

Formal Availability Analysis using Theorem Proving

Waqar Ahmed and Osman Hasan

School of Electrical Engineering and Computer Science
National University of Sciences and Technology (NUST), Islamabad, Pakistan
{waqar.ahmad,osman.hasan}@seecs.nust.edu.pk

Abstract. Availability analysis is used to assess the possible failures and their restoration process for a given system. This analysis involves the calculation of instantaneous and steady-state availabilities of the individual system components and the usage of this information along with the commonly used availability modeling techniques, such as Availability Block Diagrams (ABD) and Fault Trees (FTs) to determine the system-level availability. Traditionally, availability analyses are conducted using paper-and-pencil methods and simulation tools but they cannot ascertain absolute correctness due to their inaccuracy limitations. As a complementary approach, we propose to use the higher-order-logic theorem prover HOL4 to conduct the availability analysis of safety-critical systems. For this purpose, we present a higher-order-logic formalization of instantaneous and steady-state availability, ABD configurations and generic unavailability FT gates. For illustration purposes, these formalizations are utilized to conduct formal availability analysis of a satellite solar array, which is used as the main source of power for the Dong Fang Hong-3 (DFH-3) satellite.

Keywords: Higher-order Logic, Unavailability Fault Tree, Availability Block Diagram, Theorem Proving.

1 Introduction

Availability analysis is used to identify and assess the causes and frequencies of system failures. The outcomes of availability analysis play a vital role in ensuring failure-free operation of the given system. Due to the rapid increase in the usage of technological systems in safety and mission-critical domains, such as transportation and healthcare, the demand of their availability and thus availability analysis is also growing dramatically.

The first step, in the availability analysis, is the evaluation of basic metrics of reliability and maintainability, such as mean-time to failure (MTTF) [1], mean-time between failure (MTBF) [1] and mean-time to repair (MTTR) [1], at the individual *component level* of the given system. These metrics are then used to calculate the availability of each component of the system by using the reliability and the maintainability distributions, such as *Exponential* or *Weibull*, with failure and repair rates, $\lambda = \frac{1}{MTTF}$ and $\mu = \frac{1}{MTTR}$. The next step is the selection

of an appropriate availability modeling technique, such as Availability Block Diagrams (ABD) [2] and unavailability Fault Trees (FT) [2]. These techniques are the extension of traditionally used reliability modeling techniques, such as Reliability Block Diagram (RBD) [1] and Fault Tree (FT) [1], for availability analysis purposes. Besides these two techniques, Markov chains [3] have also been used for availability assessment. In practice, it provides much more detailed analysis compared to ABD and UFT. However, the major problem with the Markov chain based availability analysis is its exponential growth in the state-space as the system complexity increases [3]. For instance, consider the large Multistage Interconnection Networks (MINs) [3] that are mainly used in the supercomputers and multi-process systems to realize communication among thousands of processors. To conduct the Markov chain based availability analysis of a 8×8 MIN consisting of 16 switching elements, we need to consider 2^{16} possible states [3]. Although, we can somewhat reduce the number of states by taking appropriate assumptions but it can compromise the accuracy of the availability results [3]. On the other hand, ABD and UFT are intuitive and transparent methods that can be used to describe the availability of large and complex systems, like MINs [4]. The ABD and UFT based modeling techniques also allow us to estimate the availability of the given system at the *system level* and play a particularly useful role at the design stages of the system to scrutinize the design alternatives without building the actual system. Once an appropriate availability model is obtained then the next step is to perform the *system level* availability analysis of the model using an appropriate analysis technique.

Traditionally, simulation tools, such as ReliaSoft [5] and ASENT [6], are used to analyze the availability models. However, these techniques cannot be termed as accurate due to their inherent incompleteness and the involvement of pseudo-random numbers and numerical methods. Given the safety and financial-critical nature of many technological systems these days, a slight unavailability of such a system, at a particular instant, may lead to disastrous situations, including the loss of human lives or heavy financial setbacks. For instance, it is reported that the Amazon Web Service (AWS) suffered an unavailability for 12 hours, in April 21, 2011, causing hundreds of high-profile Web sites to go offline [7], which resulted in a loss of 66,240 US\$ per minute downtime of its services.

Model checking techniques have been used to overcome the above-mentioned limitations for conducting the reliability analysis (e.g., [8, 9]), which is in turn used to assess the failure free operation of a system in a given interval and is thus quite closely related to availability analysis. Stochastic Petri Nets (SPN) have also been utilized to formalize RBD and FT, which are then used to analyze the availability [10]. However, a major disadvantage of using these approaches is their inability to analyze large size systems. Moreover, the computation of probabilities in these methods [8, 9] involves numerical methods, which compromises the accuracy of the results. Leveraging upon the high expressiveness of higher-order logic and a recent formalization of probability theory [11], the higher-order-logic theorem prover HOL4 has been recently used for the formalization of Reliability Block Diagrams (RBD) [12, 13] and Fault trees (FT) [14]. These efforts clearly

indicate the effectiveness of using a higher-order-logic (HOL) theorem prover for conducting reliability and failure analysis and, in the current paper, we develop the reasoning support for availability analysis by extending the HOL4 formalizations of RBD and FT. It is important to note that our proposed approach of using HOL theorem proving for availability analysis is primarily based on deductive reasoning. The availability properties are verified by using sound reasoning process and it is supported by the fact that every new theorem is derived from already verified theorems [15]. Therefore, the analysis is much more rigorous and accurate compared to computer algebra systems (CAS), such as Mathematica [16], which simplify the given closed form expressions and returns the results in the form of symbolic expressions. This fact can be illustrate with this example that the simplification of the expression $\frac{(x^2-1)}{(x-1)}$ by CAS yields $(x + 1)$ without explicitly mentioning $(x \neq 1)$ [17]. On the other hand, HOL theorem prover cannot verify the same expression without this premise.

The main contribution of the paper is to formalize the ABD, unavailability FT gates and steady-state availability to develop a formal library of availability theory foundations. This library can then be used to model and analyze both component and system level availability properties of any system within the sound core of a theorem prover. The main challenge faced in this formalization, compared to our earlier formalizations related to reliability theory, was to introduce the notion of an availability event that is associated with each system component. Each one of these availability events consists of a sequence of multiple random variables that are functioning over time. In order to illustrate the effectiveness of our proposed formalization, we present a formal availability analysis of a satellite solar array [18, 19] that has been used as a main power source for the Dong Fang Hong-3 (DFH-3) satellite. In addition, we also provide some automated reasoning support for the availability analysis. This automation allows us to quantitatively compute the availability and unavailability of the DFH-3 satellite solar array from the given values of the failure and repair rates.

