are shown in Table 2. In this TFT, the identified independent static modules are G1, G6, G3, G7, and G4. The independent dynamic modules are G2 and G8. These independent modules are evaluated according to the process described in section III. The results of these individual solutions are

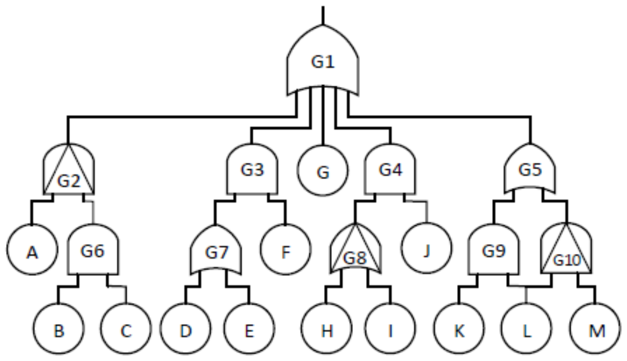


FIGURE 13. An example abstract temporal FT.

TABLE 2. Failure rate and probability of the basic events of the TFT in Fig. 13.

Basic Events	Failure rate/hour (λ)	Failure Probability
A	0.0000105852412	0.05155
B	0.0000048400967	0.02391
C	0.0000136150871	0.06581
D	0.0000050820240	0.02509
E	0.0000127413734	0.06172
F	0.0000012297731	0.00613
G	0.0000049364225	0.02438
H	0.0000209321521	0.09937
I	0.0000064757148	0.03186
J	0.0000026433921	0.01313
K	0.0000035981739	0.01783
L	0.0000151236686	0.07283
M	0.0000142433542	0.06874

combined to obtain the TE (G1) probability. According to the proposed modularisation technique, the TE probability of the TFT is 0.0289564056. To compare the result, we evaluated this TFT using PN approach without modularisation and the TE probability obtained was 0.0289692416. As can be seen, these two values start differing from the fifth digit after the decimal point. That means the result approximated by the proposed modularisation approach is very to close the solution provided by typical PN-based approach.

The second DFT, as seen in Fig. 14, was selected from [22], [66]. In [66], an inclusion-exclusion based formula is used to achieve the TE probability through cut-sets considering repeated events. Yuge and Yanagi evaluated this DFT using their proposed approach as well as using the Galileo tool and Monte Carlo Simulation by setting the failure rates of the BEs as 0.01 h^{-1}. The same DFT was evaluated in [67] under the same setting using a semi-Markov process-based approach. In [67], Aslansefat *et al.* evaluated this DFT through a hierarchical procedure in which starting from BE level, the cumulative failure distribution function of each gate was calculated and used as an input of next layer gates. This process was repeated to obtain the cumulative failure distribution function of the TE. The results produced

TABLE 3. Comparison between the results obtained by different approaches.

Time (hours)	Galileo	Yuge's method [66]	Aslansefat's method [67]	Monte Carlo	Proposed method
0	0.00000	0.00000	0.00000	0.00000	0.00000
50	0.21418	0.25653	0.19526	0.25647	0.21535
100	0.49318	0.59960	0.45148	0.59970	0.49635
150	0.68751	0.80196	0.64738	0.80212	0.69372
200	0.81010	0.90114	0.78226	0.90120	0.81890
250	0.88519	0.94864	0.86873	0.94869	0.89395
300	0.93066	0.97213	0.92168	0.97215	0.93786

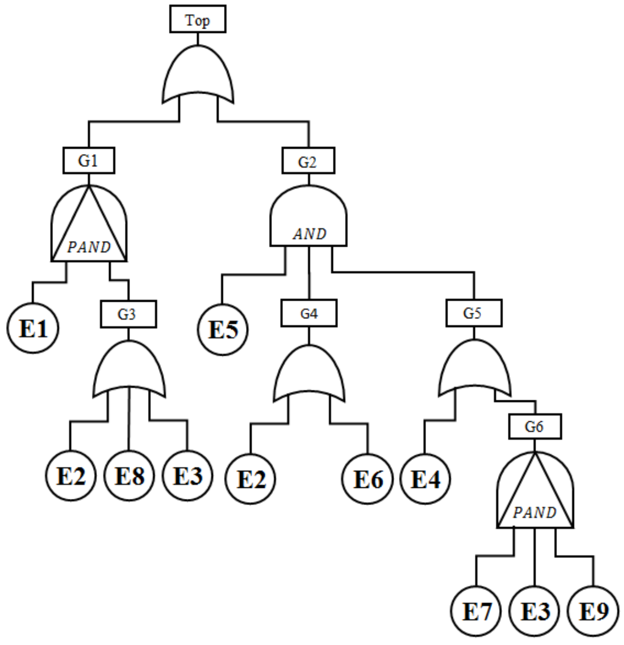


FIGURE 14. An example DFT [22], [66].

