Dynamic System Safety Analysis in HiP-HOPS with Petri Nets and Bayesian Networks

Sohag Kabir, Martin Walker, Yiannis Papadopoulos
School of Engineering and Computer Science, University of Hull, HU6 7RX, UK

Abstract: Dynamic systems exhibit time-dependent behaviours and complex functional dependencies amongst their components. Therefore, to capture the full system failure behaviour, it is not enough to simply determine the consequences of different combinations of failure events: it is also necessary to understand the order in which they fail. Pandora temporal fault trees (TFTs) increase the expressive power of fault trees and allow modelling of sequence-dependent failure behaviour of systems. However, like classical fault tree analysis, TFT analysis requires a lot of manual effort, which makes it time consuming and expensive. This in turn makes it less viable for use in modern, iterated system design processes, which requires a quicker turnaround and consistency across evolutions. In this paper, we propose for a model-based analysis of temporal fault trees via HiP-HOPS, which is a state-of-the-art model-based dependability analysis method supported by tools that largely automate analysis and optimisation of systems. The proposal extends HiP-HOPS with Pandora, Petri Nets and Bayesian Networks and results to dynamic dependability analysis that is more readily integrated into modern design processes. The effectiveness is demonstrated via application to an aircraft fuel distribution system.

Keywords: Fault tree analysis; Reliability analysis; Model-based safety analysis; Dynamic fault trees; Temporal fault trees; HiP-HOPS; Petri nets; Bayesian networks.

1. Introduction

Safety-critical systems underpin many of the advances in modern society and have become an integral part of our life. However, our reliance upon them also means the failure of such systems has the potential to cause great harm, both to people and environment. For this reason, development of such systems requires a rigorous assessment of system behaviour to ensure that they possess a high level of reliability: the ability to perform their intended functions satisfactorily for a prescribed time and under stipulated environmental conditions (Leveson, 1995). Many classical system analysis techniques such as Fault tree analysis (FTA) and Failure modes effects and criticality analysis (FMECA) are available to evaluate system reliability.

Among these techniques, FTA is one of the most common approaches for probabilistic reliability evaluation of a wide range of systems. It is a graphical method which helps determine how a system failure (or "top event") can arise from combinations of component faults and other contributing factors (known as "basic events"). FTA usually has two aspects: a qualitative aspect and a quantitative aspect. Qualitative analysis is performed by transforming fault trees into the minimal cut sets (MCS), which are the smallest combinations of basic events that are necessary and sufficient to cause the top event. In quantitative analysis, the probability of the occurrence of the top event and other quantitative reliability indexes such as importance measures are mathematically calculated, given the failure rate or probability of individual basic events.

Although FTA is widely used for system analysis, it has some known limitations. One of such limitations is that it can only evaluate reliability of static systems. Static systems are those which exhibit a single mode of

operation throughout their lifetimes. However, modern large-scale and complex systems frequently operate in multiple modes or phases, making them dynamic systems. This gives rise to a variety of dynamic failure characteristics such as functional dependencies between events and priorities of failure events. To overcome this limitation, a number of extensions to static fault trees such as dynamic fault trees (DFTs) (Dugan et al., 1992), Boolean logic Driven Markov Processes (BDMP) (Bouissou and Bon, 2003) and Pandora temporal fault trees (TFTs) (Walker, 2009) have been proposed.

In addition, even where software tool support exists, both classical and dynamic fault tree analyses can require a lot of manual effort, meaning the system analyses process is time consuming and expensive. Moreover, given the rapid iterative nature of modern system design, by the time a manual analysis is complete it may already be out of date (Kabir, 2017). These inconsistencies and discrepancies can lead to inaccurate evaluation of system reliability. Therefore, over the past two decades, there has been a lot of research on how to minimise manual effort by automatically synthesise reliability related data from formal system models. This has led to the emergence of model-based dependability analysis (MBDA). Over the years, several tools & approaches such as Hierarchically Performed Hazard Origin and Propagation Studies (HiP-HOPS) (Papadopoulos et al., 2016), Failure Propagation and Transformation Notation (FPTN) (Fenelon and McDermid, 1993), AltaRica (Arnold et al., 2000), and xSAP (Bittner et al., 2016; Bozzano et al., 2015; Bozzano and Villafiorita, 2007) etc. have been developed to support MBDA of systems.

Among these approaches, HiP-HOPS offers advanced compositional MBDA techniques with state-of-the-art tool support. System level analysis and assessment are broken down via composition into more manageable tasks, applied to individual components. A system-level failure model is then produced by composing the failure models of individual components, typically by connecting output deviations of one component to the input deviations of another. The system-level failure model is then automatically analysed to obtain dependability artefacts such fault trees and FMEA. Although HiP-HOPS primarily uses static FTA, it has also been demonstrated that HiP-HOPS can perform dynamic analysis of systems by using Pandora temporal fault trees (Kabir et al., 2017; Walker et al., 2007; Walker and Papadopoulos, 2009).

Pandora is a dynamic extension of static fault trees and defines three temporal gates in addition to the existing Boolean gates. It also provides a set of temporal laws to perform dependability analysis of dynamic systems. Qualitative analysis using Pandora can create useful insight into dynamic system failure and its basis in traditional FTA means it can be integrated into model-based design and analysis processes and tools. Pandora's temporal logic is capable of describing the local failure behaviour of components, enabling compositional synthesis of TFTs from systems models using popular modelling languages such as Matlab Simulink (MathWorks, 2017), EAST-ADL (EAST-ADL Association, 2014), or AADL (Feiler and Rugina, 2007). However, the use of Pandora in the context of HiP-HOPS was only for qualitative analysis. Although qualitative analysis can produce useful information about system reliability, it is advantageous (and sometimes necessary) to have quantitative information about the reliability of the system as well.

In the past, methodologies have been proposed to quantify Pandora temporal fault trees. For example, an analytical method has been developed (Edifor et al., 2012; Kabir et al., 2016), which uses algebraic expressions to probabilistically evaluate the temporal gates of Pandora TFTs. This approach is only applicable to systems

with exponentially distributed lifetime data and works by considering the events (both basic and intermediate events) in the TFTs as statistically independent. However, in real life systems, not all events are statistically independent, and in such situations this can lead to an inappropriate estimation of system reliability. Recently, to overcome these limitations, some preliminary ideas on Petri Nets and Bayesian Networks based quantification methods for Pandora TFT were presented by the authors in (Kabir et al., 2015) and (Kabir et al., 2014) respectively and the approaches were applied to a small system. Further research is required to integrate this work with compositional MBDA so that they can be evaluated by applying them to larger systems.

1.1 Contributions

Given the advantages offered by MBDA and the potential benefits of Pandora for dynamic dependability analysis, integrating these capabilities will open many possibilities. In this paper, we focus on integrating dynamic quantitative analysis using Pandora with compositional model-based dependability analysis via HiP-HOPS. Within this broader context smaller contributions include:

- Consolidation of the ideas presented in (Kabir et al., 2015, 2014) providing two methodologies for probabilistic analysis of Pandora TFTs. In this paper, we move forward to show how the two new proposals for analysis of TFTs can work with a technique that can compositionally produce these TFTs from smaller fragments of analysis and annotations of system models. The paper contains a much larger case study than those used in (Kabir et al., 2015, 2014). Both BN and PN analysis techniques are applied on the same system models and results are compared.
- The processes of calculating importance measures of dynamic system components using the Petri Net- and Bayesian Network-based methods.
- A method for performing diagnostic analysis of dynamic systems using the BN-based approach.

The value of the paper is mostly in the synthesis of several fragments of earlier work, application of the resultant method, comparison between two techniques for quantitative analysis and reflection on results. This work contributes to improved quantitative analysis of dynamic systems in the context of HiP-HOPS method. In addition, the model transformations described in this paper may have a more general value and could be exploited by other work on dynamic and temporal fault trees.

2. Background

2.1 Introduction to HiP-HOPS and Pandora

HiP-HOPS has a long history that goes back twenty years and it is contributing to the state-of-the-art in model-based dependability analysis. It can automatically generate fault trees and FMEA tables from system models, as well as perform quantitative analysis on the fault trees. It can semi-automatically allocate safety requirements to the system components in the form of Safety Integrity Levels (SILs). It also has the ability to perform multi-objective optimisation of the architecture of system models, automating for example decisions about the location and level of replication of components. System analysis using HiP-HOPS is done in three main steps:

1. system modelling and failure annotation

2. fault tree synthesis
3. fault tree analysis and FMEA synthesis

In the first step, a system architecture is created showing the interconnections between system components. The architecture can be arranged hierarchically, i.e., components comprising the system can themselves contain subsystems with their own components. Afterwards, failure annotations are added to the system components to define how they may fail. This dependability related information includes component failure modes and expressions for output deviations, which describe how a component can fail and how it responds to failures that occur in other parts of the system. The expression for the output deviations show how the deviations in the component outputs can be caused either by the internal failure of that component or by corresponding deviations in the component's input. Such deviations can be user defined but typically include omission (O) of output, unexpected commission (C) of output, incorrect output, or too late or early arrival of output (Papadopoulos et al., 2001). Quantitative data can also be entered to facilitate quantitative analysis in a later phase through parametric distribution functions (e.g. failure rate or scale and shape parameters of exponential and Weibull distributions, respectively). Modelling and annotation of the system with dependability information can be done using popular modelling tools like Matlab Simulink or SimulationX (ESI ITI GmbH, 2017).

In the second step, the annotated system model is synthesised to obtain fault trees. The process of constructing fault trees starts with a deviation of system output (top event) and traverses the system architecture deductively, i.e., from the system level outputs to the component level failures, to examine the propagation of failures through connections between components. In this way the process traverses the whole architecture and combines the local fault trees from the individual components until no connected components remain. The result is a single fault tree (or set of fault trees) which represents all the possible combinations of component failures that can lead to the system failure.

In the final phase, the synthesised fault trees are analysed both qualitatively and quantitatively. Qualitative analysis shows the cause of system failure in terms of component failure. Additionally, FMEA tables are generated automatically showing the system failure effects for each component failure (or combination of failures). In quantitative analysis, probability of system failure is estimated based on the failure rate/probability of the basic events.

Generally, temporal dependencies among the events are not considered in traditional FTA. However, HiP-HOPS is able to consider them using Pandora temporal fault trees (TFTs). It has been shown in (Walker et al., 2007; Walker and Papadopoulos, 2009) how Pandora can be integrated within HiP-HOPS to enable qualitative compositional temporal fault tree analysis of dynamic systems. Pandora is a dynamic extension of static fault trees, which defines temporal gates such as Priority-AND (PAND) and Priority-OR (POR) to express sequential relationships between events. The basis of Pandora is an updated version of the long-established Priority-AND (PAND) gate, which dates back to the 1970s (Fussell et al., 1976). Recently, Chiacchio et al. (2017) discussed the implications related to the use of the PAND gate for dynamic failure behaviour modelling with respect to design optimisation and sensitivity analysis.