2 Probability and Reliability in HOL

Mathematically, a measure space is defined as a triple (Ω, Σ, μ) , where Ω is a set, called the sample space, Σ represents a σ -algebra of subsets of Ω , where the subsets are usually referred to as measurable sets, and μ is a measure with domain Σ . A probability space is a measure space (Ω, Σ, Pr) , such that the measure, referred to as the probability and denoted by Pr , of the sample space is 1. In the HOL formalization of probability theory [11], given a probability space p , the functions `space`, `subsets` and `prob` return the corresponding Ω , Σ and Pr , respectively. This formalization also includes the formal verification of some of the most widely used probability axioms, which play a pivotal role in formal reasoning about reliability properties. A random variable is a measurable function between a probability space and a measurable space. The measurable functions belong to a special class of functions, which preserves the property that the inverse image of each measurable set is also measurable. A measurable space

refers to a pair (S, \mathcal{A}) , where S denotes a set and \mathcal{A} represents a nonempty collection of sub-sets of S . Now, if S is a set with finite elements, then the corresponding random variable is termed as a discrete random variable otherwise it is called a continuous one.

Now, reliability $R(t)$ is defined as the probability of a system or component performing its desired task over certain interval of time and expressed mathematically in terms of random variable as $R(t) = Pr(X > t)$. This concept can be formalized in HOL4 as follows:

$\vdash \forall p \ X \ t. \text{Reliability } p \ X \ t = \text{distribution } p \ X \ \{y \mid \text{Normal } t < y\}$

where the variables $p : (\alpha \rightarrow \text{bool}) \# ((\alpha \rightarrow \text{bool}) \rightarrow \text{bool}) \# ((\alpha \rightarrow \text{bool}) \rightarrow \text{real})$, $X : (\alpha \rightarrow \text{extreal})$ and $t : \text{real}$ represent a probability space, a random variable and a real number respectively. The function **Normal** takes a real number as its inputs and converts it to its corresponding value in the *extended – real* data-type, i.e., it is the *real* data-type with the inclusion of positive and negative infinity. The function **distribution** takes three parameters: a probability space p , a random variable X and a set of *extended – real* numbers and outputs the probability of a random variable X that acquires all the values of the given set in probability space p .

3 Instantaneous and Steady-state Availabilities

The instantaneous or point availability $A_{inst}(t)$ of a system or component can be defined as the probability that the given system or component is properly functioning at a given time instant t . If there are no repairs required after the fault has occurred then the availability $A(t)$ is simply equal to the reliability $R(t)$ of the system. However, if the system or component requires repair, then the availability can be considered as the function of two random variables, i.e., $X_i = T_i + D_i$, where T_i is the working time in the i^{th} period and D_i is the repair time in the i^{th} period. If the time when a system starts working in the k^{th} period is $S_k = \sum_{i=1}^{k-1} X_i$ then the considered system is said to be available at time t when there exists a period such that $S_k \leq t < S_k + T_k$. Now, the corresponding availability event constituted by these random variables can be formalized in HOL4 as follows:

Definition 1: $\vdash \forall p \ X \ t. \text{avail_event } p \ L \ n \ t =$
 $\{x \mid \text{SIGMA } (\lambda a. \text{FST } (\text{EL } a \ L) \ x + \text{SND } (\text{EL } a \ L) \ x) (\text{count } n) \leq t \wedge$
 $t < \text{SIGMA } (\lambda a. \text{FST } (\text{EL } a \ L) \ x + \text{SND } (\text{EL } a \ L) \ x) (\text{count } n) +$
 $\text{FST } (\text{EL } n \ L) \ x\} \cap \text{p-space } p$

The above definition takes a probability space p , a list of random variable pairs L , representing the working and repair time random variables, a number n and a time variable t and returns the corresponding availability event. The function **SIGMA** takes an arbitrary function f and a set s and returns the sum of all the values obtained by applying the function f on each element of the given set. The HOL4 function **count** takes a number n and returns a set containing all the natural numbers less than the given number n . Similarly, the function **EL** takes an index variable and a list and retrieves the list element located at the given index number. The HOL4 functions **FST** and **SND** are primarily used to access the

first and second elements in a pair. Definition 1 models the corresponding event of the i^{th} working interval only. To cover all the working intervals, we take the union of these availability events, corresponding to the pairs of random variable in list L , in HOL4 as follows:

Definition 2: $\vdash \forall p L t. \text{union_avail_events } p L t =$
 $\text{BIGUNION (IMAGE (\lambda a. avail_event } p L a t) (\text{count (LENGTH L))})$

An interesting property of the availability event is that its probability, also known as instantaneous availability, is always greater or equal to the corresponding reliability, i.e., $R_{T_1}(t) \leq A_{inst}(t)$, where T_1 is the first time-to-work random variable. This property can be formally verified, based on Definitions 1 and 2, in HOL4 as follows:

Theorem 1: $\vdash \forall p t L. \text{prob_space } p \wedge (0 \leq t) \wedge \neg \text{NULL } L \wedge$
 $(\forall n. \text{avail_event } p L n t \in \text{events } p) \wedge$
 $(\forall a b. (a \neq b) \Rightarrow$
 $\text{DISJOINT (avail_event } p L a t) (\text{avail_event } p L b t)) \Rightarrow$
 $(\text{Reliability } p (\text{FST (HD } L)) t \leq \text{prob } p (\text{union_avail_events } p L t))$

The first two assumptions ensure that p is a valid probability space and time index t must be positive. The next two assumptions make sure that the given list of random variables must not be empty and the availability events are in the events space p . The last assumption ensures that the availability events are disjoint. The conclusion models the property that the instantaneous availability is always greater or equal to reliability. The function `Reliability` takes a probability space p , a random variable that is associated with the system or component and a time variable t and returns the reliability of the system or component [12].

Consider that the failure and repair random variables are exhibiting exponential distributions with failure and repair rates λ and μ , respectively, then the instantaneous availability at the component level can be expressed mathematically as follows [1]:

$$A_{inst}(t) = \frac{\mu}{\mu + \lambda} + \frac{\lambda}{\mu + \lambda} e^{-(\lambda + \mu)t} \quad (1)$$

where the failure and repair rates are the mean-time-to-failure (MTTF) and mean-time-to-repair (MTTR), i.e. $\lambda = \frac{1}{MTTF}$ and $\mu = \frac{1}{MTTR}$, which are basic metrics for reliability and maintainability, respectively.