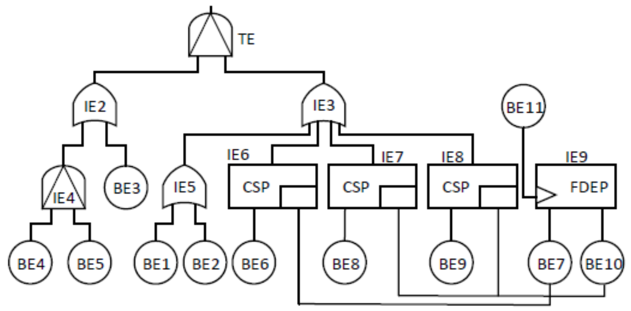


FIGURE 15. SAP2 DFT [43].

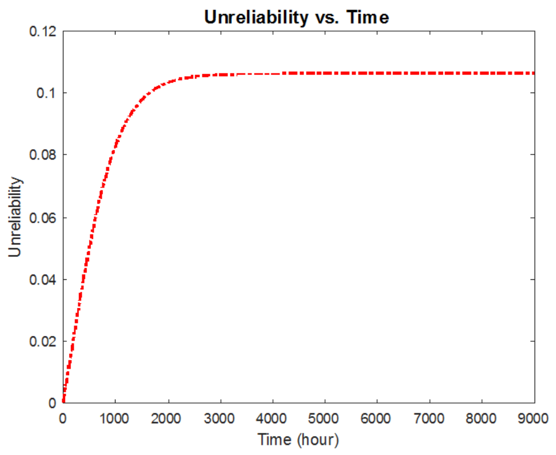


FIGURE 16. Numerical results of DFT illustrated in figure 15.

by different approaches were compared in [66], [67] and it was argued that the results produced by Galileo tool were more accurate because of the tool's ability to provide more exact closed-form solution. In this paper, for comparison, we have evaluated the DFT of Fig.14 using the proposed modularisation approach under the same setting as used in other mentioned papers. Table 3 provides a comparison of the probability of the TE for mission times ranging from 0 to 300 hours. From the comparison, it can be seen that the results produced by the proposed approach are almost the same as the results produced by the Galileo tool. Therefore, it can be said that the proposed modularisation is effective in evaluating DFTs without losing accuracy.

As we know the exact results belong to Galileo. To compare the different results provided in table 3, the Mean Absolute Percentage Error (MAPE) of each method with regards to the reference method (Galileo) has been used in table 4. It can be seen the proposed method has less deviation percentage in comparison to the others such as Yuge's, Aslansefat's or Monte Carlo based methods.

The DFTs in the examples seen so far contains only a limited number of dynamic gates, e.g., PAND and POR gates. To illustrate the evaluation of DFTs with a wider number of dynamic gates, the DFT shown in Fig. 15 is selected. This is the DFT of a subsystem of a real industrial plant, taken from [43]. The quantitative parameters for the BEs of this DFT are shown in Table 5. Considering the mission time 0 to 9000 hours, the unreliability of the SAP2 system can be achieved as shown in Fig. 16, which is in line with the original study presented in [43].

The fourth example used in this paper is the TFT (see Fig. 17) of an aircraft fuel distribution system taken from [11]. Table 6 shows the numerical data for the BEs of this TFT. This TFT is the most complex compared to the other DFTs shown earlier. It presents many complex relationships and dependencies among the events at different levels of the tree.

TABLE 4. A comparison of Mean Absolute Percentage Error (MAPE) of existing methods with regards to Galileo's results.

	Yuge's method [66]	Aslansefat's method [67]	Monte Carlo	Proposed method
Mean Absolute Percentage Error (MAPE)	13.47	4.89	13.48	0.82

TABLE 5. Failure rates of the basic events of the DFT in Fig. 15.

Basic Events	Failure rate/hour (λ)
BE1	0.00010
BE2	0.00091
BE3	0.00010
BE4	0.00017
BE5	0.00075
BE6	0.00091
BE7	0.00450
BE8	0.00086
BE9	0.00045
BE10	0.00790
BE11	0.00015

TABLE 6. Failure rate, failure probability, and criticality ranking of the basic events of the TFT in Fig. 17.