In Pandora, the PAND gate represents a sequence of events. It is true only if: a) all input events occur; b) all events occur in a sequence from left to right; and c) no input events occur simultaneously. In Pandora, the

PAND is represented by the symbol '<'; however, to avoid confusion between a PAND gate and the less than sign, in this paper we use '⊲' as the symbol for the PAND gate. The POR gate also indicates temporal ordering, but rather than a conjunction of events that must occur in sequence, the POR gate instead specifies that one event has "priority" over the others. The gate is only true if this priority event occurs first, but unlike the PAND gate, it does not require the subsequent events to occur. It is therefore useful in representing trigger conditions in which the priority event occurring first leads to different consequences than if any of the other events occurred first. In Pandora, the POR is represented using the symbol '|', but this symbol is often confused with the conditional probability symbol. To avoid this confusion, in this paper, the symbol 'ɤ' is used to represent the POR gate.

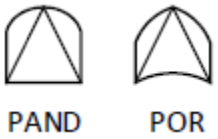


Fig. 1. Graphical representation of PAND and POR gates

Pandora also provides temporal laws to combine these gates with Boolean gates to allow qualitative analysis. Pandora enables the determination of the minimal cut sequences (MCSQs) of TFTs, which are the smallest sequences of events that are needed to cause the top events, similar to the minimal cut sets of classical fault trees. Pandora assumes that the occurrence of the events is instantaneous, i.e., they go from 'non-fail' to 'fail' with no delay. Events are considered as non-repairable, i.e., the event is false as long as the fault has not occurred and when the fault occurs, the event become true and will remain true afterward.

2.1.1 Representation of time in Pandora

Pandora is designed to be generic and does not make many assumptions about the model of time used in any system; time can be discrete or continuous, point or interval-based, or some hybrid. However, whichever way the time is represented, the model of time must be linear, i.e., branching of time is not permitted in Pandora. As the events in Pandora are persistent and occur instantly, there are only three possible temporal relations between two events: before, simultaneous, and after. For this reason, in Pandora "the exact time at which an event occurs is not important — the only thing that matters is when it occurs relative to the other events, i.e., which comes first, which comes second, which comes last etc." (Walker, 2009).

Instead of absolute time values, Pandora uses sequence values to represent the relative order in which events occur. An event that does not occur is given the sequence value 0 (i.e., false). All sequence values greater than zero indicate logical truth, i.e., they indicate that the event occurred. The value itself also shows the order in which the events occurred, e.g. an event with sequence value 1 occurred first, while an event with sequence value 2 occurred second.

Sequence values can be used to define temporal truth tables, since they combine both the Boolean state of an event (occurrence/non-occurrence) together with the relative time of occurrence. Table 1 shows the temporal truth table for the gates of Pandora.

Table 1. Temporal truth table for the gates in Pandora

X	Y	X OR Y	X AND Y	X POR Y	X PAND Y
0	0	0	0	0	0
0	1	1	0	0	0
1	0	1	0	1	0
1	1	1	1	0	0
1	2	1	2	1	2
2	1	1	2	0	0
2	2	2	2	0	0

2.2 Petri Nets and Bayesian Networks

Petri Nets (PNs) are a formal graphical and mathematical modelling tool which is appropriate for specifying and analysing the behaviour of complex, distributed and concurrent systems (Murata, 1989). A classical Petri Net is a bipartite directed graph represented graphically by a finite set of places, a finite set of transitions, and a finite set of directed arcs. Places are visually represented by circles, while transitions are indicated by rectangles. Input and output arcs are directed edges drawn from input places to transitions and transitions to output places, respectively. Input places are known as the pre-conditions of a transition and a specified number of tokens are needed to be available in the input places for the transition to be enabled. Tokens are usually represented by black dots. On the other hand, output places are known as post-conditions of a transition and on firing/ triggering of a transition a specified number of tokens are consumed from each input place and a specified number of tokens are deposited to the output places.

The inhibitor arc is a special form of arc which increases the modelling power of PNs by allowing to model non-occurrence of events. Similar to other arcs, an inhibitor arc also connects a place and a transition and is usually graphically represented by an arc that ends in a small circle. The interesting property of an inhibitor arc is that it enables the transition when the input place has no token and disables the transition when the place has a token (opposite behaviour of a normal arc). At the same time no tokens are consumed through an inhibitor arc.

It is problematic to model time-dependent behaviour using classical PNs. Stochastic Petri Nets (SPN) (Molloy, 1982) are an extension of PNs that allow transition delays to be exponentially distributed. To allow both immediate and timed transition in the same Petri Net, SPNs are extended to introduce Generalized Stochastic Petri Nets (GSPN) (Marsan et al., 1996). In GSPNs, timed transitions are usually represented graphically by white bars and fire after a random period of time. On the other hand, immediate transitions are graphically represented by black bars and they fire immediately after being enabled. Immediate transitions have priority over timed transitions, i.e., if an immediate and a timed transition are enabled at the same time, the immediate transition fires first. If two or more timed transitions are enabled to fire at the same time (e.g., all have same firing rate), then one of them is selected randomly to fire first.

PNs have been used to model both functional and non-functional behaviour of systems. For example, in (Fanti et al., 2014) a Petri Net has been used to model an Integrated System (IS) for the Healthcare at Home (HAH) management system, and the PN model was subsequently simulated to observe the behaviour of the IS. Some

researchers have used Petri nets to model functional behaviour of the system and then use another safety analysis method e.g. FTA to perform the safety analysis based on the non-functional behaviour visible in the PNs (e.g. (Reza et al., 2009)). Petri Nets have also been used to evaluate Fault Trees. For example, Bobbio et al. (1999) and Helmer et al. (2007) used PNs to quantify classical FTs; and Codetta-Raiteri (2005) and Zhang et al. (2009) used PNs for the quantification of DFTs.

Bayesian Networks (BNs) are directed acyclic graphs that represent a set of random variables and their conditional dependencies (Pearl, 1988). The nodes of the BN represent the random variables and the directed arcs represent dependencies or cause-effect relations among the nodes. In a BN, a node X is said to be the parent of another node Y if there exists an arc from node X to node Y. Parent nodes have a direct effect on their child nodes and it is defined as $Pr\{X_i|Parents(X_i)\}$ which quantifies the effect of the parents on the child node. If a node has no parent, then it is known as a root node, and if a node has no children, then it is a leaf node.

Using the conditional independence assumptions of BNs, the joint probability distribution of a set of random variables $\{X_1, X_2, X_3, \dots, X_{n-1}, X_n\}$ can be determined using a chain rule as:

$$Pr(X_1, X_2, X_3, \dots, X_{n-1}, X_n) = \prod_{i=1}^{n} Pr(X_i|Parents(X_i)) \quad (1)$$

BNs have been applied in the field of dependability because of their ability to combine different sources of information to provide a global safety assessment and to enable robust, probabilistic reasoning in conditions of uncertainty. Usually BNs are used in two different ways to perform dependability analysis of systems. In the first way, systems are modelled using BNs and then the analysis is performed on that model. Researchers like (Doguc and Ramirez-Marquez, 2009; Huang et al., 2008; Neil et al., 2008; Neil and Marquez, 2012; Torres-Toledan and Sucar, 1998) have shown the ways of modelling systems directly in Bayesian Networks, and subsequent analysis on that system model. The second approach of using BNs in dependability analysis domain is to transform other representations like FTs and DFTs into BNs. For example, Bobbio et al. (2001) have proposed a way of translating fault trees into BNs. They represented the basic events of the FTs as the root nodes of the BNs and the intermediate events (Boolean gates) as the intermediate nodes of the BNs. As the Boolean gates of the FTs represent deterministic causal relationships between the input and output events, the non-root nodes are actually deterministic nodes and all the entries in the CPT are either 0 or 1. In this paper, a single circle is used to represent a root node and doubled circle is used to present deterministic nodes (intermediate and leaf nodes). The prior probabilities of the root nodes are same as the probabilities assigned to the basic events (leaf nodes) in the FTs. Methods for transforming dynamic fault trees (DFTs) into a dynamic Bayesian network (DBN) have been proposed in (Boudali and Dugan, 2006, 2005). To allow reliability analysis based on automatic translation of DFT to DBN, a tool called RADYBAN (Reliability Analysis with DYnamic Bayesian Networks) has been developed by Montani et al. (2008).

3. Quantitative Model-Based Dynamic Analysis in HiP-HOPS

To perform quantitative analysis of dynamic failure behaviour, Pandora has been integrated with the HiP-HOPS compositional model-based dependability analysis process. The workflow for this process is shown in Fig. 2.

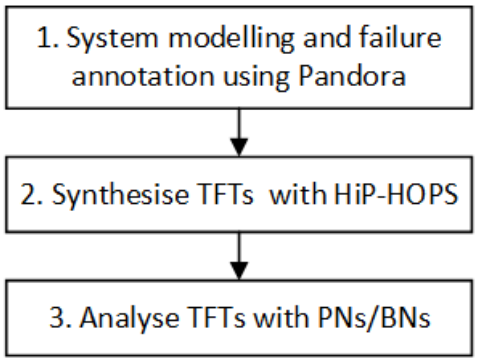
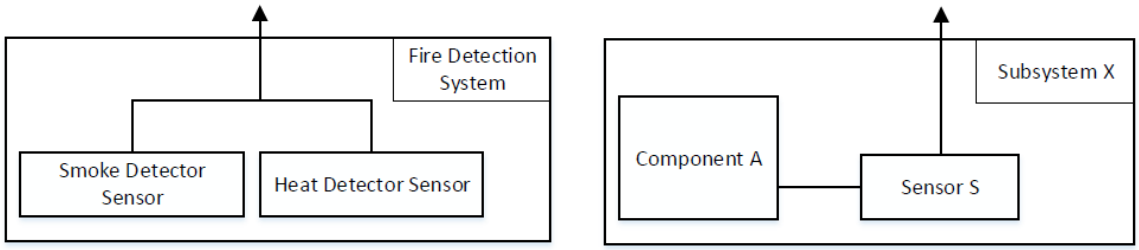


Fig. 2. Integrated workflow for using Pandora MBDA via HiP-HOPS

The three steps shown in the above workflow are similar to the steps in the HiP-HOPS as seen in section 2.1. In the first step, system modelling and failure annotation is defined by showing the interconnections among the system components and how they can fail. In order to capture the sequence dependent behaviour of components, temporal logic is used in addition to the Boolean logic. For instance, consider the fire detection system in Fig. 3(a). The output of this system is dependent on the output of both the smoke detection sensor and the heat detector sensor, i.e., if both the smoke detector and heat detector sensors detect smoke and heat respectively, the system will produce an output confirming that fire is detected. As a result, the output deviation of the fire detection system can be expressed using Boolean logic as:

O-Fire Detection System = O-Smoke Detector Sensor **AND** O-Heat Detector Sensor

Where "O-Fire Detection System" is the omission of output from the fire detection system, "O-Smoke Detector Sensor" is the omission of output from the smoke detector, "O-Heat Detector Sensor" is the omission of output from the heat detector, and 'AND' represents a Boolean AND operation.