Now, we can formalize the instantaneous availability, given in Equation 1, as follows:

Definition 3: $\vdash \forall p L m. \text{inst_avail_exp } p L m =$
 $\forall t. \text{prob } p (\text{union_avail_events } p L (\&t)) =$
 $\frac{\text{SND } m}{(\text{SND } m + \text{FST } m)} + \frac{\text{FST } m}{(\text{SND } m + \text{FST } m)} * \text{exp } (-(\text{SND } m + \text{FST } m) * \&t)$

where the variables `FST m` and `SND m` represent failure and repair rates, respectively.

The steady-state availability of any component, which reflects the long-term availability after the system becomes stable, can be evaluated by taking the limit

as t approaches infinity in Equation (1).

$$A_{steady} = \lim_{t \rightarrow \infty} A_{inst}(t) = \frac{\mu}{\mu + \lambda} \quad (2)$$

The above equation can be formally verified in HOL4 as follows:

Theorem 2: $\vdash \forall p \ L \ m. \text{prob_space } p \wedge (0 < \text{FST } m \wedge 0 < \text{SND } m) \wedge$
 $(\forall t. (\forall a \ b. a \neq b \Rightarrow$
 $\text{DISJOINT } (\text{avail_event } p \ L \ a \ t) (\text{avail_event } p \ L \ b \ t)) \wedge$
 $(\forall n. \text{avail_event } p \ L \ n \ t \in \text{events } p)) \wedge \text{inst_avail_exp } p \ L \ m \Rightarrow$

$$(\lim (\lambda t. \text{prob } p (\text{union_avail_events } p \ L \ (&t))) = \frac{\text{SND } m}{(\text{SND } m + \text{FST } m)})$$

The assumptions of the above theorem are quite similar to those used in Theorem 1. The proof of Theorem 2 is primarily based on the fact that the negative exponential function tends to zero as its exponent tends to infinity.

4 Availability Block Diagrams

Availability Block Diagram (ABD) are graphical structures that represent the system components and their interconnections in the form of blocks and connector lines, respectively. The system is termed as available, if at least one path of properly available components from the input to output exists.

The availability of a system with components connected in series is considered to be available at time instant t only if all of its components are available at time t , as depicted in Figure 1(a). If $A_{inst_i}(t)$ is a mutually independent event that represents the instantaneous availability of the i^{th} component of a serially connected system with N components at time instant t , then the steady-state availability of the complete system can be expressed as [20]:

$$\lim_{t \rightarrow \infty} Pr(\bigcap_{i=1}^N A_{inst_i}(t)) = \prod_{i=1}^N \left(\frac{\mu_i}{\mu_i + \lambda_i} \right) \quad (3)$$

The series ABD configuration can be formalized as:

Definition 4: $\vdash (\forall p. \text{series_struct } p \ [] = \text{p.space } p) \wedge$
 $(\forall p \ h \ t. \text{series_struct } p \ (h::t) = h \cap \text{series_struct } p \ t)$

The above function takes a list of events corresponding to the availability of individual components of the given system and the probability space p and returns the intersection of all of the elements in a given list and the whole probability space, if the given list is empty. Based on this definition, Equation (3) can be formally verified as follows:

Theorem 3: $\vdash \forall p \ L \ M. \text{(A1): prob_space } p \wedge \text{(A2): } (0 \leq t) \wedge$

$\text{(A3): } (\forall z. \text{MEM } z \ M \Rightarrow 0 < \text{FST } z \wedge 0 < \text{SND } z) \wedge$

$\text{(A4): } (\text{LENGTH } L = \text{LENGTH } M) \wedge$

$\text{(A5): } (\forall t'. \neg \text{NULL } (\text{union_avail_event_list } p \ L \ (&t'))) \wedge$

$\text{(A6): } (\forall z. \text{MEM } z \ (\text{union_avail_event_list } p \ L \ (&t'))) \Rightarrow z \in \text{events } p) \wedge$

$\text{(A7): mutual_indep } p \ (\text{union_avail_event_list } p \ L \ (&t')) \wedge$

$\text{(A8): inst_avail_exp_list } p \ L \ M \Rightarrow$

$$(\lim (\lambda t. \text{prob } p (\text{series_struct } p \ (\text{union_avail_event_list } p \ L \ (&t)))) = \text{list_prod } (\text{steady_state_avail_list } M))$$

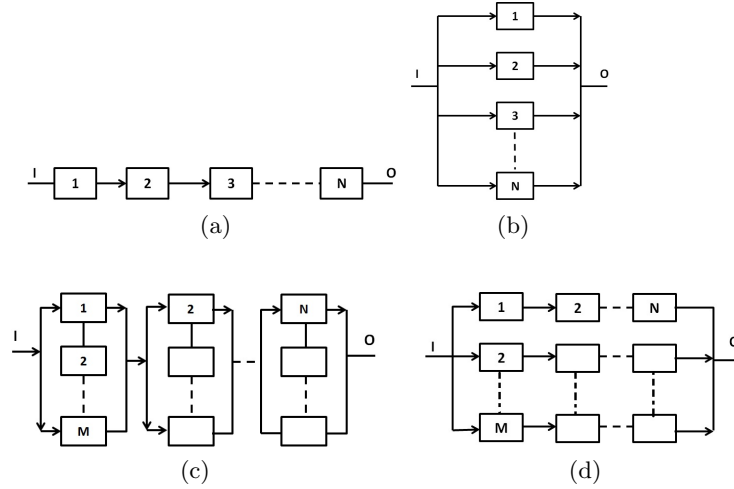


Fig. 1: ABDs (a) Series (b) Parallel (c) Series-Parallel (d) Parallel-Series

where the function `union_avail_event_list` can be obtained by mapping the function `union_avail_event` on every element of the given random variable list. The function `list_prod` returns the product of given real number list. The first two assumptions (A1-A2) ensure that p is a valid probability space and the time t must be positive. The assumptions (A3-A4) guarantee that the failure and repair rates are positive and the length of failure-repair random variable and the corresponding rate lists are equal. The next two assumptions (A5-A6) make sure that the length of availability event list, representing the availability of individual components, must not be empty and each availability event in a `avail_event_list` is in events space p . The last two assumptions (A7-A8) provide the mutual independence among all the availability events and the instantaneous availability of each component. The conclusion of the theorem represents Equation (3) as the function `steady_state_avail_list` takes a list of pairs, representing the failure and repair rates, and returns a list of steady-state availabilities, corresponding to each component of the given system.