Basic Events	Failure rate/hour (λ)	Rank
I-SCP	5.84267E-5	8
I-CSP	5.84267E-5	5
I-SOV	1.65633E-3	2
I-SIV	1.65633E-3	4
I-CSV	1.65633E-3	1
I-SCV	1.65633E-3	3
I-CRL	2.21127E-6	7
I-HiSOF	4.06861E-5	9
I-HiSIF	4.06861E-5	10
I-HiSEF	4.06861E-5	6
I-SIL	1.65633E-3	4
I-SOL	3.31774E-5	11

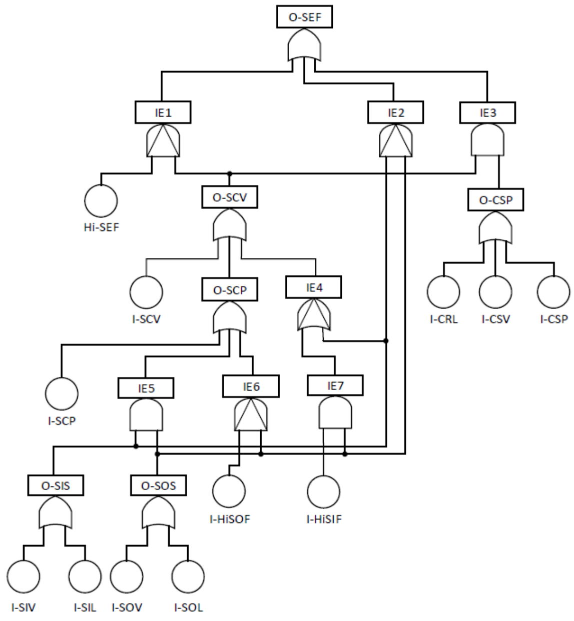


FIGURE 17. TFT of the aircraft of the fuel distribution system.

Using a laptop with a 64-bit Intel core i7 processor at 2.8 GHz (8 CPUs) and 16 GB RAM, an attempt was made to quantify this TFT using the PN-based approach proposed in [11]. The approach failed to provide a solution because after generating a certain number of states, due to the state space explosion the approach could not proceed further. However, with the help of the High-Performance Computing (HPC) facility at the University of Hull, we were able to find a solution to this TFT. Across 10 executions, finding a solution required, on an average, 747 seconds. The modularisation technique proposed in this paper was able to avoid the state space explosion problem without using the HPC facility and provided a solution in a matter of seconds. Moreover, for 100 hours of mission time, the values estimated by both the PN and modularisation approaches for the TE probability are 0.049. The criticality of the BEs are also calculated, the events are ranked based on their criticality and the ranking is shown in Table 6. This ranking agrees with the ranking suggested in [11] by the PN-based approach.

To reveal the capabilities of the proposed method in the case of having non-exponential failure distribution, an example of DFT consists of PAND, AND and OR gates with ten basic events (seven exponentially distributed BEs and three BEs with Weibull failure distribution) has been considered. It is assumed that three BEs including I, J and K have Weibull failure distribution with the scale factor of $\omega = 20$ and the shape factor of $\alpha = 0.1$ that can be formulated as follows:

$$P = e^{-(t/\omega)^{\alpha}} \tag{12}$$

The rest of the BEs obeys exponential failure distribution with the failure rate of $\lambda_A = 0.0110$, $\lambda_B = 0.0120$, $\lambda_C = 0.0130$, $\lambda_D = 0.0140$, $\lambda_E = 0.0150$, $\lambda_H = 0.0011$, and $\lambda_L = 0.0015$ failure per hour, respectively. Figure 18 illustrates the DFT of the elucidated example.

Considering the mission time of zero to three hundred hours, the unreliability of the system can be obtained as shown in figure 19. In this figure, the blue curve is the result that can be obtained from Algebraic solution provided by [28] and red circles are the results obtained from the proposed method in which a combination of PN and Monte Carlo simulation with $10e6$ iterations has been used. As can be seen, both solutions have the same results.

From the results obtained by the modularised approach for five different examples, we can see that the proposed modularisation technique estimates results without unacceptable loss of accuracy while making a major improvement in the execution performance. Regarding the computation

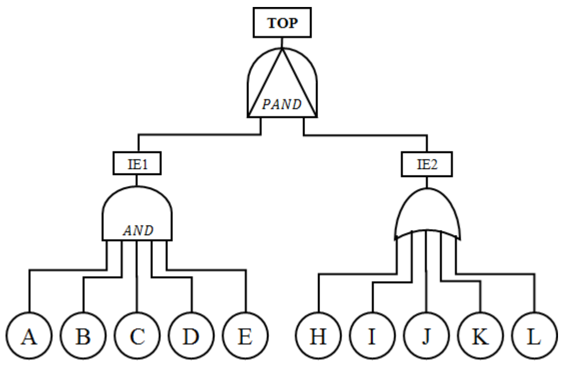


FIGURE 18. An example DFT consists of exponential and Weibull BEs.