(a) Fire detection system (b) A hypothetical subsystem

Fig. 3. Fire detection system and a hypothetical system

Now consider the system in Fig. 3(b), which is a hypothetical subsystem of a bigger system. This subsystem contains a component *A* and a sensor *S*. *A* performs some useful operation in the system and *S* monitors the functionality of *A*. Failure of *A* has no catastrophic effect on the overall system performance unless the failure goes undetected. If *A* fails then *S* can detect it, hence the subsystem *X* can produce an output by raising an alarm. If *A* and *S* both fail, then we have to consider two scenarios: *A* fails before *S* fails and *A* fails after *S* fails. In the first scenario, subsystem *X* will be able to produce output because by the time *S* fails it has already done its job of detecting the failure of *A*. On the other hand, in the second scenario, as *S* fails first, the failure of *A* will go undetected, thus no output is produced by *X*. The above scenario can be described by using a PAND gate and the failure behaviour of *X* can be expressed using temporal logic as:

O-X = O-S ◅ O-A

Once all the components in the system architecture are annotated with failure expressions, in the second step, TFTs can be synthesised from the annotated system model by using HiP-HOPS. This step is the same as the second step of HiP-HOPS technique as described in section 2.1. However, the outcome would be a TFT showing the failure behaviour of the system. In the third step, the quantitative analysis of TFTs is done using either the Petri Nets or Bayesian Networks. As mentioned earlier, some preliminary ideas on Petri Net- and Bayesian Network-based approaches for quantitative analysis of Pandora TFTs have been presented in (Kabir et al., 2015, 2014). In the following sections, we extended the above ideas by providing detailed descriptions of the approaches and operationalize the approaches.

3.1 Petri Net Based Approach

The main idea of the Petri Net based approach is to translate Pandora temporal fault trees into generalised stochastic Petri Nets (GSPN), thus providing a state-space solution to TFTs. In the translation process, each temporal fault tree node is translated to a particular sub-PN with a place indicating the status of the temporal fault tree node. Places are therefore used to indicate the state of the system, while timed transitions represent random faults and immediate transitions indicate the propagation of failures.

The mapping of a basic event to GSPN is shown in Fig. 4. A token in the place *Working* represents a fully functional system component at time 0. The firing rate of the timed transition *Fail* corresponds to the rate of the exponential probability density function (PDF) associated with the transition, which is characterised by the exponential distribution of the failure rate of the component. As a result, if the failure rate of a component is defined as λ, then the probability that the transition is fired at the time instant t is $1 - e^{-\lambda t}$ with $\lambda < 1$.

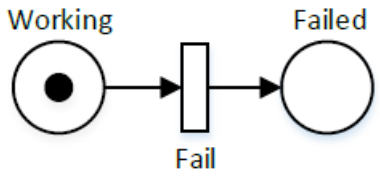


Fig 4. GSPN of a basic event

Boolean gates (AND and OR gates) in Pandora TFT are mapped to PN based on the work presented in (Bobbio et al., 1999), and shown in Figs. 5 and 6, respectively.

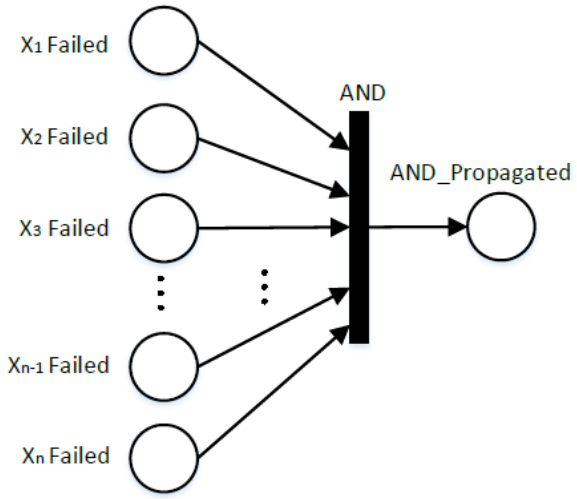


Fig. 5. GSPN of *n* input AND gate

The places X_1 *Failed*, X_2 *Failed*, X_3 *Failed*, ⋯, X_{n-1} *Failed*, and X_n *Failed* represent the input events to the AND gate. All the places are connected to an immediate transition (*AND*). As a result, the transition will fire when all the input places have a token (i.e., all input events occur) and on firing of the transition a token will be deposited to the place representing the outcome of the AND gate (*AND_Propagated*). Unlike the AND gate, the OR gate represents a logical disjunction of the input events, and means the outcome of the OR gate becomes true if at least one of the input events become true. The OR gate is translated to a GSPN model by creating an immediate transition for each place corresponding to input events of the OR gate (see Fig. 6). Hence, as soon as one of the input places gets a token, the transition connected to that place will fire and the place representing the outcome of the OR gate (*OR*) will gain a token.

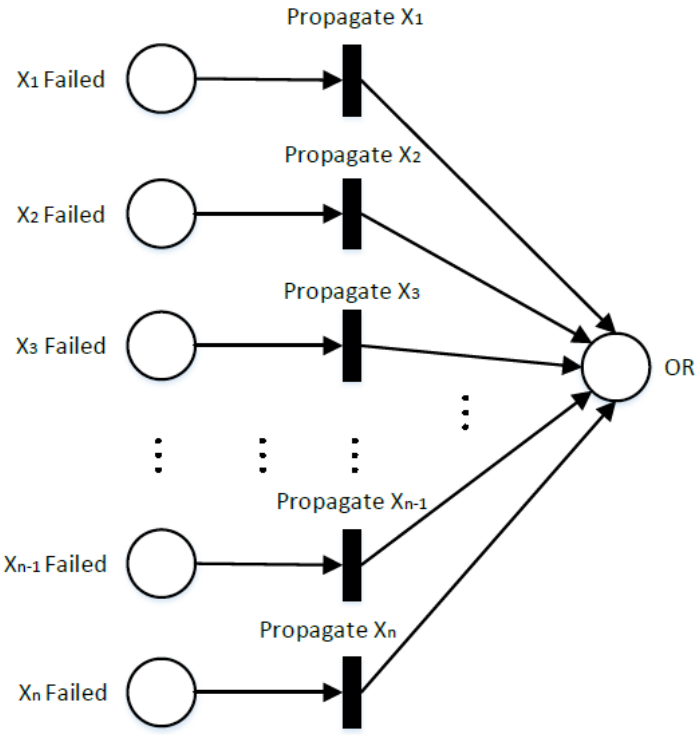


Fig. 6. GSPN of *n* input OR gate

The mapping of the temporal gates to GSPN models is not as straightforward as that of the Boolean gates. This is due to the fact that the Boolean gates are stateless in the sense that they do not need to remember the order of occurrence of the events. However, along with the occurrence of the input events, the temporal gates must also remember the order of occurrence of input events. Fig. 7 shows the mapping of an *n* input PAND gate to a GSPN model. The transformation is made in such a way that ensures the exact sequencing of events to make the PAND outcome true. The place X_1 *Failed* is connected to the transition T_1 and all other places corresponding to other input events are connected to this transition using inhibitor arcs. As a result, the transition T_1 will fire when there exists a token in place X_1 *Failed* and no token in any other connected places (X_2 *Failed* to X_n *Failed*). Once this transition fires, the place P_1 would get a token and essentially will represent the scenario when only the left-most input of the PAND gate occurs but not the others. Similarly, the transition T_2 will fire to deposit a token to place P_2 when the second input becomes true (i.e., place X_2 *Failed* gets a token) and all the subsequent inputs remain false. In a similar fashion, the firing of transition T_{n-1} deposits a token to place P_{n-1}, thus representing that the $(n-1)^{th}$ event to the PAND gate has occurred and the n^{th} event is still to occur. So, the presence of a token in each of the places from P_1 to P_{n-1} confirms that events X_1 *Failed* to X_{n-1} *Failed* have occurred in a sequence from left to right and the event X_n Failed is yet to occur. Now, the places P_1 to P_{n-1} and the place X_n *Failed* are connected to an immediate transition T_n and the firing of this transition would deposit a token to the place PAND which essentially represents the scenario when the PAND outcome becomes true, i.e., all input events to the PAND gate occurred and they occurred obeying the sequence.

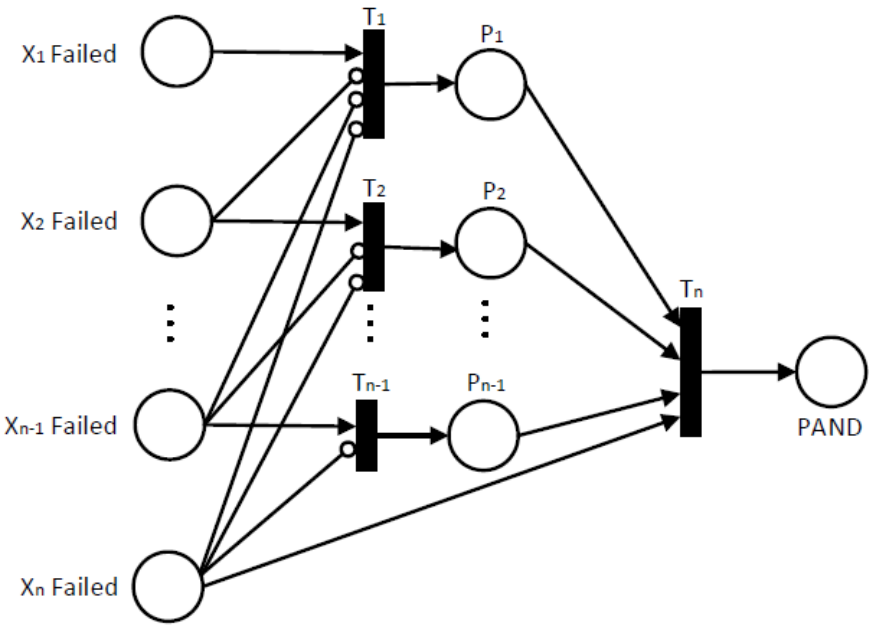


Fig. 7. GSPN of *n* input PAND gate

The POR gate indicates that one particular input event has priority and must occur first. That means, if the disjunction of the non-priority events occurs at all then it should occur after the priority event. For example, let us consider that X_1, X_2, X_3, ⋯, X_{n-1}, and X_n are *n* different input events to a POR gate where the event X_1 is the priority event. The outcome of the POR gate can become true if at least one of the following two conditions becomes true.