Similarly, the availability of a system with parallel connected components, depicted in Figure 1(b), mainly depends on the component with the maximum availability. In other words, the system will continue functioning as long as at least one of its components remains functional. Mathematically [20]:

$$\lim_{t \rightarrow \infty} Pr\left(\bigcup_{i=1}^N A_{inst_i}(t)\right) = 1 - \prod_{i=1}^N \left(1 - \frac{\mu_i}{\mu_i + \lambda_i}\right) \quad (4)$$

Now, the availability of a system with a parallel structure is defined as:

Definition 5: $\vdash (\text{parallel_struct } [] = \{\}) \wedge$
 $(\forall h \ t. \text{parallel_struct } (h::t) = h \cup \text{parallel_struct } t)$

The function `parallel_struct` accepts a list of reliability events and returns the parallel structure reliability event by recursively performing the union op-

eration on the given list of reliability events or an empty set if the given list is empty. We can now verify Equation (4) as follows:

Theorem 4: $\vdash \forall p \text{ L M.}$

$$(\lim (\lambda t. \text{prob } p (\text{parallel_struct } p (\text{union_avail_event_list } p \text{ L } (\&t)))) = 1 - \text{list_prod } (\text{one_minus_list } (\text{steady_state_avail_list } M)))$$

The above theorem is verified under the same assumptions as Theorem 3. The conclusion of the theorem represents Equation (4) where, the function `one_minus_list` accepts a list of *real* numbers $[x_1, x_2, \dots, x_n]$ and returns the list of *real* numbers such that each element of this list is 1 minus the corresponding element of the given list, i.e., $[1 - x_1, 1 - x_2, \dots, 1 - x_n]$. The proof of Theorem 4 is based on Theorem 3 along with the fact that given a list of n mutually independent events, the complement of these n events are also mutually independent.

If in each serial stage the components are connected in parallel, as shown in Figure 1(c), then the configuration is termed as a series-parallel structure. If $A_{inst_{ij}}(t)$ is the event corresponding to the instantaneous availability of the j^{th} component connected in an i^{th} subsystem at time instant t , then the steady-state availability of the complete system can be expressed as follows [20]:

$$\lim_{t \rightarrow \infty} Pr(\bigcap_{i=1}^N \bigcup_{j=1}^M A_{inst_{ij}}(t)) = \prod_{i=1}^N (1 - \prod_{j=1}^M (1 - \frac{\mu_{ij}}{\mu_{ij} + \lambda_{ij}})) \quad (5)$$

By extending the ABD formalization approach, presented in Theorems 3 and 4, we formally verify the generic availability expression for series-parallel ABD configuration, given in Equation (5), in HOL4 as follows:

Theorem 5: $\vdash \forall p \text{ L M. prob_space } p \wedge (\text{LENGTH } L = \text{LENGTH } M) \wedge$
 $(\forall z. \text{MEM } z (\text{FLAT } M) \Rightarrow 0 < \text{FST } z \wedge 0 < \text{SND } z) \wedge$
 $(\forall n. n < \text{LENGTH } L \Rightarrow (\text{LENGTH } (\text{EL } n \text{ L}) = \text{LENGTH } (\text{EL } n \text{ M}))) \wedge$
 $(\forall t'. (\forall z. \text{MEM } z (\text{list_union_avail_event_list } p \text{ L } (\&t')) \Rightarrow \neg \text{NULL } z) \wedge$
 $(\forall z'. \text{MEM } z' (\text{FLAT } (\text{list_union_avail_event_list } p \text{ L } (\&t')))) \Rightarrow$
 $z' \in \text{events } p) \wedge$
 $\text{mutual_indep } p (\text{FLAT } (\text{list_union_avail_event_list } p \text{ L } (\&t')))) \wedge$
 $\text{two_dim_inst_avail_exp } p \text{ L M} \Rightarrow$
 $(\lim (\lambda t. \text{prob } p$
 $(\text{series_parallel_struct } p (\text{list_union_avail_event_list } p \text{ L } (\&t)))) =$
 $\text{list_prod } (\text{one_minus_list } (\text{MAP } (\lambda a. \text{compl_steady_state_avail } a) M)))$

where the function `list_union_avail_event_list` is obtained by mapping the function `union_avail_event_list` on each element of the given random variable list.

The function `series_parallel_struct` models the series-parallel ABD by first mapping the function `parallel_struct` on each element of the given event list and then applying the function `series_struct` to this obtained list. Similarly, the function `compl_steady_state_avail` returns a list of one minus steady-state availabilities.

The functions `list_prod` and `one_minus_list` are used to model the product and complement of steady-state availabilities, respectively. The assumptions are similar to the ones used in Theorems 3 and 4 with the extension that the given

lists are two-dimensional lists. The HOL4 function `FLAT` is used to convert a two dimensional list into a single list. The conclusion models the right-hand-side of Equation (5). The proof of the above theorem uses Theorems 3 and 4 and also requires a lemma that given the list of mutually independent reliability events, an event corresponding to the series-parallel structure and a reliability event are also independent in probability.

If the components in these reserved *subsystems* are connected serially then the structure is called a parallel-series structure, as depicted in Figure 1(d). If $A_{ij}(t)$ is the event corresponding to the availability of the j^{th} component connected in a i^{th} subsystem at time t , then the steady-state availability becomes:

$$\lim_{t \rightarrow \infty} Pr\left(\bigcup_{i=1}^M \bigcap_{j=1}^N A_{ij}(t)\right) = 1 - \prod_{i=1}^M \left(1 - \prod_{j=1}^N \frac{\mu_{ij}}{\mu_{ij} + \lambda_{ij}}\right) \quad (6)$$

The above equation is also verified as a HOL4 theorem in our development and more details about it can be found in [21].

5 Unavailability Fault Trees

Unavailability FT is a graphical technique consisting of internal nodes, which are represented by gates like OR, AND and XOR, and the external nodes, that model the unavailability events, which are associated with the occurrence of faults in components of the given system. The generic nature of these gates allows us to construct an efficient and accurate unavailability fault tree (FT) model for any given system. This FT can in turn be used to investigate the potential causes of a fault occurrence, which makes the system unavailable, and the calculation of minimal number of unavailability events, known as minimal cut-set (MCS), that contribute towards the occurrence of a *top event*, i.e., a critical event, which can cause the whole system unavailable upon its occurrence.