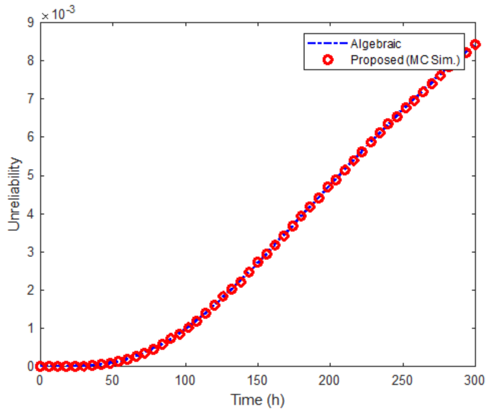


FIGURE 19. A comparison of the numerical unreliability results (Algebraic and proposed method) of the DFT shown in figure 18.

complexity of the proposed hybrid method, we can divide it into three parts, (A) Algebraic, (B) Petri Nets and (C) Monte Carlo Simulation. The computation complexity of probabilistic algebraic method that represents DFT operation can be achieved by $O(a)$ where a is the number of gates in a DFT [68]. As mentioned before, the Petri Nets model is converted to a reachability tree that is equivalent to a Markov chain. The cumulative required time to obtain the probability vector in a Markov model can be calculated as (13) assuming that $n \leq x[00]$ and $x[01]$.

$$O((x[00] + x[01])n) \tag{13}$$

where $x[00]$ stands for the number of non-zero transitions between functional states and $x[01]$ stands for the number of non-zero transitions between functional states and absorbing failure state(s). Therefore, [69] showed that the computation complexity of the model can be simplified to $O(n^2)$ where n can be determined by the total number of states. However, in the proposed method, DFTs are solved through the hierarchical model and the computation complexity is reduced to $O(k)$ where k is the number of gates in DFT [67]. This helps the proposed approach to avoid the issues associated with the state-space explosion. The computation complexity of Monte Carlo Simulation is $O(h)$ where h is the number of steps in each iteration. However, based on [70], in an optimized implementation, it can be reduced to $O(h^{-2})$. It is assumed that numerical discretization of the problem follows weak convergence, and quality obeyed by Euler–Maruyama and Milstein schemes.

In the proposed hybrid method, currently, repairable DFTs are not considered and it can be considered in the future research work. In addition, components are considered to have binary states (working and failed), thus complex failure behaviour of components with multiple modes of operation are not considered. In the future, research can be performed to find ways to incorporate such complex failure behaviour of components, for instance, using a concept like complex BEs. Currently, uncertainty with failure data is not considered. Exploring the ways of handling uncertainty could be a potential future research avenue.

V. CONCLUSION

In this paper, we have addressed the limitations of the existing modularisation techniques for DFT analysis by proposing a novel approach based on algebraic solutions and PN to quantify dynamic fault trees. We have outlined the differences and contribution by our approach against existing methods extensively, both in the related work and motivation section and as part of the detailed discussion of our method. The effectiveness of the approach is evaluated by applying it to five different DFTs. The comparison of the results approximated by the proposed approach with the existing approaches confirms that the modularisation approach yields comparable results with regards to accuracy. However, the time required by our modular approach to evaluate DFTs is significantly less than that required by the classical state-space based DFT evaluation approaches. Thus, the modular approach proposed in this paper allows analysis of much larger systems with complex inter-dependencies among the components than with the classical non-modular approaches. As the proposed approach allows the use of both constant failure probabilities and rates in the same analysis, it is therefore equally applicable to evaluate hardware, software and human operator failures. At present, we have considered PN as a solution technique to alleviate the limitations of Markov chain based modularisation techniques for DFT evaluation. There exist other solution techniques like semi-Markov process, SBDD, etc., which can relax the limitation of the Markov chain, therefore, in the future, it is worth exploring these alternative solutions as part of modularisation.

ACKNOWLEDGMENT

The authors would like to thank EDF Energy R&D UK Centre and AURA Innovation Centre for their support.

REFERENCES

[1] V. Calderaro, V. Lattarulo, A. Piccolo, and P. Siano, "Optimal switch placement by alliance algorithm for improving microgrids reliability," *IEEE Trans. Ind. Informat.*, vol. 8, no. 4, pp. 925–934, Nov. 2012.
[2] B. Zhao, H. Aydin, and D. Zhu, "On maximizing reliability of real-time embedded applications under hard energy constraint," *IEEE Trans. Ind. Informat.*, vol. 6, no. 3, pp. 316–328, Aug. 2010.
[3] S. Kabir, "An overview of fault tree analysis and its application in model based dependability analysis," *Expert Syst. Appl.*, vol. 77, pp. 114–135, Jul. 2017.

[4] J. B. Dugan, S. J. Bavuso, and M. A. Boyd, ''Dynamic fault-tree models for fault-tolerant computer systems,'' *IEEE Trans. Rel.*, vol. 41, no. 3, pp. 363–377, Sep. 1992.

[5] J. B. Dugan, S. J. Bavuso, and M. A. Boyd, ''Fault trees and Markov models for reliability analysis of fault-tolerant digital systems,'' *Rel. Eng. Syst. Saf.*, vol. 39, no. 3, pp. 291–307, Jan. 1993.

[6] H. Boudali, P. Crouzen, and M. Stoelinga, ''Dynamic fault tree analysis using input/output interactive Markov chains,'' in *Proc. 37th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Washington, DC, USA, Jun. 2007, pp. 708–717.