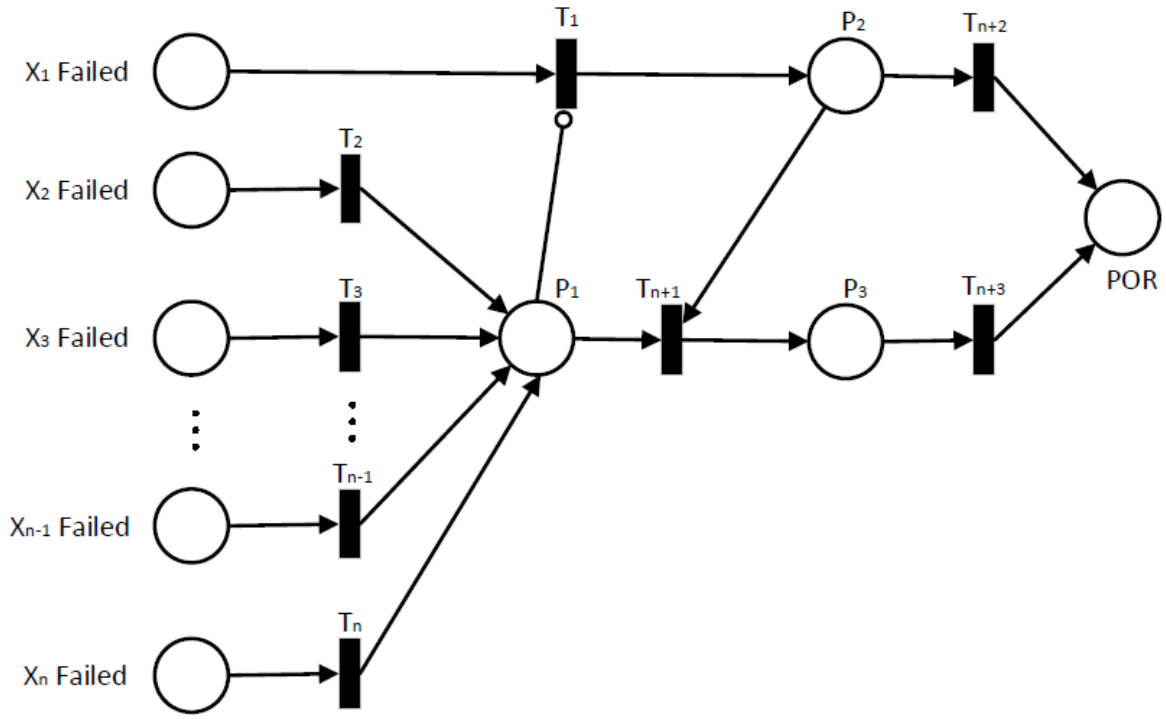


Fig. 8. GSPN of *n* input POR gate

1. Condition 1: $X_1 \wedge \overline{(X_2 \vee X_3 \vee \ldots \vee X_{n-1} \vee X_n)}$
2. Condition 2: $X_1 \lhd (X_2 \vee X_3 \vee \ldots \vee X_{n-1} \vee X_n)$

The first condition refers to a scenario where only the priority event occurs. The second condition represents a scenario where one or more non-priority events may have occurred but they occurred after the priority event. The exact logical mapping of an *n* input POR gate to GSPN is shown in Fig. 8. In the figure, the place P_1 represents the logical OR of the non-priority events. The places P_2 and P_3 represent the conditions 1 and 2 respectively. The place named POR represents the logical OR of the two conditions, i.e., the outcome of the POR gate.

Condition 2 actually represents a two input PAND gate where the left input is the priority events of the POR gate and the right input is the logical OR of all the non-priority events of the POR gate. From the structure of the PN model in Fig. 8, it can be seen that the place P_2 will always get a token first and subsequently enables the transition T_{n+2} to deposit a token to the place *POR*. So, the POR outcome will never become true because of a token in place P_3. As a result, the place P_3 becomes redundant in this PN model, which in effect makes the transitions T_{n+1} and T_{n+3} extraneous as well. The refined PN model for the *n* input POR gate is shown in Fig. 9.

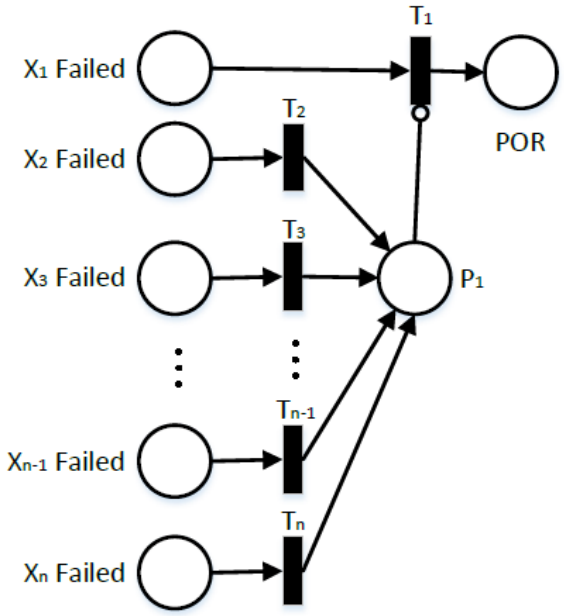


Fig. 9. Refined GSPN of *n* input POR gate

In order to probabilistically evaluate Pandora TFTs via PN models to obtain information about system unreliability, at first we need to transform the TFT to the GSPN model. The basic events and the intermediate events (logic gates) of the TFT are transformed to GSPN model following the process described above. In the GSPN, there will be a place representing the top event of the TFT. The GSPN of the TFT can be simulated to obtain information about system reliability for specified mission time.

3.2 Bayesian Network Based Approach

The primary goal of the Bayesian network based approach is to translate the Pandora TFTs into Bayesian Networks, thus performing quantitative analysis of dynamic systems using the BN model. The translation of TFT to BN is done in two steps (see Fig. 10). In the first step, graphical mapping of TFT to BN is done by translating basic events to root nodes, intermediate events (logic gates) to intermediate nodes, and top event to leaf node. In the second step, numerical mapping is done by populating prior probability values for the root nodes based on failure probability of the basic events and conditional probability tables for other nodes based on the type of logic gates they represent.

Pandora TFT features both Boolean and temporal gates. The outcome of temporal gates depends on the order in which their input events occur, and this order is represented using sequence values. On the other hand, the outcomes of Boolean gates do not depend on the sequencing of the input events. However, in a TFT, the output of a temporal gate can be connected to the input of a Boolean gate and vice versa. For this reason, it is necessary to define temporal behaviour for the Boolean gates.

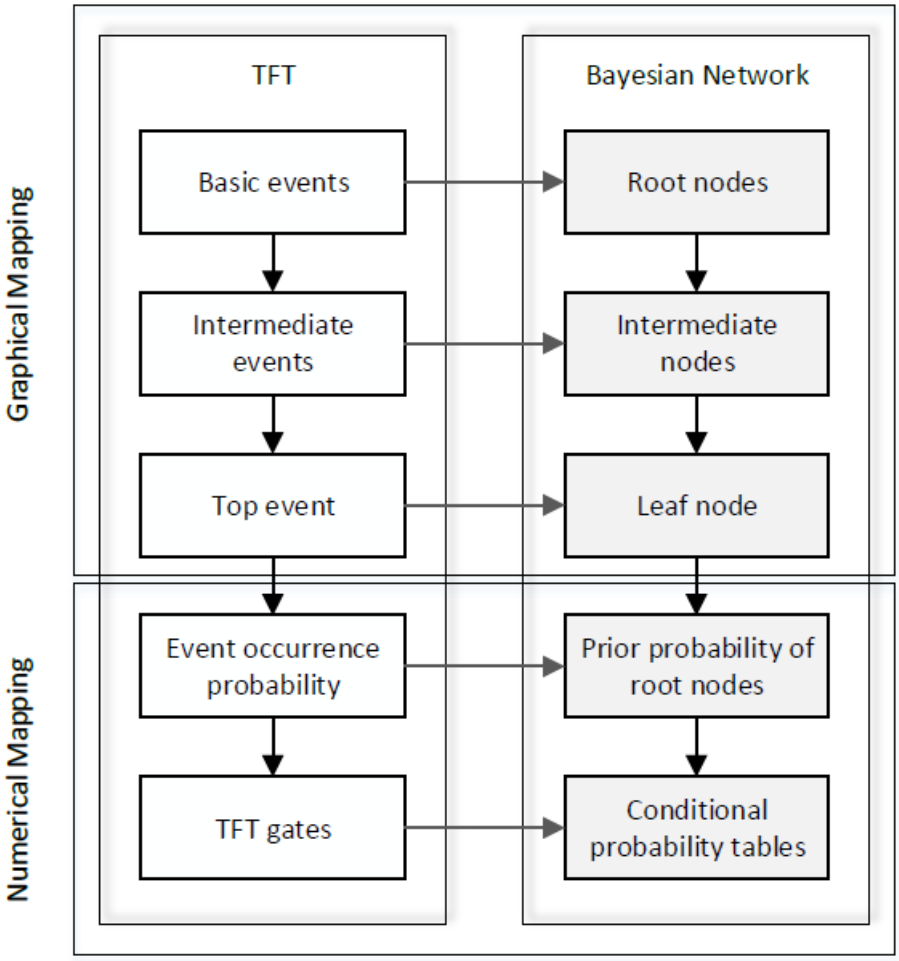


Fig. 10. TFT to Bayesian Network mapping

To represent sequence values, which are used in Pandora to represent sequencing among events, in this paper, the mission time *T* is divided into *N* equal intervals from *t=0* to *t=T*, where each interval represents a possible non-zero sequence value, during which an event occur. Initially, all components are assumed to be fully operational and therefore all events are given an initial sequence value of 0 (i.e., the component has not yet failed). This value holds until the occurrence of a component failure, i.e., if the component can survive till the end of the mission time then it will continue carrying the sequence value 0. If a component fails in interval 1, then it will have the sequence value 1, but if the component fails in any other interval *i* where $1 < i \leq N$ then it can have any sequence value in between 1 to *i* based on its relative position to its immediate predecessor.

An event having a sequence value '*i*' is considered to be in '*State i*' and it has an associated probability value representing the probability of the event being in '*State i*'. As the root nodes in the BN represent different basic events, we need to define prior probability tables for the root nodes, where each entry in a prior probability table of a node represents the probability of the respective event being in a particular state. For exponentially distributed failure rate, the probability of a component being failed in the interval *[t1, t2]* (e.g. in *State i*) can be obtained by integrating the probability density function (PDF) of exponential distribution, $\lambda e^{-\lambda t}$, in the following way:

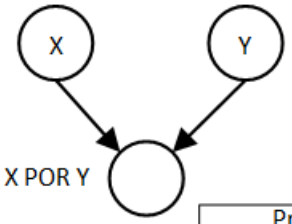
$$\Pr\{BE_i\}_{[t1,t2]} = \int_{t1}^{t2} \lambda e^{-\lambda t}\, dt \qquad (2)$$

Once prior probability values are assigned to each root node, conditional probability values are generated for all the intermediate nodes. Remember that Pandora represents the outcome of every gate with a sequence value, showing not only whether that gate is true or false but also the relative order in which the gates become true (see Table 1). This is purely deterministic: the outcome of each gate depends solely on the sequence values of its input events. Consequently, the probability of an intermediate BN node (representing a gate) being in a certain state can either be 0 or 1, depending on the state of its parent nodes.