We can formalize the unavailability event of a system by taking the complement of the availability event with respect to the probability space p .

Definition 6: $\vdash \forall p \ \neg \ \mathbf{x} \ \mathbf{t}$.

`union_unavail_events p L t = p_space p DIFF union_avail_events p L t`

The instantaneous unavailability of the system can be expressed as follows:

$$\overline{A_{inst}}(t) = \frac{\lambda}{\mu + \lambda} - \frac{\lambda}{\mu + \lambda} e^{-(\lambda + \mu)t} \quad (7)$$

The HOL4 formalization of the above equation is as follows:

Definition 7: $\vdash \forall p \ \mathbf{L} \ \mathbf{m} \ \mathbf{inst_unavail_exp} \ \mathbf{p} \ \mathbf{L} \ \mathbf{m} =$







$\forall \mathbf{t} \ \mathbf{prob} \ \mathbf{p} \ (\mathbf{union_unavail_events} \ \mathbf{p} \ \mathbf{L} \ (\&\mathbf{t})) =$

$$\frac{\mathbf{FST} \ \mathbf{m}}{(\mathbf{SND} \ \mathbf{m} + \mathbf{FST} \ \mathbf{m})} - \frac{\mathbf{FST} \ \mathbf{m}}{(\mathbf{SND} \ \mathbf{m} + \mathbf{FST} \ \mathbf{m})} * \mathbf{exp} \ (- (\mathbf{SND} \ \mathbf{m} + \mathbf{FST} \ \mathbf{m}) * \&\mathbf{t})$$

If the occurrence of the unavailability event at the output is caused by the occurrence of all the input unavailability events then this kind of behavior can be modeled by using the AND unavailability FT gate, as shown in Table 1.

$$Pr\left(\bigcap_{i=2}^N \overline{A_{inst_i}}(t)\right) = \prod_{i=2}^N \frac{\lambda_i}{\lambda_i + \mu_i} \quad (8)$$

Table 1: HOL Formalization of Fault Tree Gates

Unavail. FT Gates	HOL Formalization
	$\vdash \forall p L t. \text{AND_unavail_FT_gate } p L t =$ $\text{inter_list } p (\text{union_unavail_event_list } p L t)$
	$\vdash \forall p L t. \text{OR_unavail_FT_gate } p L t =$ $\text{union_list } (\text{union_unavail_event_list } p L t)$
	$\vdash \forall p L1 L2 t. \text{NAND_unavail_FT_gate } p L1 L2 t =$ $\text{inter_list } p (\text{compl_list } p (\text{union_unavail_event_list } p L1 t)) \cap$ $\text{inter_list } p (\text{union_unavail_event_list } p L2 t)$
	$\vdash \forall p L t. \text{NOR_unavail_FT_gate } p L t =$ $p_space p \text{ DIFF } \text{union_list } (\text{union_unavail_event_list } p L t)$
	$\vdash \forall p A B. \text{XOR_FT_unavail_gate } p A B =$ $((p_space p \text{ DIFF } A \cap B) \cup (A \cap p_space p \text{ DIFF } B))$
	$\vdash \forall p A. \text{NOT_unavail_FT_gate } p A = (p_space p \text{ DIFF } A)$

The above equation can be formalized in HOL4 as follows:

Theorem 6: $\vdash \forall p L M. \text{prob_space } p \wedge$
 $(\forall z. \text{MEM } z M \Rightarrow 0 < \text{FST } z \wedge 0 < \text{SND } z) \wedge (\text{LENGTH } L = \text{LENGTH } M) \wedge$
 $(\forall t'. \neg \text{NULL } (\text{union_unavail_event_list } p L (\&t'))) \wedge$
 $(\forall z. \text{MEM } z (\text{union_unavail_event_list } p L (\&t'))) \Rightarrow z \in \text{events } p) \wedge$
 $\text{mutual_indep } p (\text{union_unavail_event_list } p L (\&t')) \wedge$
 $\text{inst_unavail_exp_list } p L M \Rightarrow$
 $(\lim (\lambda t. \text{prob } p (\text{AND_unavail_FT_gate } p (\text{union_avail_event_list } p L (\&t)))) =$
 $\text{list_prod } (\text{steady_state_unavail_list } M))$

The assumptions of the above theorem are similar to the ones used in Theorem 2 and the conclusion of Theorem 5 represents Equation (8).

In the OR unavailability FT gate, the occurrence of the output unavailability event depends upon the occurrence of any one of its input unavailability event. The function `OR_unavail_FT_gate`, given in Table 1, models this behavior as it returns the union of the input unavailability list L by using the recursive function `union_list`. The NOR unavailability FT gate, modeled by using the function `NOR_unavail_FT_gate`, given in Table 1, can be viewed as the complement of the OR unavailability FT gate and its output unavailability event occurs if none of the input unavailability event occurs.

Similarly, the NAND unavailability FT gate, represented by the function `NAND_unavail_FT_gate` in Table 1, models the behavior of the occurrence of an output unavailability event when at least one of the unavailability events at its input does not occur. This type of gate is used in unavailability FTs when the non-occurrence of the unavailability event in conjunction with the other unavailability events causes the top unavailability event to occur. This behavior can be expressed as the intersection of complementary and normal events, where the complementary events model the non-occurring unavailability events and the normal events model the occurring unavailability events. The output unavailability event occurs in the 2-input XOR unavailability FT gate if only one, and not both, of its input unavailability events occur. The HOL4 representation of the behaviour of the `XOR_unavail_FT_gate` is also presented in Table 1. The function `NOT_unavail_FT_gate` accepts an unavailability event A and probability space p and returns the complement to the probability space p of the given input unavailability event A . The verification of the corresponding unavailability expressions, of the above-mentioned unavailability FT gates, is presented in Table 2. These expressions are verified under the same assumptions as the ones used for Theorem 6 and the proofs are mainly based on some fundamental mutual independence properties of the given unavailability events along with some axioms of probability theory.

