Towards Formal Reliability Analysis of Logistics Service Supply Chains using Theorem Proving

Waqar Ahmed[1], Osman Hasan[1] and Sofiene Tahar[2]

[1] School of Electrical Engineering and Computer Science
National University of Sciences and Technology
Islamabad, Pakistan
{waqar.ahmad,osman.hasan}@seecs.nust.edu.pk
[2] Department of Electrical and Computer Engineering
Concordia University
Montreal, QC, Canada
tahar@ece.concordia.ca

Abstract

Logistics service supply chains (LSSCs) are composed of several nodes, with distinct behaviors, that ensure moving a product or service from a producer to consumer. Given the usage of LSSC in many safety-critical applications, such as hospitals, it is very important to ensure their reliable operation. For this purpose, many LSSC structures are modelled using Reliability Block Diagrams (RBDs) and their reliability is assessed using paper-and-pencil proofs or computer simulations. Due to their inherent incompleteness, these analysis techniques cannot ensure accurate reliability analysis results. In order to overcome this limitation, we propose to use higher-order-logic (HOL) theorem proving to conduct the RBD-based reliability analysis of LSSCs in this paper. In particular, we present the higher-order-logic formalizations of LSSC with different and same types of capacities. As an illustrative example, we also present the formal reliability analysis of a simple three-node corporation.

1 Introduction

Logistics service supply chain (LSSC) decisions are usually impossible to reverse, and their impact may span several decades. These decisions are very difficult to make given the involvement of several elements of uncertainty, such as changing demand patterns and weather conditions or failing components, associated with these decisions. On the other hand, the reliability of LSSCs, i.e., the ability to perform well when parts of the system fail, is very important as LSSCs are used in many safety-critical applications, such as medicine [14] and space logistics [18]. Moreover, ensuring that the inventory is delivered on time can be of great significance to many companies. Generally, the reliability of a LSSC can be increased by adding more redundancy in it but this choice eventually results in increasing the overall cost, which is also undesirable in many cases. Therefore, it is very important to judge the reliability of the LSSC and its associated cost before development [19]. This kind of reliability analysis is frequently based on Reliability Block Diagrams (RBDs) [23], which are graphical structures consisting of blocks and connectors (lines). The main idea is to represent the structure of the given LSSC in terms of an appropriate RBD [15]. Now, based on this RBD, the reliability characteristics of the overall system can be judged based on the failure rates of individual components, whereas the overall system failure happens if all the paths for successful execution fail.

Traditionally, the RBD-based analysis of LSSC has been done using paper-and-pencil proof methods and computer simulations. Due to the involvement of manual manipulation and simplification, paper-and-pencil proof methods are error-prone and the problem gets more severe while analyzing large LSSCs. Moreover, it is possible, in fact a common occurrence, that many key assumptions required for the analytical proofs are in the mind of the mathematician and are not documented. These missing assumptions are thus not communicated to the supply chain designers and are ignored in the LSSC implementations, which may also lead to erroneous designs. RBD-based computer simulators, such as ReliaSoft [20] and ASENT [5], generate samples from the exponential and Weibull random variables to model the reliabilities of the sub-modules of the given LSSC. This data is then manipulated using computer arithmetic and numerical techniques to compute the reliability of the complete LSSC. These software are more scalable than the paper-and-pencil proof methods. However, they cannot ensure absolute correctness as well due to the involvement of pseudo-random numbers and numerical methods.

Formal methods [10], which are computer based mathematical reasoning techniques, has been used to overcome the inaccuracy limitations of the paper-and-pencil proof methods and simulation for communication networks. The main idea behind the formal analysis of a system is to first construct a mathematical model of the given system using a state-machine or an appropriate logic and then use logical reasoning and deduction methods to formally verify that this system exhibits the desired characteristics, which are also specified mathematically using an appropriate logic. For instance, Petri nets have been used for the RBD based analysis of a LSSC [15]. The technique has been used to automatically evaluate the reliability of a few node corporations, but the analysis is not scalable for large systems due to the state-space explosion problem [10]. Moreover, generic mathematical RBD relationships cannot be verified using such state-based petri nets techniques, which limits the scope of this approach. Similarly, a Colored Petri Nets (CPN) based tool has been used to model dynamic RBDs (DRBDs) [21], which are used to describe the dynamic reliability behavior of systems. The CPN verification tools, based on model checking principles, are then used to verify behavioral properties of the DRBDs models to identify design flaws [21]. However, due to the state-based model, only state related property verification, like deadlock checks, is supported by this approach and generic reliability relationships cannot be verified.