In (Kabir et al., 2014), the authors have shown the translation and CPT generation process for the POR gate, as seen in Fig. 11. The mapping of a PAND gate by diving the mission time into 2 intervals is shown in Fig. 12. The CPT of the PAND gate resembles its temporal truth table. As seen in the CPT of the PAND gate, the PAND outcome becomes true only in one scenario when the first input (X) is in *State 1* and the second input (Y) is in *State 2*, i.e., they occur in a sequence from left to right. In this case, the state associated with the PAND outcome is *State 2* because this is the state when the last input of the PAND becomes true.

X	State 0	State 1	State 2
Pr {X}	P0	P1	P2

Y	State 0	State 1	State 2
Pr {Y}	P0	P1	P2



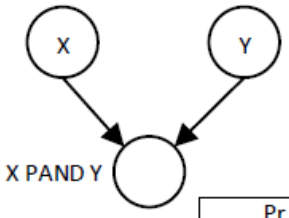
		Pr {X POR Y}		
X	Y	State 0	State 1	State 2
State 0	State 0	1	0	0
State 0	State 1	1	0	0
State 0	State 2	1	0	0
State 1	State 0	0	1	0
State 1	State 1	1	0	0
State 1	State 2	0	1	0
State 2	State 0	0	0	1
State 2	State 1	1	0	0
State 2	State 2	1	0	0

Fig. 11. BN of POR gate

Bayesian Network of the AND and OR gates are shown in Figs. 13 and 14, respectively. From the CPT of the AND gate in Fig. 13, we can see that the AND outcome becomes true (1s in column *State 1* or *State 2*) when the input events are either in *State 1* or *State 2*. If any of the input event is in *State 0* (logical false) then the outcome of the gate is false (1s in the column *State 0*). So the CPT of the POR, AND and the OR gates are of the same pattern and entries in the table are either 0 or 1, but the positions of the 0s and 1s change according to the logical specification of the gates.

X	State 0	State 1	State 2
Pr {X}	P0	P1	P2

Y	State 0	State 1	State 2
Pr {Y}	P0	P1	P2

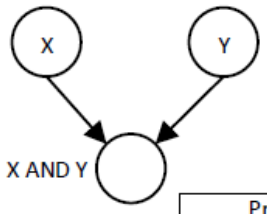


		Pr {X PAND Y}		
X	Y	State 0	State 1	State 2
State 0	State 0	1	0	0
State 0	State 1	1	0	0
State 0	State 2	1	0	0
State 1	State 0	1	0	0
State 1	State 1	1	0	0
State 1	State 2	0	0	1
State 2	State 0	1	0	0
State 2	State 1	1	0	0
State 2	State 2	1	0	0

Fig. 12. BN of PAND gate

X	State 0	State 1	State 2
Pr {X}	P0	P1	P2

Y	State 0	State 1	State 2
Pr {Y}	P0	P1	P2



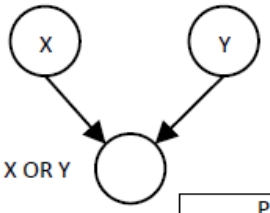
		Pr {X AND Y}		
X	Y	State 0	State 1	State 2
State 0	State 0	1	0	0
State 0	State 1	1	0	0
State 0	State 2	1	0	0
State 1	State 0	1	0	0
State 1	State 1	0	1	0
State 1	State 2	0	0	1
State 2	State 0	1	0	0
State 2	State 1	0	0	1
State 2	State 2	0	0	1

Fig. 13. BN of AND gate

Once we have the Pandora TFT of the failure behaviour of a system, we can translate the TFT to BN model and subsequently perform predictive reasoning on the Bayesian Network to obtain system unreliability. This is done by following the direction of the BN arcs from the root nodes towards the leaf node. In this process, failure probability data of the root nodes are used to obtain the probability of system failure, i.e., data about causes are used to obtain new belief about the effects. Using the facility of observing the status of a node, we can also perform diagnostic analysis on the BN, i.e., reasoning from effects back to their causes. If the analysts have the evidence that the system has failed, then based on this evidence the analysts' belief about the failure probability of the components can be updated. That means we now have to put an observation on the leaf node of the BN and work backwards (in the opposite direction of the BN arcs) towards the root nodes to update the probability of root nodes.

X	State 0	State 1	State 2
Pr {X}	P0	P1	P2

Y	State 0	State 1	State 2
Pr {Y}	P0	P1	P2



		Pr {X OR Y}		
X	Y	State 0	State 1	State 2
State 0	State 0	1	0	0
State 0	State 1	0	1	0
State 0	State 2	0	0	1
State 1	State 0	0	1	0
State 1	State 1	0	1	0
State 1	State 2	0	1	0
State 2	State 0	0	0	1
State 2	State 1	0	1	0
State 2	State 2	0	0	1

Fig. 14. BN of OR gate

To facilitate the diagnostic analysis, the Bayes theorem shown in eq. (3) is used.

$$Pr(A|B) = \frac{Pr(B|A)Pr(A)}{Pr(B)} \tag{3}$$

where $Pr(A|B)$ is the conditional (posterior) probability of event *A* given evidence about the probability of event *B*, *Pr(A)* and *Pr(B)* are the prior probability of event *A* and *B* respectively.

3.3 Importance Measures

Importance measures determine the various contributions of basic or intermediate events to the occurrence of the top event or how a change in any of these events can affect the occurrence of the top event. This information can serve as a useful source of data for resource allocation (upgrade, maintenance etc.) and helps stakeholders in improving system dependability (safety, reliability, availability etc.). In classical FTA, different importance measures such as Birnbaum importance measures, Fussel-Vesely importance measures, and Risk Reduction Worth (RRW) (Vesely et al., 2002) are widely used. Cheok et al. (1998) showed how these different measures can be calculated. In this paper, we use RRW to measure the criticality of basic events. Note that other importance measures can also be used for the above purpose; however, for the purposes of brevity, we have focused on RRW. RRW shows the effect of BE on TE with respect to non-occurrence of BE_i. RRW represents the decrease of TE probability when a given BE is assured not to occur. Using the PN based method, the RRW of basic event BE_i can be computed as:

$$RRW(BE_i) = Pr\{TE\} - \Pr\{TE|BE_i = 0\} \tag{4}$$

Where $Pr\{TE\}$ is the probability of the top event and $Pr\{TE \mid BE_i = 0\}$ is the probability of the top event given that the basic event BE_i has not occurred (i.e., it is fully functional). In a PN model, to make a component fully functional, we simply need to remove the token from the place representing the component. Once we remove the token from a place, then even though the place is connected to a timed transition, during the simulation the

transition will never fire and thus the component corresponding to the place will not contribute to the occurrence of the top event.

RRW of event BE_i can be computed from the BN model as:

$$RRW(BE_i) = Pr\{TE\} - \Pr\{TE|BE_i = State\ 0\} \quad (5)$$

Where $\Pr\{TE|BE_i = State\ 0\}$ is the probability of the top event given that the BE_i is in *State 0*, i.e., BE_i has not occurred. This value can be obtained from the BN model by running a query on the leaf node by observing the node representing BE_i in *State 0*.

4. Case Study and Evaluation

4.1 System Description

To illustrate the idea of model-based compositional analysis of dynamic behaviour of system via HiP-HOPS, we use an Aircraft Fuel Distribution System (AFDS) first presented in (Edifor et al., 2014), reworked and shown in Fig. 15. The system consists of 2 engines, 5 fuel tanks to store fuel, 7 bi-directional fuel pumps embedded with speed sensor to pump fuel throughout the systems, particularly towards the engines, 11 valves that can be used to activate or close some paths according to system requirement in different situations, 6 flow meters to measure the rate of fuel flow through the pipes, a refuelling point to refill the tanks, and 2 jettison points to release fuel to the atmosphere if required.

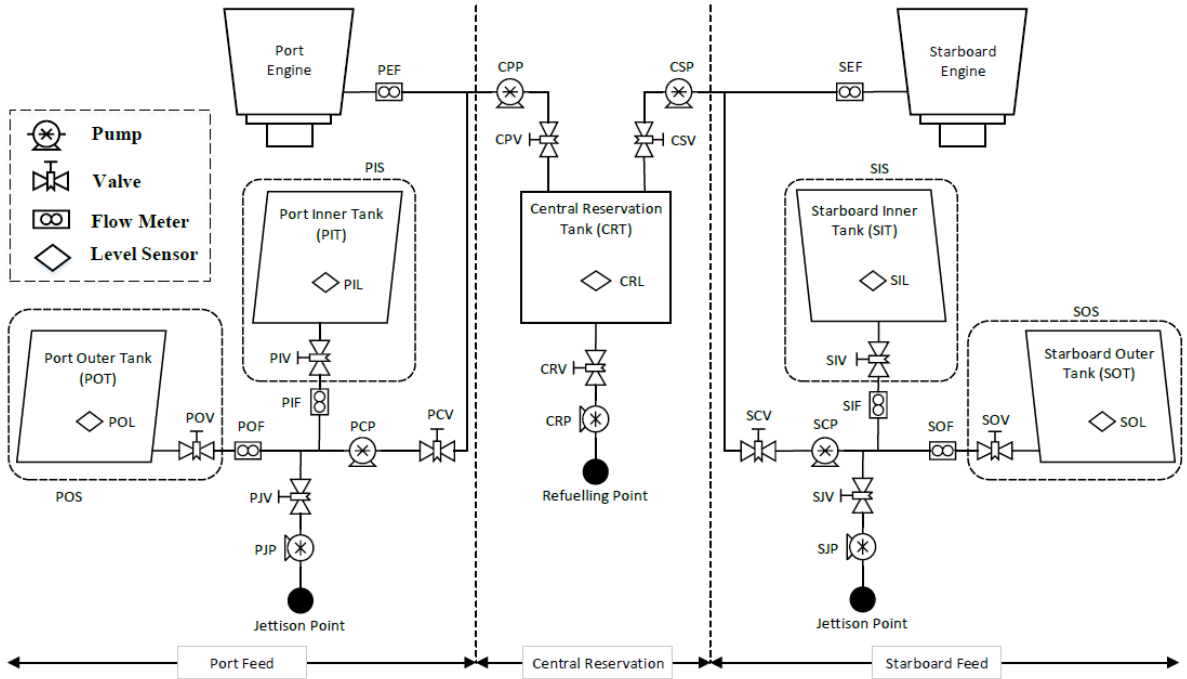


Fig. 15. Aircraft Fuel Distribution System

In order to perform compositional analysis, the AFDS is divided into three sub-systems: Port Feed (PF), Central Reservation (CR), and Starboard Feed (SF). PF and SF contain identical sets of components and they are further divided into several subsystems. For example, the SF contains Starboard Inner Subsystem (SIS) and Starboard

Outer Subsystem (SOS). These subsystems are further decomposed into system components. For instance, the SIS contains the Starboard Inner Tank (SIT), fuel level sensor (SIL), and the valve SIV.