The principle of inclusion exclusion (PIE) forms an integral part of the reasoning involved in verifying the unavailability of a FT. In FT based unavailability analysis, firstly all the basic unavailability events are identified that can cause the occurrence of the system top unavailability event. These unavailability events are then combined to model the overall fault behavior of the given system by using the fault gates. These combinations of basic unavailability events, called cut sets, are then reduced to minimal cut sets (MCS) by using set-theory rules, such as idempotent, associative and commutative. The PIE is then used to evaluate the overall failure probability of the given system.

If \bar{A}_i represent the i^{th} basic unavailability event or a combination of unavailability events then the overall unavailability of the given system can be expressed in terms of the probabilistic inclusion-exclusion principle as follows:

$$\mathbb{P}\left(\bigcup_{i=1}^n \bar{A}_i\right) = \sum_{J \neq \{\}, J \subseteq \{1, 2, \dots, n\}} (-1)^{|J|-1} \mathbb{P}\left(\bigcap_{j \in J} \bar{A}_j\right) \quad (9)$$

The above equation has been formalized in HOL4 as follows [14]:

Theorem 7: $\vdash \forall p \text{ L t. prob_space } p \wedge$
 $(\forall x. \text{MEM } x \text{ (union_avail_event_list } p \text{ L t)} \Rightarrow x \in \text{events } p) \Rightarrow$
 $(\text{prob } p \text{ (union_list (union_avail_event_list } p \text{ L t))} =$
 $\text{sum_set } \{y \mid y \subseteq \text{set (union_avail_event_list } p \text{ L t)} \wedge y \neq \{\}\}$
 $(\lambda t. -1 \text{ pow (CARD } y - 1) * \text{prob } p \text{ (BIGINTER } y))$)

The function `sum_set` recursively sums the return value of the function f , which is applied on each element of the given set s . In the above theorem, the set s is represented by the term $\{x \mid C(x)\}$ that contains all the values of x , which satisfy condition C . Whereas, the λ abstraction function $(\lambda t. -1 \text{ pow (CARD } t - 1)$

Table 2: Unavailability Fault Tree Gates

Unavailability FT Gates	Conclusions of the formally verified Theorems
$\lim_{t \rightarrow \infty} \bar{A}_{OR}(t) = \lim_{t \rightarrow \infty} Pr\left(\bigcup_{i=1}^N A_{inst_i}(t)\right)$ $= 1 - \prod_{i=2}^N \left(1 - \frac{\lambda_i}{\lambda_i + \mu_i}\right)$	$\text{lim } (\lambda t. \text{ prob p}$ $\text{(OR_unavail_FT_gate p L \&t) =}$ $1 - \text{list_prod (one_minus_list}$ $\text{(steady_state_unavail_list M))}$
$\lim_{t \rightarrow \infty} \bar{A}_{NOR}(t) = 1 - \lim_{t \rightarrow \infty} \bar{A}_{OR}(t)$ $= \prod_{i=2}^N \left(1 - \frac{\lambda_i}{\lambda_i + \mu_i}\right)$	$\text{(lim } (\lambda t. \text{ prob p}$ $\text{(NOR_unavail_FT_gate p L \&t)) =}$ $\text{list_prod (one_minus_list}$ $\text{(steady_state_unavail_list M}$
$\lim_{t \rightarrow \infty} \bar{A}_{NAND}(t) =$ $\lim_{t \rightarrow \infty} Pr\left(\bigcap_{i=2}^k A_i(t) \cap \bigcap_{j=k}^N \bar{A}_j(t)\right) =$ $\prod_{i=2}^k \left(1 - \frac{\mu_i}{\mu_i + \lambda_i}\right) * \prod_{j=k}^N \frac{\lambda_j}{\mu_j + \lambda_j}$	$\text{(lim } (\lambda t. \text{ prob p}$ $\text{(NAND_unavail_FT_gate p L1 L2 t) =}$ $\text{list_prod (steady_state_avail M1) *}$ $\text{list_prod (steady_state_unavail_list M2}$
$\lim_{t \rightarrow \infty} \bar{A}_{XOR}(t) =$ $\lim_{t \rightarrow \infty} Pr(\bar{A}(t)B(t) \cup A(t)\bar{B}(t)) =$ $\left(1 - \frac{\lambda_1}{\lambda_1 + \mu_1}\right) * \frac{\lambda_2}{\lambda_2 + \mu_2} + \frac{\lambda_1}{\lambda_1 + \mu_1} * \left(1 - \frac{\lambda_2}{\lambda_2 + \mu_2}\right)$	$\text{(lim } (\lambda t. \text{ prob p}$ $\text{(XOR_unavail_FT_gate p A B \&t)) =}$ $\text{(1 - (steady_state_unavail M1)) *}$ $\text{(steady_state_unavail M2) +}$ $\text{(steady_state_unavail M1) *}$ $\text{(1 - (steady_state_unavail M2))}$
$\lim_{t \rightarrow \infty} \bar{A}_{NOT}(t) = Pr(A(t)) = \left(1 - \frac{\lambda}{\lambda + \mu}\right)$	$\text{lim } (\lambda t. \text{ prob p (NOT_FT_gate p A \&t) =}$ $\text{FST m / (FST m + SND m)}$

* `prob p (BIGINTER t)` models $(-1)^{|J|-1} \mathbb{P}(\bigcap_{j \in J} \bar{A}_j)$, such that the functions `CARD` and `BIGINTER` return the number of elements and the intersection of all the elements of the given set, respectively.

The proof script [21] of the above-mentioned formalizations of ABD and unavailability FT gates and the PIE principle is composed of more than 9000 lines of HOL script and took about 350 man-hours. The main outcome of this formalization is that the definitions and theorems of ABDs and FT gates can be used to capture the behavior of wide variety of real-world systems and analyze their corresponding availability in higher-order logic.

6 Application: Satellite Solar Arrays

As an illustrative application to demonstrate the effectiveness of our availability theory related formalization, we consider a solar array that has been used in the DFH-3 Satellite, which was launched by the People's Republic of China on May 12, 1997 [18, 19]. Solar arrays are one of the most vital components of the satellites because the mission success heavily depends upon the continuous reliable source of power. The satellite's solar array is a mechanical system, which

mainly consists of various mechanisms, including: deployable, synchronization, locking and orientation.