[7] H. Boudali, P. Crouzen, and M. Stoelinga, ''A compositional semantics for dynamic fault trees in terms of interactive Markov chains,'' in *Proc. Int. Symp. Automat. Technol. Verification Anal.* Berlin, Germany: Springer, 2007, pp. 441–456.

[8] H. Boudali, P. Crouzen, and M. Stoelinga, ''A rigorous, compositional, and extensible framework for dynamic fault tree analysis,'' *IEEE Trans. Depend. Sec. Comput.*, vol. 7, no. 2, pp. 128–143, Apr. 2010.

[9] D. Codetta-Raiteri, ''The conversion of dynamic fault trees to stochastic Petri nets, as a case of graph transformation,'' *Electron. Notes Theor. Comput. Sci.*, vol. 127, no. 2, pp. 45–60, Mar. 2005.

[10] T. P. K. Nguyen, J. Beugin, and J. Marais, ''Method for evaluating an extended fault tree to analyse the dependability of complex systems: Application to a satellite-based railway system,'' *Rel. Eng. Syst. Saf.*, vol. 133, pp. 300–313, Jan. 2015.

[11] S. Kabir, M. Walker, and Y. Papadopoulos, ''Dynamic system safety analysis in HiP-HOPS with Petri nets and Bayesian networks,'' *Saf. Sci.*, vol. 105, pp. 55–70, Jun. 2018.

[12] H. Song and E. Schnieder, ''Evaluating fault tree by means of colored Petri nets to analyze the railway system dependability,'' *Saf. Sci.*, vol. 110, pp. 313–323, Dec. 2018.

[13] H. Boudali and J. B. Dugan, ''A continuous-time Bayesian network reliability modeling, and analysis framework,'' *IEEE Trans. Rel.*, vol. 55, no. 1, pp. 86–97, Mar. 2006.

[14] S. Montani, L. Portinale, A. Bobbio, and D. Codetta-Raiteri, ''Radyban: A tool for reliability analysis of dynamic fault trees through conversion into dynamic Bayesian networks,'' *Rel. Eng. Syst. Saf.*, vol. 93, no. 7, pp. 922–932, Jul. 2008.

[15] D. Marquez, M. Neil, and N. Fenton, ''Solving dynamic fault trees using a new hybrid Bayesian network inference algorithm,'' in *Proc. 16th Medit. Conf. Control Automat.*, Jun. 2008, pp. 609–614.

[16] L. Xing, O. Tannous, and J. B. Dugan, ''Reliability analysis of non-repairable cold-standby systems using sequential binary decision diagrams,'' *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 42, no. 3, pp. 715–726, May 2012.

[17] D. Ge, M. Lin, Y. Yang, R. Zhang, and Q. Chou, ''Quantitative analysis of dynamic fault trees using improved sequential binary decision diagrams,'' *Rel. Eng. Syst. Saf.*, vol. 142, pp. 289–299, Oct. 2015.

[18] S. Khan, J.-P. Katoen, M. Volk, and M. Bouissou, ''Synergizing reliability modeling languages: BDMPs without repairs and DFTs,'' in *Proc. IEEE 24th Pacific Rim Int. Symp. Dependable Comput. (PRDC)*, Dec. 2019, pp. 266–275.

[19] P.-Y. Piriou, J.-M. Faure, and J.-J. Lesage, ''Finding the minimal cut sequences of dynamic, repairable, and reconfigurable systems from generalized Boolean logic driven Markov process models,'' *Proc. Inst. Mech. Eng., O, J. Risk Rel.*, pp. 1–12, Feb. 2019.

[20] S. Distefano and A. Puliafito, ''Dependability evaluation with dynamic reliability block diagrams and dynamic fault trees,'' *IEEE Trans. Depend. Sec. Comput.*, vol. 6, no. 1, pp. 4–17, Jan. 2009.

[21] S. Distefano and A. Puliafito, ''Reliability and availability analysis of dependent-dynamic systems with DRBDs,'' *Rel. Eng. Syst. Saf.*, vol. 94, no. 9, pp. 1381–1393, Sep. 2009.

[22] P. Zhu, J. Han, L. Liu, and M. J. Zuo, ''A stochastic approach for the analysis of fault trees with priority AND gates,'' *IEEE Trans. Rel.*, vol. 63, no. 2, pp. 480–494, Jun. 2014.

[23] F. Chiacchio, J. I. Aizpurua, L. Compagno, and D. D'Urso, ''SHyFTOO, an object-oriented Monte Carlo simulation library for the modeling of stochastic hybrid fault tree automaton,'' *Expert Syst. Appl.*, vol. 146, May 2020, Art. no. 113139.