The AFDS has two main functions: storing fuel in the tanks and distributing the fuel throughout the system. These functions are provided in two different phases—refuelling and consumption. In the refuelling phase, fuel is injected to the Central Reservation Tank (CRT) and subsequently automatically distributed to the other tanks. In the consumption phase (during taxiing, take-off, cruising, approaching, and landing), fuel is consumed by the engines and adequate level of fuel is fed to the engines from the appropriate tanks. Under normal operating conditions, in the consumption phase, the jettison points are kept closed and the Port Outer Subsystem (POS) feeds the Port Engine (PE) from Port Outer Tank (POT) and the SOS feeds the Starboard Engine (SE) from the Starboard Outer Tank (SOT). In this case, in the SF subsystem, the valve SIV in the SIS remains closed and the SIS is in dormant mode. If for some reason fuel flow is disrupted from the SOS, SIS is activated (SOS is deactivated) and fuel is drawn from the SIT to feed the SE. Now, if the SIS failed to provide fuel to the SE, the whole SF subsystem is deactivated and fuel is drawn from the Central Reservation Tank (CRT) of the CR subsystem. The same operation is possible in the PF subsystem as well.

As already mentioned, jettison points are kept close during normal operation. However, in any emergency condition, the jettison points could be opened by opening the corresponding valves to release the fuel to the atmosphere. Jettison of fuel may be required during an emergency landing where the mass of the aircraft needs to be reduced by discarding extra fuel. In the presence of failure, the system has to change its configuration by activating and/or deactivating system components to compensate for the effects of the failure and to continue system operation (either with equal or degraded functionality). A centralised computerised control system performs the required activation or deactivation or other calibration of the system components.

4.2 Model-based Analysis of the System

As mentioned earlier, in HiP-HOPS, model-based analysis of system is performed in a compositional fashion. For this reason, the AFDS system is divided into smaller subsystems and the subsystems are further divided into smaller subsystems to the level of components. Components are annotated with their failure behaviour and their behaviour are combined to obtain the behaviour of the subsystem possessing them. While annotating components, the following naming convention is used:

- O-CompX - omission of functionality of component X
- I-CompX – internal failure of component X
- Hi-CompX – erroneous high reading from component X

For instance, omission of functionality of level sensor SOL in the SOS is caused by an internal failure of SOL. This behaviour can be expressed as:

$$\text{O-SOL} = \text{I-SOL}$$

Other components can be similarly annotated. As already mentioned these component annotations can then be composed to obtain annotations for subsystems. For example, the SOS contains both the SOL and SOV, hence the failure behaviour of the SOS (omission of its functionality) can be annotated as:

$$O\text{-}SOS = O\text{-}SOL + O\text{-}SOV$$

Because the expressions are compositional, O-SOL and O-SOV can be expanded with their own causes (I-SOL and I-SOV respectively) and in this way we can automatically synthesise the expression for the full subsystem:

$$O\text{-}SOS = (I\text{-}SOL) + (I\text{-}SOV)$$

And using this same principle, we can in turn derive the full failure expression for the entire system, representing the logic of the system fault tree.

From the architecture of the AFDS, it is seen that the failure of PF, CR, and SF will lead to the complete failure of the system. Due to the fault tolerant strategy in place, failure of either of PF or SF together with CR will lead to degraded functionality of the system. At the same time, failure of any one of the subsystems (SOS, SIS, POS, and PIS) will not affect the normal functionality of the system, i.e., the system will still be able to provide fuel to both the engines. Moreover, in the condition of failure of either of PF and SF, the system can continue to operate normally by substituting the failed subsystem's functionality using the CR subsystem. In this paper, we consider the failure of the SF subsystem as the top event of the temporal fault tree (note that failure of PF will be symmetrical).

Model-based analysis of the AFDS is performed to obtain the causes of SF failure and the temporal fault tree obtained from the analysis is shown in Fig. 16. From the TFT, it can be seen that some intermediate events share other intermediate events, which make them statistically dependent. For example, intermediate events IE2, IE5, IE6, and IE7 have the intermediate O-SOS as their common input and as a result they become statistically dependent. If we use the analytical approach to quantify this TFT, the statistical dependence among the events will be ignored. However, the approaches proposed in this paper can quantify this TFT by taking account of these dependencies and hence produce a more realistic result.

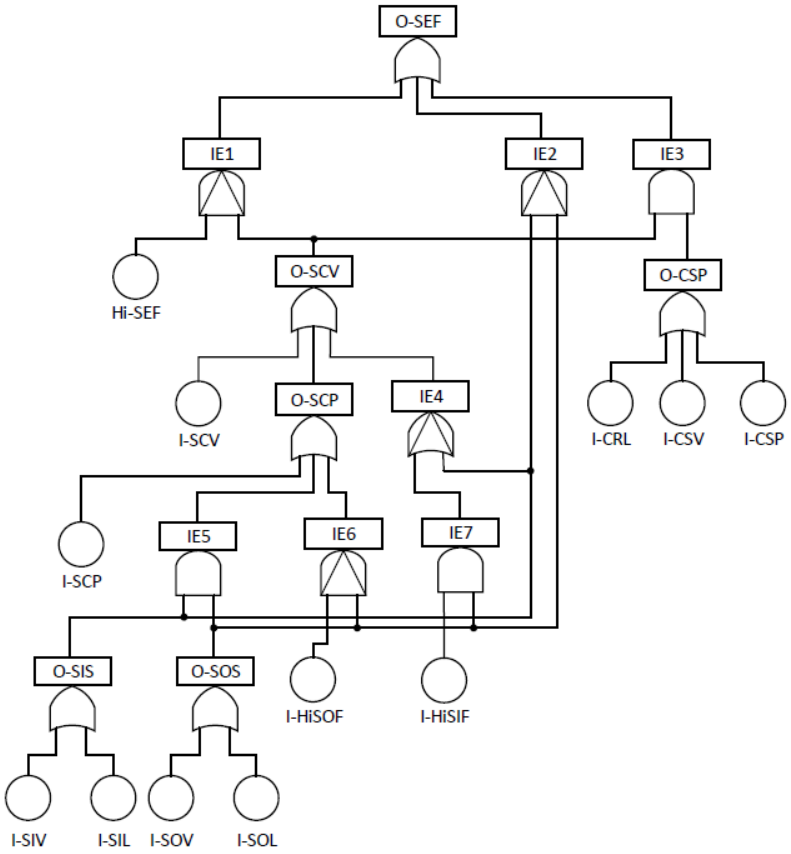


Fig. 16. Pandora TFT of the AFDS

For quantitative analysis of the TFT in Fig. 16, the TFT is mapped to GSPN and Bayesian Network models and shown in Figs. 17 and 18, respectively. The failure rates of the basic events of the TFT are shown in Table 2. As seen in Fig. 17, the time transitions are characterised by the failure rate of basic events.

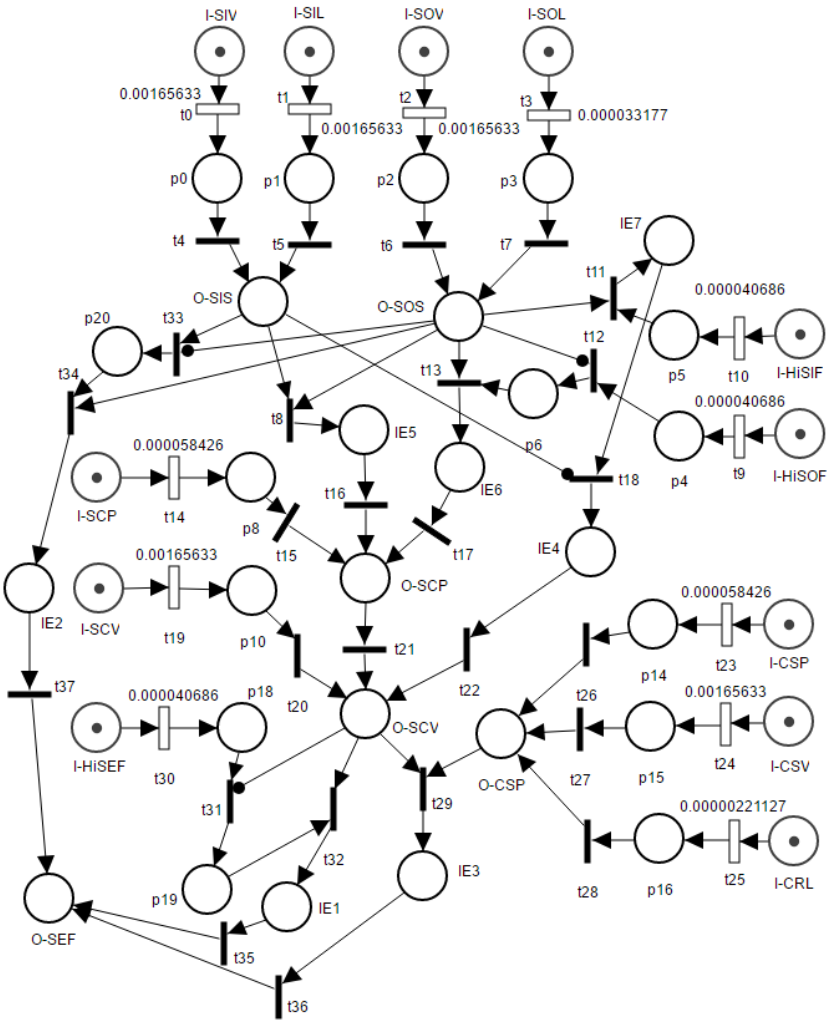


Fig. 17. Petri Net model of the TFT in Fig. 16

Table 2. Failure rate of the basic events (Edifor et al., 2014)

Basic Events	Failure Rate (per hour)
I-SCP	5.84267E-5
I-CSP	5.84267E-5
I-SOV	1.65633E-3
I-SIV	1.65633E-3
I-CSV	1.65633E-3
I-SCV	1.65633E-3
I-CRL	2.21127E-6
I-HiSOF	4.06861E-5
I-HiSIF	4.06861E-5
I-HiSEF	4.06861E-5
I-SIL	1.65633E-3
I-SOL	3.31774E-5

The prior probability table of each root node of the BN is populated based on the failure rate of the corresponding component. As described in section 3.2, in order to generate the prior probability values of the events, we need to define a mission time and the number of intervals (N) to divide the mission time into discrete intervals. The prior probability values for the root nodes of the BN of Fig. 18 are calculated assuming a mission

time of 5000 hours and dividing the mission time into *4* intervals (*N=4*), shown in Table 3. The conditional probability table of each intermediate node of the BN is populated based on the type of TFT gate it represents.