The solar array can be modeled by using series-parallel ABD configurations, shown in Figure 2, and based on the availability of its individual components, such as electric detonator (ED), the cutting knife (CK), the starting spring (SS), hing bearing (HB) and hing of locking mechanism (HL), the overall availability of the solar array can be evaluated [18]. The HOL4 formalization of the solar array ABD (Figure 2) is as follows:

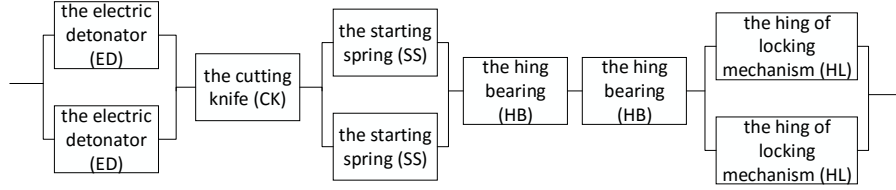


Fig. 2: Solar Array ABD

Definition 8: $\vdash \forall p \ X_ED \ X_CK \ X_SS \ X_HB \ X_HL \ t.$

```
RO_ABD p X_ED X_CK X_SS X_HB X_HL t =
series_parallel_struct p
  (list_union_avail_event_list
    ([ [X_ED;X_ED]; [X_CK]; [X_SS;X_SS]; [X_HB]; [X_HB]; [X_HL;X_HL] ]) t)
```

We verified the following theorem for the availability of the satellite solar array:

Theorem 8: $\vdash \forall p \ X_ED \ X_CK \ X_SS \ X_HB \ X_HL.$

```
(lim (λt. prob p ( RO_ABD p X_ED X_CK X_SS X_HB X_HL &t)) =
(1 - (1 - steady_state_avail ED) pow 2) * steady_state_avail CK *
(1 - (1 - steady_state_avail SS) pow 2) *
((steady_state_avail HB) pow 2) * (1 - (1 - steady_state_avail HL) pow 2))
```

We have omitted the assumptions of this theorem here due to space limitations and the complete formalization is available at [21]. The proof of the above theorem is primarily based on Theorem 5 and is very straightforward.

An unavailability FT can be constructed by considering the faults in the solar array mechanical components, which are the fundamental causes of satellite' solar array mechanisms failure. The unavailability FT for the solar array of the DFH-3 Satellite that was launched by the People's Republic of China on May 12, 1997 [19] is depicted in Figure 3 and we formally analyze this FT in this paper. The proposed FT formalization (functions `OR_unavail_FT_gate` and `AND_unavail_FT_gate`, given in Table 1) is used to model the MCS of the unavailability of the solar array as follows:

```
Definition 9:  $\vdash \forall p \ x1 \ x2 \ x3 \ x4 \ x5 \ x6 \ x7 \ x8 \ x9 \ x10 \ x11 \ x12 \ x13 \ x14 \ t.$ 
Solar_unavail_FT p x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x11 x12 x13 x14 t =
OR_unavail_FT_gate
  [OR_unavail_FT_gate (union_avail_event_list p [x1; x2; x3; x4] t);
   AND_unavail_FT_gate p (union_avail_event_list p [x5; x6] t);
   OR_unavail_FT_gate
     (union_avail_event_list p [x7; x8; x9; x10; x11; x12; x13; x14] t)]
```

The overall unavailability of a solar array can now be verified as follows:

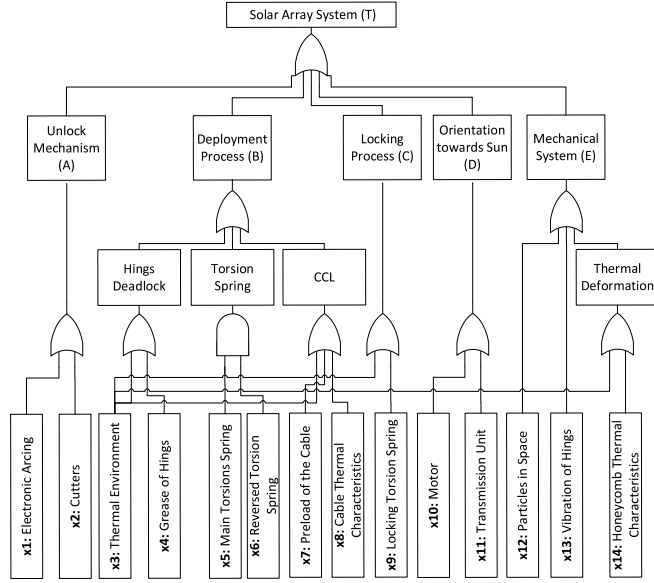


Fig. 3: Solar Array Unavailability FT

Theorem 9: $\vdash \forall p \ x1 \ x2 \ x3 \ x4 \ x5 \ x6 \ x7 \ x8 \ x9 \ x10 \ x11 \ x12 \ x13 \ x14.$

$(\text{lim}(\lambda t.$

$\text{Solar_unavail_FT } p \ x1 \ x2 \ x3 \ x4 \ x5 \ x6 \ x7 \ x8 \ x9 \ x10 \ x11 \ x12 \ x13 \ x14 \ \&t)) =$
 $1 - (\text{list_prod } (\text{steady_state_unavail_list } [x5;x6]) *$
 $(1 - \text{list_prod } (\text{one_minus_list } (\text{steady_state_unavail_list}$
 $[c1;c2;c3;c4;c6;c7;c8;c9;c10;c11;c12;c13;c14])))$

Again all quantifiers and the assumptions of the above theorem have not been included due to space limitations and the complete theorem can be found at [21]. The proof of the above theorem utilizes the PIE principle (Theorem 7) and the unavailability FT gates with their corresponding mathematical expression, given in Tables 1 and 2.

The proof script [21] for Theorems 8 and 9 is composed of about 100 lines of HOL code compared to about 9000 lines of code that had to be written to formalize the foundational availability concepts. This straightforward reasoning clearly indicates the usefulness of our work. The distinguishing features of the formally verified Theorems 8 and 9, compared to the other existing availability analysis alternatives, include their generic nature, i.e., all the variables are universally quantified and thus can be specialized to obtain the availability for any given failure and repair rates, and their guaranteed correctness due to the involvement of a sound theorem prover in their verifications. Moreover, the usage of a theorem prover in their verification ensures that all the required assumptions for the validity of the results are explicitly included in the theorems, which is quite important for designing accurate systems.