[24] F. Chiacchio, J. I. Aizpurua, L. Compagno, S. M. Khodayee, and D. D'Urso, ''Modelling and resolution of dynamic reliability problems by the coupling of simulink and the stochastic hybrid fault tree object oriented (SHyFTOO) library,'' *Information*, vol. 10, no. 9, p. 283, Sep. 2019.

[25] F. Chiacchio, A. Iacono, L. Compagno, and D. D'Urso, ''A general framework for dependability modelling coupling discrete-event and time-driven simulation,'' *Rel. Eng. Syst. Saf.*, vol. 199, Jul. 2020, Art. no. 106904.

[26] K. D. Rao, V. Gopika, V. V. S. S. Rao, H. S. Kushwaha, A. K. Verma, and A. Srividya, ''Dynamic fault tree analysis using Monte Carlo simulation in probabilistic safety assessment,'' *Rel. Eng. Syst. Saf.*, vol. 94, no. 4, pp. 872–883, Apr. 2009.

[27] G. Manno, F. Chiacchio, L. Compagno, D. D'Urso, and N. Trapani, ''MatCarloRe: An integrated FT and Monte Carlo Simulink tool for the reliability assessment of dynamic fault tree,'' *Expert Syst. Appl.*, vol. 39, no. 12, pp. 10334–10342, 2012.

[28] G. Merle, J.-M. Roussel, J.-J. Lesage, and A. Bobbio, ''Probabilistic algebraic analysis of fault trees with priority dynamic gates and repeated events,'' *IEEE Trans. Rel.*, vol. 59, no. 1, pp. 250–261, Mar. 2010.

[29] G. Merle, J.-M. Roussel, and J.-J. Lesage, ''Quantitative analysis of dynamic fault trees based on the structure function,'' *Qual. Rel. Eng. Int.*, vol. 30, no. 1, pp. 143–156, Feb. 2014.

[30] K. J. Sullivan, J. B. Dugan, and D. Coppit, ''The galileo fault tree analysis tool,'' in *29th Annu. Int. Symp. Fault-Tolerant Comput. Dig. Papers*, 1999, pp. 232–235.

[31] M. Batteux, T. Prosvirnova, A. Rauzy, and L. Kloul, ''The AltaRica 3.0 project for model-based safety assessment,'' in *Proc. 11th IEEE Int. Conf. Ind. Informat. (INDIN)*, Jul. 2013, pp. 741–746.

[32] K. Aslansefat, S. Kabir, Y. Gheraibia, and Y. Papadopoulos, ''Dynamic fault tree analysis: State-of-the-art in modeling, analysis, and tools,'' in *Reliability Management and Engineering: Challenges and Future Trends*. Boca Raton, FL, USA: CRC Press, 2020, ch. 4, pp. 73–111.

[33] F. A. Patterson-Hine and J. B. Dugan, ''Modular techniques for dynamic fault tree-analysis,'' in *Proc. Annu. Rel. Maintainability Symp.*, 1992, pp. 363–369.

[34] L. L. Pullum and J. B. Dugan, ''Fault tree models for the analysis of complex computer-based systems,'' in *Proc. Annu. Rel. Maintainability Symp.*, 1996, pp. 200–207.

[35] J. B. Dugan, B. Venkataraman, and R. Gulati, ''DIFtree: A software package for the analysis of dynamic fault tree models,'' in *Proc. Annu. Rel. Maintainability Symp.*, 1997, pp. 64–70.

[36] R. E. Bryant, ''Graph-based algorithms for Boolean function manipulation,'' *IEEE Trans. Comput.*, vol. C-35, no. 8, pp. 677–691, Aug. 1986.

[37] Y. Dutuit and A. Rauzy, ''A linear-time algorithm to find modules of fault trees,'' *IEEE Trans. Rel.*, vol. 45, no. 3, pp. 422–425, Sep. 1996.

[38] R. Gulati and J. B. Dugan, ''A modular approach for analyzing static and dynamic fault trees,'' in *Proc. Annu. Rel. Maintainability Symp.*, 1997, pp. 57–63.

[39] A. Anand and A. K. Somani, ''Hierarchical analysis of fault trees with dependencies, using decomposition,'' in *Proc. Annu. Rel. Maintainability Symp.*, 1998, pp. 69–75.

[40] R. Manian, J. B. Dugan, D. Coppit, and K. J. Sullivan, ''Combining various solution techniques for dynamic fault tree analysis of computer systems,'' in *Proc. 3rd IEEE Int. High-Assurance Syst. Eng. Symp.*, Washington, DC, USA, 1998, pp. 21–28.

[41] C.-Y. Huang and Y.-R. Chang, ''An improved decomposition scheme for assessing the reliability of embedded systems by using dynamic fault trees,'' *Rel. Eng. Syst. Saf.*, vol. 92, no. 10, pp. 1403–1412, Oct. 2007.

[42] O. Yevkin, ''An improved modular approach for dynamic fault tree analysis,'' in *Proc. Annu. Rel. Maintainability Symp.*, Jan. 2011, pp. 1–5.