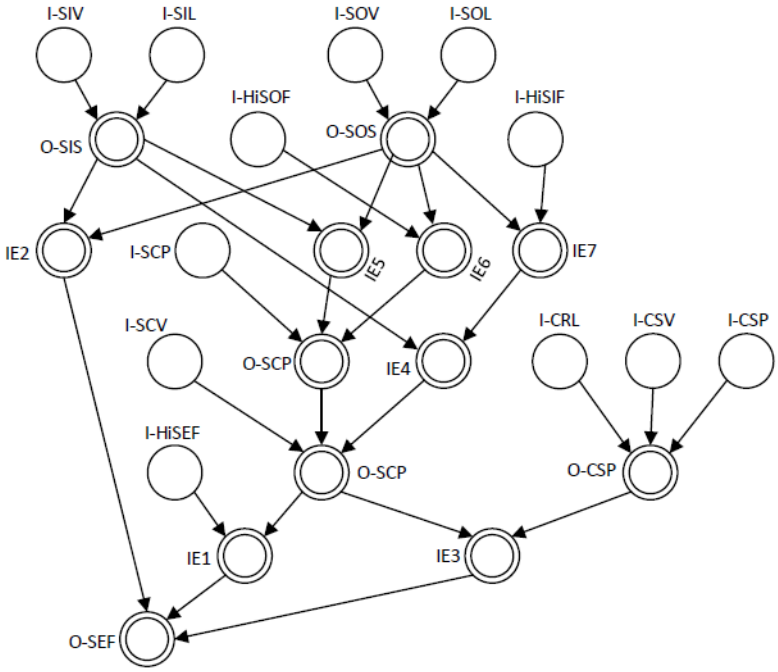


Fig. 18. Bayesian Network Model of the TFT in Fig. 16

Table 3. Prior probability of the root nodes for *N=4* and *t=5000* hours

Basic Events	State 0	State 1	State 2	State 3	State 4
I-SCP	0.7467	0.0704	0.0655	0.0609	0.0565
I-CSP	0.7467	0.0704	0.0655	0.0609	0.0565
I-SOV	0.0003	0.8739	0.1102	0.0139	0.0017
I-SIV	0.0003	0.8739	0.1102	0.0139	0.0017
I-CSV	0.0003	0.8739	0.1102	0.0139	0.0017
I-SCV	0.0003	0.8739	0.1102	0.0139	0.0017
I-CRL	0.9890	0.0028	0.0027	0.0028	0.0027
I-HiSOF	0.8159	0.0496	0.0471	0.0448	0.0426
I-HiSIF	0.8159	0.0496	0.0471	0.0448	0.0426
I-HiSEF	0.8159	0.0496	0.0471	0.0448	0.0426
I-SIL	0.0003	0.8739	0.1102	0.0139	0.0017
I-SOL	0.8471	0.0406	0.0390	0.0374	0.0359

Reliability analysis of the AFDS is performed by running queries on both the PN and BN models. Fig. 19 shows the comparison of the system unreliability estimated by the two approaches with different mission times. As seen in the figure, although the PN and BN-based approaches model time in continuous and discrete domains respectively, the system unreliability estimated by the approaches are very close to each other.

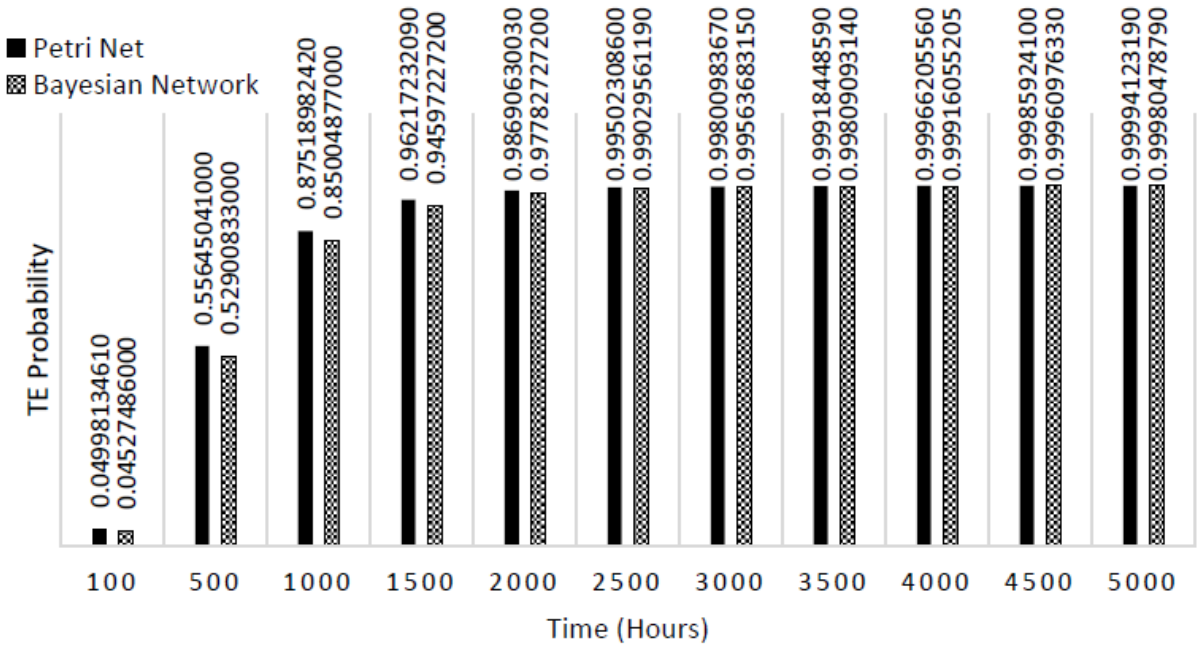


Fig. 19. Comparison of TE probability estimated by PN and BN-based methods

Criticality of the basic events listed in Table 2 is calculated by using both PN and BN-based approaches following the procedure described in section 3.3. According to both approaches I-CSV is the most critical basic event of the TFT in Fig. 16, the next most critical basic event is I-SOV, and the third most critical basic event is I-SCV. This gives an indication to the designer of the system where to put more design efforts to increase reliability of the system. For example, the event I-CSV corresponds to an internal failure of CSV (central starboard valve) in the CR subsystem and I-SOV corresponds to the valve SOV (starboard outer valve) in the SOS subsystem which itself is a part of SF subsystem. Therefore, if the analysts want to improve the reliability of the system then they may consider replacing the above mentioned critical components using components with higher reliability or they may consider introducing redundant components in parallel with the critical components.

Diagnostic analysis of the system was performed by observing the BN node *O-SEF* in Fig. 18 to be in failed state. This depicts the scenario where the analyst has the evidence that the top event of the TFT has occurred. Based on this evidence the analyst's beliefs about the prior probability of the basic events have been updated and the posterior failure probability values of the basic events are shown in Table 4. Completely new predictive analysis could be done based on these updated beliefs regarding the failure probability of the basic events.

Table 4. Posterior probability of the basic events

Basic Events	State 0	State 1	State 2	State 3	State 4
I-SCP	0.7467	0.0704	0.0655	0.0609	0.0565
I-CSP	0.7467	0.0704	0.0655	0.0609	0.0565
I-SOV	0.0003	0.8739	0.1102	0.0139	0.0017
I-SIV	0.0003	0.8739	0.1102	0.0139	0.0017
I-CSV	0.0001	0.8741	0.1102	0.0139	0.0017
I-SCV	0.0003	0.8739	0.1102	0.0139	0.0017
I-CRL	0.9890	0.0028	0.0027	0.0028	0.0027
I-HiSOF	0.8159	0.0496	0.0471	0.0448	0.0426
I-HiSIF	0.8159	0.0496	0.0471	0.0448	0.0426
I-HiSEF	0.8159	0.0496	0.0471	0.0448	0.0426
I-SIL	0.0003	0.8739	0.1102	0.0139	0.0017
I-SOL	0.8471	0.0406	0.0390	0.0374	0.0359

5. Discussion

Model-based analysis can reduce the effort needed for the design of dependable systems by integrating architectural and safety models. This integration also supports greater automation of dependability analysis processes. As an example of an automated MBDA technique, HiP-HOPS has been successfully used in complex automotive and other systems in case studies e.g. by Volvo (Papadopoulos and Grante, 2005) and Daimler (Papadopoulos et al., 2001). The HiP-HOPS tool can automatically generate dependability artefacts such as fault trees and FMEA from large models in a matter of seconds. To extend the capabilities of HiP-HOPS to dynamic systems, the approaches proposed in this paper combine the advantages of the existing HiP-HOPS approach with the temporal analysis capabilities of Pandora. This combination allows users to model and analyse failure behaviour of dynamic systems in a more intuitive way.

In particular, the Petri Net and Bayesian network based approaches proposed in this paper for probabilistic evaluation Pandora TFTs enhance the capability of HiP-HOPS for quantitative analysis of dynamic systems. Furthermore, they are not limited solely to exponentially distributed failure data, which makes them applicable for wider range of systems. In addition to that, the approaches are robust and the results produced by them are more realistic in the sense that they not only capture the time-dependent behaviour of system components but also estimate system reliability by considering statistical dependence among the events, which is not considered in existing analytical approaches.

In order to fully automate both the qualitative and quantitative analysis process using HiP-HOPS, it is necessary to automate the TFT to Petri Net and/or Bayesian Network translation and subsequent analysis process. In our view, in the context of HiP-HOPS, it might be easier to use Bayesian Networks for the purpose of quantifying Pandora TFTs because the TFT to BN translation process is more straightforward (one-to-one) and this approach does not suffer from state space explosion. The resulting approach is therefore more practical for

MBDA of dynamic systems. Moreover, BN-based approach can also be used for diagnostic analysis of systems in addition to predictive analysis. However, as the BN-based approach uses a discrete model of time, the granularity of time discretisation must be decided upon as part of the transformation process.

On the other hand, as the PN-based method models time in a continuous domain, it does not have to consider the issue of granularity of time discretisation. But, in the context of HiP-HOPS, this approach is more expensive as the auto transformation from TFT to Petri Net is not straightforward. This approach also generally suffers from the state space explosion problem. Furthermore, during analysis using the PN-based method, the PN models are solved using Monte Carlo simulations, which are inherently time consuming. For these reasons, the PN-based method could limit the applicability of the HiP-HOPS for larger systems.

6. Conclusion

In this paper, we proposed two approaches for probabilistic evaluation of Pandora TFTs and showed how they can be integrating with compositional model-based dependability analysis via HiP-HOPS. These techniques represent an advancement to the quantification of Pandora TFTs and allow faster, automatic quantitative analysis of dynamic systems using HiP-HOPS. In the discussion that followed the case study, we explored the relative advantages of the two techniques in the context of HiP-HOPS.

The work presented in this paper has the potential to be combined with the other advanced features of HiP-HOPS, such as architectural optimisation, maintenance, safety requirement allocation, and safety case generation for dynamic systems. Work to implement these techniques as part of the HiP-HOPS software tool is ongoing. The public version of the tool does not yet incorporate these calculations but prototype implementations exist and have been used for the purposes of the case study It is envisioned that the software version of the approaches should be able to identify independent dynamic modules from other static modules of a fault tree. This partitioning will give the ability to use different solution techniques on different modules, i.e., use combinatorial solutions for static modules and state space solutions for dynamic modules, thus improving the scalability of the approaches. In turn, this will enable fast compositional synthesis of temporal fault trees by HiP-HOPS and then effective analysis of those fault trees in Pandora, enabling effective automation of dependability analysis of complex dynamic systems.