In order to facilitate the use of our formally verified results by industrial design engineers for their availability analysis, we have also developed a set

of SML scripts to automate the simplification step of these theorems for any given failure and repair rate values corresponding to the DFH-3 satellite solar array components. For instance, the `auto_solar_RBD_avail` script automatically computes the availability up to 12 decimal places based on Theorem 8 as follows:

```

⊢ prob_space p ∧
(∀t'. (∀z. MEM z (FLAT (list_union_avail_event_list
  [[X_ED;X_ED];[X_CK];[X_SS;X_SS];[X_HB];[X_HB];[X_HL;X_HL]] (&t')))) ⇒
z ∈ events p) ∧
mutual_indep p (FLAT
(list_union_avail_event_list
  [[X_ED;X_ED];[X_CK];[X_SS;X_SS];[X_HB];[X_HB];[X_HL;X_HL]] (&t')))) ∧
two_dim_inst_avail_exp p
[[X_ED;X_ED];[X_CK];[X_SS;X_SS];[X_HB];[X_HB];[X_HL;X_HL]]
[[ (0.1,0.3);(0.1,0.3)];[(0.2,0.5)]; [(0.3,0.4); (0.3,0.4)]; [(0.7,0.8)];
[(0.7,0.8)]; [(0.5,0.5); (0.5,0.5)]] ⇒
lim (λt. prob p (RO_ABD p X_ED X_CK X_SS X_HB X_HL &t)) = 0.116618075802

```

This `auto_solar_RBD_avail` script can be used for any values of the failure and repair rates and can be easily extended to be used for the instantiation of the generic result of Theorems 9 [21]. With a very little modification, these kind of automation scripts can facilitate industrial design engineers to accurately determine the availability of many other safety-critical systems.

7 Conclusion

The foremost requirements to conduct the formal availability analysis within a theorem prover is to formalize the ABD configurations, i.e., series, parallel, series-parallel and parallel-series, unavailability FT gates, such as AND, OR, NAND, NOR, XOR and NOT, and instantaneous and steady-state availability. This paper fulfills the above-mentioned requirement and thus provides a framework, which can be used to carry out the formal availability analysis of any system within a sound core of HOL4 theorem prover. For illustration, our formalizations are utilized to conduct the formal availability analysis of an satellite solar array and the results have been found to be more rigorous than the existing availability analysis alternatives. However, this formalization is only limited to static ABD and UFT models and cannot express the time varying system states, dependent systems and non-series-parallel topologies. This limitation can be removed by extending the present formalization to dynamic ABD and dynamic UFT. This can be done by combining this formalization of ABD and UFT with the recently proposed Markov chain formalization [22] in HOL4.

References

1. Trivedi, K.S.: Probability and Statistics with Reliability, Queuing and Computer Science Applications. 2nd edn. John Wiley and Sons Ltd. (2002)
2. Stapelberg, R.F.: Handbook of Reliability, Availability, Maintainability and Safety in Engineering Design. Springer Science & Business Media (2009)
3. Blake, J.T., Trivedi, K.S.: Multistage Interconnection Network Reliability. Transactions on Computers **38**(11) (1989) 1600–1604

4. Bistouni, F., Jahanshahi, M.: Analyzing the Reliability of Shuffle-exchange Networks using Reliability Block Diagrams. *Reliability Engineering & System Safety* **132** (2014) 97–106
5. ReliaSoft: <http://www.reliasoft.com/> (2016)
6. ASENT: <https://www.raytheonagle.com/asent/rbd.htm> (2016)
7. Bailis, P., Kingsbury, K.: The network is reliable. *Queue* **12**(7) (2014) 20
8. Robidoux, R., Xu, H., Xing, L., Zhou, M.: Automated Modeling of Dynamic Reliability Block Diagrams Using Colored Petri Nets. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans* **40**(2) (2010) 337–351
9. Bozzano, M., Cimatti, A., Katoen, J.P., Nguyen, V.Y., Noll, T., Roveri, M.: The COMPASS Approach: Correctness, Modelling and Performability of Aerospace Systems. In: *Computer Safety, Reliability, and Security*. Volume 5775 of LNCS. Springer (2009) 173–186
10. Signoret, J.P., Dutuit, Y., Cacheux, P.J., Folleau, C., Collas, S., Thomas, P.: Make your Petri Nets Understandable: Reliability Block Diagrams Driven Petri Nets. *Reliability Engineering & System Safety* **113** (2013) 61–75
11. Mhamdi, T., Hasan, O., Tahar, S.: On the Formalization of the Lebesgue Integration Theory in HOL. In: *Interactive Theorem Proving*. Volume 6172 of LNCS. Springer (2011) 387–402
12. Ahmed, W., Hasan, O., Tahar, S., Hamdi, M.S.: Towards the Formal Reliability Analysis of Oil and Gas Pipelines. In: *Intelligent Computer Mathematics*. Volume 8543 of LNCS. Springer (2014) 30–44
13. Ahmed, W., Hasan, O., Tahar, S.: Formal Reliability Analysis of Wireless Sensor Network Data Transport Protocols using HOL. In: *Wireless and Mobile Computing, Networking and Communications, IEEE* (2015) 217–224
14. Ahmed, W., Hasan, O.: Towards Formal Fault Tree Analysis Using Theorem Proving. In: *Conferences on Intelligent Computer Mathematics*. Volume 9150 of LNCS. Springer (2015) 39–54
15. Gordon, M., Melham, T.: *Introduction to HOL: A Theorem Proving Environment for Higher-Order Logic*. Cambridge Press (1993)
16. Mathematica: www.wolfram.com (2008)
17. Harrison, J., Théry, L.: Extending the HOL theorem prover with a computer algebra system to reason about the reals. In: *Higher Order Logic Theorem Proving and Its Applications*. Volume 780 of LNCS. Springer (1994) 174–184
18. Wu, H.C., Wang, C.J., Liu, P.: Reliability Analysis of Deployment Mechanism of Solar Arrays. *Applied Mechanics and Materials* **42** (2011) 139–142
19. Wu, J., Yan, S., Xie, L.: Reliability Analysis Method of a Solar Array by using Fault Tree Analysis and Fuzzy Reasoning Petri Net. *Acta Astronautica* **69**(11) (2011) 960–968
20. Ebeling, C.E.: *An Introduction to Reliability and Maintainability Engineering*. Tata McGraw-Hill Education (2004)
21. Ahmed, W.: Formalization of Availability Block Diagram and Unavailability FT. <http://save.seecs.must.edu.pk/availability/> (2016)
22. Liu, L.Y.: Formalization of Discrete-time Markov Chains in HOL. PhD thesis, Concordia University (2013)