[43] F. Chiacchio, M. Cacioppo, D. D'Urso, G. Manno, N. Trapani, and L. Compagno, ''A Weibull-based compositional approach for hierarchical dynamic fault trees,'' *Rel. Eng. Syst. Saf.*, vol. 109, pp. 45–52, Jan. 2013.

[44] R. Zurawski and M. Zhou, ''Petri nets and industrial applications: A tutorial,'' *IEEE Trans. Ind. Electron.*, vol. 41, no. 6, pp. 567–583, 1994.

[45] S. Kabir and Y. Papadopoulos, ''Applications of Bayesian networks and Petri nets in safety, reliability, and risk assessments: A review,'' *Saf. Sci.*, vol. 115, pp. 154–175, Jun. 2019.

[46] C. Fecarotti, J. Andrews, and R. Chen, ''A Petri net approach for performance modelling of polymer electrolyte membrane fuel cell systems,'' *Int. J. Hydrogen Energy*, vol. 41, no. 28, pp. 12242–12260, 2016.

[47] B. Le and J. Andrews, ''Petri net modelling of bridge asset management using maintenance-related state conditions,'' *Struct. Infrastruct. Eng.*, vol. 12, no. 6, pp. 730–751, Jun. 2016.

[48] S. Bernardi, J. Campos, and J. Merseguer, ''Timing-failure risk assessment of UML design using time Petri net bound techniques,'' *IEEE Trans. Ind. Informat.*, vol. 7, no. 1, pp. 90–104, Feb. 2011.

[49] V. Volovoi, ''Modeling of system reliability Petri nets with aging tokens,'' *Rel. Eng. Syst. Saf.*, vol. 84, no. 2, pp. 149–161, May 2004.

[50] W. Reisig, *Petri Nets: An Introduction*, vol. 4. Berlin, Germany: Springer, 2012.

[51] S. Kabir, M. Yazdi, J. I. Aizpurua, and Y. Papadopoulos, "Uncertainty-aware dynamic reliability analysis framework for complex systems," *IEEE Access*, vol. 6, pp. 29499–29515, 2018.

[52] S. Kabir, M. Walker, Y. Papadopoulos, E. Rüde, and P. Securius, "Fuzzy temporal fault tree analysis of dynamic systems," *Int. J. Approx. Reasoning*, vol. 77, pp. 20–37, Oct. 2016.

[53] G. S. Hura and J. W. Atwood, "The use of Petri nets to analyze coherent fault trees," *IEEE Trans. Rel.*, vol. R-37, no. 5, pp. 469–474, Dec. 1988.

[54] M. Malhotra and K. S. Trivedi, "Dependability modeling using Petri-nets," *IEEE Trans. Rel.*, vol. 44, no. 3, pp. 428–440, Sep. 1995.

[55] A. Bobbio, G. Franceschinis, R. Gaeta, and L. Portinale, "Exploiting Petri nets to support fault tree based dependability analysis," in *Proc. 8th Int. Workshop Petri Nets Perform. Models*, Zaragoza, Spain, 1999, pp. 146–155.

[56] X. Zhang, Q. Miao, X. Fan, and D. Wang, "Dynamic fault tree analysis based on Petri nets," in *Proc. 8th Int. Conf. Rel., Maintainability Saf.*, Chengdu, China, Jul. 2009, pp. 138–142.

[57] S. Kabir, M. Walker, and Y. Papadopoulos, "Quantitative evaluation of pandora temporal fault trees via Petri nets," *IFAC-PapersOnLine*, vol. 48, no. 21, pp. 458–463, 2015.

[58] Z. W. Birnbaum and J. D. Esary, "Modules of coherent binary systems," *J. Soc. Ind. Appl. Math.*, vol. 13, no. 2, pp. 444–462, Jun. 1965.

[59] P. Chatterjee, "Modularization of fault trees: A method to reduce the cost of analysis," *SIAM Rel. Fault Tree Anal.*, vol. 8, no. 4, pp. 101–137, 1975.

[60] A. Rosenthal, "Decomposition methods for fault tree analysis," *IEEE Trans. Rel.*, vol. R-29, no. 2, pp. 136–138, Jun. 1980.

[61] M. O. Locks, "Modularizing, minimizing, and interpreting the K&H fault-tree," *IEEE Trans. Rel.*, vol. R-30, no. 5, pp. 411–415, Dec. 1981.

[62] J. M. Wilson, "Modularizing and minimizing fault trees," *IEEE Trans. Rel.*, vol. R-34, no. 4, pp. 320–322, Oct. 1985.

[63] T. Kohda, E. J. Henley, and K. Inoue, "Finding modules in fault trees," *IEEE Trans. Rel.*, vol. 38, no. 2, pp. 165–176, Jun. 1989.