Acknowledgements

This work was partly funded by the DEIS H2020 project (Grant Agreement 732242).

References

Arnold, A., Point, G., Griffault, A., Rauzy, A., 2000. The AltaRica formalism for describing concurrent systems. Fundam. Informaticae 40, 109–124.

Bittner, B., Bozzano, M., Cavada, R., Cimatti, A., Gario, M., Griggio, A., Mattarei, C., Micheli, A., Zampedri, G., 2016. The xSAP safety analysis platform, in: International Conference on Tools and Algorithms for the Construction and Analysis of Systems. pp. 533–539.

Bobbio, A., Franceschinis, G., Gaeta, R., Portinale, L., 1999. Exploiting Petri Nets to Support Fault Tree Based Dependability Analysis, in: 8th International Workshops on Petri Nets and Performance Models. IEEE, Zaragoza, pp. 146–155.

Bobbio, A., Portinale, L., Minichino, M., Ciancamerla, E., 2001. Improving the analysis of dependable systems by mapping fault trees into Bayesian networks. Reliab. Eng. Syst. Saf. 71, 249–260. doi:10.1016/S0951-8320(00)00077-6

Boudali, H., Dugan, J.B., 2006. A Continuous-Time Bayesian Network Reliability Modeling, and Analysis Framework. IEEE Trans. Reliab. 55, 86–97.

Boudali, H., Dugan, J.B., 2005. A discrete-time Bayesian network reliability modeling and analysis framework. Reliab. Eng. Syst. Saf. 87, 337–349. doi:10.1016/j.ress.2004.06.004

Bouissou, M., Bon, J.-L., 2003. A new formalism that combines advantages of fault-trees and Markov models: Boolean logic driven Markov processes. Reliab. Eng. Syst. Saf. 82, 149–163. doi:10.1016/S0951-8320(03)00143-1

Bozzano, M., Cimatti, A., Lisagor, O., Mattarei, C., Mover, S., Roveri, M., Tonetta, S., 2015. Safety assessment of AltaRica models via symbolic model checking. Sci. Comput. Program. 98, 464–483. doi:10.1016/j.scico.2014.06.003

Bozzano, M., Villafiorita, A., 2007. The FSAP/NuSMV-SA Safety Analysis Platform. Int. J. Softw. Tools Technol. Transf. - Spec. Sect. Adv. Autom. Verif. Crit. Syst. 9, 5–24.

Cheok, M.C., Parry, G.W., Sherry, R.R., 1998. Use of importance measures in risk-informed regulatory applications. Reliab. Eng. Syst. Saf. 60, 213–226.

Chiacchio, F., Aizpurua, J.I., D'Urso, D., Compagno, L., 2017. Coherence region of the Priority-AND gate: Analytical and numerical examples. Qual. Reliab. Eng. Int. 1–9.

Codetta-Raiteri, D., 2005. The Conversion of Dynamic Fault Trees to Stochastic Petri Nets, as a case of Graph Transformation. Electron. Notes Theor. Comput. Sci. 127, 45–60.

Doguc, O., Ramirez-Marquez, J.E., 2009. A generic method for estimating system reliability using Bayesian networks. Reliab. Eng. Syst. Saf. 94, 542–550.

Dugan, J.B., Bavuso, S.J., Boyd, M.A., 1992. Dynamic fault-tree models for fault-tolerant computer systems. IEEE Trans. Reliab. 41, 363–377. doi:10.1109/24.159800

EAST-ADL Association, 2014. EAST-ADL V2.1.12 specification [WWW Document]. URL http://www.east-adl.info/Specification.html

Edifor, E., Walker, M., Gordon, N., 2012. Quantification of Priority-OR Gates in Temporal Fault Trees, in: Ortmeier, F., Daniel, P. (Eds.), Computer Safety, Reliability, and Security SE - 9, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 99–110.

Edifor, E., Walker, M., Gordon, N., Papadopoulos, Y., 2014. Using Simulation to Evaluate Dynamic Systems with Weibull or Lognormal Distributions, in: Proceedings of the Ninth International Conference on

Dependability and Complex Systems DepCoS-RELCOMEX. pp. 177–187.

ESI ITI GmbH, 2017. SimulationX [WWW Document]. URL http://www.simulationx.com/ (accessed 1.1.17).

Fanti, M.P., Mangini, A.M., Ukovich, W., Lesage, J., Viard, K., 2014. A Petri Net Model of an Integrated System for the Health Care at Home Management, in: IEEE International Conference on Automation Science and Engineering (CASE). pp. 582--587.

Feiler, P., Rugina, A., 2007. Dependability Modeling with the Architecture Analysis & Design Language (AADL). Technical Report, Carnegie Mellon University.

Fenelon, P., McDermid, J.A., 1993. An Integrated Toolset For Software Safety Analysis. J. Syst. Softw. 21, 279–290.

Fussell, J.B., Aber, E.F., Rahl, R.G., 1976. On the Quantitative Analysis of Priority-AND Failure Logic. IEEE Trans. Reliab. R-25, 324–326.

Helmer, G., Wong, J., Slagell, M., Honavar, V., Miller, L., Wang, Y., Wang, X., Stakhanova, N., 2007. Software fault tree and coloured Petri net – based specification , design and implementation of agent-based intrusion detection systems. Int. J. Inf. Comput. Secur. 1, 109–142.

Huang, Y., McMurran, R., Dhadyalla, G., Jones, R.P., 2008. Probability based vehicle fault diagnosis : Bayesian network method. J. Intell. Manuf. 19, 301–311. doi:10.1007/s10845-008-0083-7

Kabir, S., 2017. An overview of Fault Tree Analysis and its application in model based dependability analysis. Expert Syst. Appl. 77, 114–135. doi:10.1016/j.eswa.2017.01.058

Kabir, S., Papadopoulos, Y., Walker, M., Parker, D., Aizpurua, J.I., Lampe, J., Rüde, E., 2017. A model-based extension to HiP-HOPS for dynamic fault propagation studies, in: 5th International Symposium on Model-Based Safety and Assessment. pp. 163–178. doi:10.1007/978-3-319-64119-5_11

Kabir, S., Walker, M., Papadopoulos, Y., 2015. Quantitative evaluation of Pandora Temporal Fault Trees via Petri Nets. IFAC-PapersOnLine 48, 458–463. doi:10.1016/j.ifacol.2015.09.569

Kabir, S., Walker, M., Papadopoulos, Y., 2014. Reliability Analysis of Dynamic Systems by Translating Temporal Fault Trees into Bayesian Networks, in: Ortmeier, F., Rauzy, A. (Eds.), Model-Based Safety and Assessment, Lecture Notes in Computer Science. Springer International Publishing, Cham, pp. 96–109. doi:10.1007/978-3-319-12214-4_8

Kabir, S., Walker, M., Papadopoulos, Y., Rüde, E., Securius, P., 2016. Fuzzy temporal fault tree analysis of dynamic systems. Int. J. Approx. Reason. 77, 20–37. doi:10.1016/j.ijar.2016.05.006

Leveson, N.G., 1995. Safeware: System Safety and Computers. Addison-Wesley.

Marsan, M.A., Balbo, G., Conte, G., Donatelli, S., Franceschinis, G., 1996. Modeling With Generalized Stochastic Petri Nets. Wiley, West Sussex.

MathWorks, 2017. MATLAB SIMULINK [WWW Document]. URL

https://uk.mathworks.com/products/simulink.html (accessed 1.13.17).

Molloy, M.K., 1982. Performance analysis using stochastic Petri nets. IEEE Trans. Comput. c-31, 913–917.

Montani, S., Portinale, L., Bobbio, A., Codetta-Raiteri, D., 2008. Radyban: A tool for reliability analysis of dynamic fault trees through conversion into dynamic Bayesian networks. Reliab. Eng. Syst. Saf. 93, 922–932.

Murata, T., 1989. Petri Nets : Properties , Analysis and Applications. Proc. IEEE 77, 541–580.

Neil, M., Marquez, D., 2012. Availability modelling of repairable systems using Bayesian networks. Eng. Appl. Artif. Intell. 25, 698–704.

Neil, M., Tailor, M., Marquez, D., Fenton, N., Hearty, P., 2008. Modelling dependable systems using hybrid Bayesian networks. Reliab. Eng. Syst. Saf. 93, 933–939.

Papadopoulos, Y., Grante, C., 2005. Evolving car designs using model-based automated safety analysis and optimisation techniques. J. Syst. Softw. 76, 77–89. doi:10.1016/j.jss.2004.06.027

Papadopoulos, Y., Mcdermid, J., Sasse, R., Heiner, G., 2001. Analysis and synthesis of the behaviour of complex programmable electronic systems in conditions of failure. J. Reliab. Eng. Syst. Saf. 71, 229–247.

Papadopoulos, Y., Walker, M., Parker, D., Sharvia, S., Bottaci, L., Kabir, S., Azevedo, L., Sorokos, I., 2016. A synthesis of logic and bio-inspired techniques in the design of dependable systems. Annu. Rev. Control 41, 170–182. doi:10.1016/j.arcontrol.2016.04.008

Pearl, J., 1988. Probabilistic reasoning in intelligent systems: Networks of Plausible Inference. Morgan Kaufmann, San Francisco, California.

Reza, H., Pimple, M., Krishna, V., Hilde, J., 2009. A Safety Analysis Method Using Fault Tree Analysis and Petri Nets, in: 2009 Sixth International Conference on Information Technology: New Generations. IEEE, Las Vegas, pp. 1089–1094.

Torres-Toledan, J.G., Sucar, L.E., 1998. Bayesian networks for reliability analysis of complex systems, in: Progress in Artificial Intelligence—IBERAMIA 98. pp. 195–206.

Vesely, W.E., Stamatelatos, M., Dugan, J., Fragola, J., Minarick, J., Railsback, J., 2002. Fault tree handbook with aerospace applications, NASA office of safety and mission assurance, Washington DC.

Walker, M., 2009. Pandora: A Logic for the Qualitative Analysis of Temporal Fault Trees. PhD Thesis, University of Hull.

Walker, M., Bottaci, L., Papadopoulos, Y., 2007. Compositional Temporal Fault Tree Analysis, in: Proceedings of the 26th International Conference on Computer Safety, Reliability and Security (SAFECOMP'07). pp. 106–119.

Walker, M., Papadopoulos, Y., 2009. Qualitative temporal analysis: Towards a full implementation of the Fault Tree Handbook. Control Eng. Pract. 17, 1115–1125.

Zhang, X., Miao, Q., Fan, X., Wang, D., 2009. Dynamic fault tree analysis based on Petri nets, in: 8th International Conference on Reliability, Maintainability and Safety(ICRMS). IEEE, Chengdu, pp. 138–142.