[64] A. Bobbio, "System modelling with Petri nets," in *Systems Reliability Assessment*. Dordrecht, The Netherlands: Springer, 1990, pp. 103–143.

[65] W. Vesely, J. Dugan, J. Fragola, J. Minarick, and J. Railsback, "Fault tree handbook with aerospace applications," NASA Office Saf. Mission Assurance, Washington, DC, USA, Tech. Rep. Version 1.1, 2002.

[66] T. Yuge and S. Yanagi, "Quantitative analysis of a fault tree with priority AND gates," *Rel. Eng. Syst. Saf.*, vol. 93, no. 11, pp. 1577–1583, Nov. 2008.

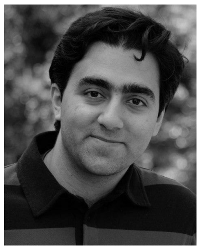
[67] K. Aslansefat and G.-R. Latif-Shabgahi, "A hierarchical approach for dynamic fault trees solution through semi-Markov process," *IEEE Trans. Rel.*, early access, Jul. 16, 2019, doi: 10.1109/TR.2019.2923893.

[68] J.-M. Fourneau and N. Pekergin, "A numerical analysis of dynamic fault trees based on stochastic bounds," in *Proc. Int. Conf. Quant. Eval. Syst.* Cham, Switzerland: Springer, 2015, pp. 176–191.

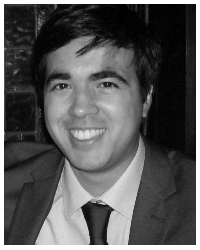
[69] G. Ciardo, R. A. Marie, B. Sericola, and K. S. Trivedi, "Performability analysis using semi-Markov reward processes," *IEEE Trans. Comput.*, vol. 39, no. 10, pp. 1251–1264, 1990.

[70] H. A. Lay, Z. Colgin, V. Reshniak, and A. Q. M. Khaliq, "On the implementation of multilevel Monte Carlo simulation of the stochastic volatility and interest rate model using multi-GPU clusters," *Monte Carlo Methods Appl.*, vol. 24, no. 4, pp. 309–321, Dec. 2018.

SOHAG KABIR received the Ph.D. degree in computer science and the M.Sc. degree in embedded systems from the University of Hull, U.K., in 2016 and 2012, respectively. He was a Research Associate with the Dependable Intelligent Systems (DEIS) Research Group, University of Hull. He has worked in EU projects on safety, including MAENAD and DEIS. He is currently working as an Assistant Professor with the Department of Computer Science, University of Bradford, U.K. His research interests include model-based safety assessment, probabilistic risk and safety analysis, fault tolerant computing, and stochastic modeling and analysis.



KOOROSH ASLANSEFAT (Member, IEEE) was born in Tehran, Iran, in 1989. He received the B.Sc. degree in marine electronic and communication engineering from Chabahar Maritime University, Chabahar, Iran, in 2011, and the M.Sc. degree in control engineering from Shahid Beheshti University, Tehran, Iran, in 2014. He is currently pursuing the Ph.D. degree with the University of Hull, Hull, U.K., working on data-driven reliability-centered evolutionary and automated maintenance for offshore wind farms. His main research interests are in Markov modeling, performance assessment, artificial intelligence, optimization, and stochastic modeling.



IOANNIS SOROKOS received the B.Sc. degree in computer science from the Athens University of Economics and Business, Greece, in 2011, and the M.Sc. and Ph.D. degrees in computer science from the University of Hull, U.K., in 2017 and 2013, respectively. He is currently a Postdoctoral Researcher with the University of Hull. His research interests include model-based dependability analysis and assurance, metaheuristic optimization, artificial intelligence, computer graphics, and computational game theory.

YIANNIS PAPADOPOULOS has pioneered work on model-based dependability assessment and evolutionary optimization of complex engineering systems known as Hierarchically Performed Hazard Origin and Propagation Studies (HiP-HOPS). He has coauthored EAST-ADL, an emerging automotive architecture description language working with Volvo, Honda, Continental, Honeywell, and DNV-GL, among others. He is currently a Professor and Leader of the Dependable Intelligent Systems Research Group, University of Hull. He is also actively involved in two technical committees of IFAC (TC 1.3 & 5.1). He is also working on new metaheuristics inspired by the hunting behavior of penguins and developing technologies for self-certification of cyber-physical and autonomous systems. He is interested in digital art and various aspects of philosophy and its interactions with science.

SAVAS KONUR (Member, IEEE) is currently a Reader in computer science with the University of Bradford. He has published in numerous prestigious journals, as well as leading conferences. His research interests mainly involve computational modeling, formal verification, high-performance stochastic simulations and machine learning with applications to real-time and safety critical systems, membrane computing, and systems and synthetic biology. He has led several research projects (funded by EPSRC, Innovate U.K., and EU Access Innovation), requiring a wide range of interdisciplinary collaborations.