Higher-order logic [7] is a system of deduction with a precise semantics and can be used to formally model any system that can be described mathematically including recursive definitions, random variables, RBDs, and continuous components. Similarly, interactive theorem provers are computer based formal reasoning tools that allow us to verify higher-order-logic properties under user guidance. The foremost requirement for reasoning about reliability related properties of a LSSC in a theorem prover is the availability of the higher-order-logic formalization of probability theory. Hurd's formalization of measure and probability theories [13] is a pioneering work in this regard. Building upon this formalization, most of the commonly-used continuous random variables [9] and some reliability theory fundamentals [11][1] have been formalized using the HOL theorem prover [22]. However, the foundational formalization of probability theory [13] only supports the whole universe as the probability space. This feature limits its scope in many aspects [16] and one of the main limitations, related to RBD-based analysis, is the inability to reason about multiple continuous random variables [9][11]. Some recent probability theory formalizations [16][12] allow using any arbitrary probability space that is a subset of the universe and thus are more flexible than Hurd's formalization of probability theory. Particularly, Mhamdi's probability theory formalization [16], which is based on extended-real numbers (real numbers including $\pm\infty$), has been recently used to reason about the RBD-based reliability analysis of a series pipelines structure [4] and failure analysis of satellite solar arrays [3], which involves multiple exponential random variables.

In this paper, given the involvement of several elements of continuous and random nature in LSSCs, we propose to conduct the formal RBD-based reliability analysis of a LSSC within the sound core of a higher-order-logic theorem prover [22]. For this purpose, we plan to build upon the recently proposed higher-order-logic formalization of series RBD, which has been used to conduct reliability analysis of simple oil and gas pipeline [4]. However, this foundational formalization of a series RBD [4] has limited scope and cannot be used to analyze the RBD model of a given LSSC due to the redundancies in these models. The main contribution of this paper is the extension of the series RBD formalization to series-parallel RBD configurations in order to model LSSC scenarios, including the cases when the capacities are different and of same types. For illustration purposes, the paper also presents the formal analysis of a simple LSSC that has been analysed using Petri Nets before [15]. Thanks to the sound reasoning process, the results obtained from the formal reliability analysis of the LSSC scenarios can help design engineers validating the reliability results that are generally obtained through traditional techniques. These accurately determined reliability results can bring many other benefits including trade-off studies for different LSSC designs in order to optimize reliability and cost.

The paper is organized as follows: Sections 2 and 3 present a brief description about the HOL theorem prover and the formalization of probability theory and random variables. Section 4 provides the RBD-based formalization of LSSC scenarios with different and same type of capacities in HOL. Section 5 presents the formal reliability analysis of a three node corporation LSSC by utilizing series and series-parallel RBD configurations. Finally, Section 6 concludes the paper.

2 HOL Theorem Prover

HOL is an interactive theorem prover developed at the University of Cambridge, UK, for conducting proofs in higher-order logic. It utilizes the simple type theory of Church [8] along with Hindley-Milner polymorphism [17] to implement higher-order logic. HOL has been successfully used as a verification framework for both software and hardware as well as a platform for the formalization of pure mathematics.

The HOL core consists of only 5 basic axioms and 8 primitive inference rules, which are implemented as ML functions. Soundness is assured as every new theorem must be verified by applying these basic axioms and primitive inference rules or any other previously verified theorems/inference rules.

Table 1 provides the mathematical interpretations of some frequently used HOL symbols and functions, which are inherited from existing HOL theories, in this paper.

3 Probability Theory and Random Variables

Based on the measure theoretic foundations, a probability space is defined as a triple (Ω, Σ, Pr), 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 Pr is a measure with domain Σ and is 1 for the whole sample space. In the HOL probability theory formalization [16], given a probability space p, the functions `space` and `subsets` return the corresponding Ω and Σ, respectively. Based on this definition, all basic probability axioms have been verified. Now, a random variable is a measurable function between a probability space and a measurable space, which essentially is a pair $(S, \mathcal{A})$, where S denotes a set and $\mathcal{A}$ represents a nonempty collection of sub-sets of S. A random variable is termed as discrete if S is a set with finite elements and continuous otherwise.

HOL Symbol	Standard Symbol	Meaning
$\wedge$	*and*	Logical *and*
$\vee$	*or*	Logical *or*
$\neg$	*not*	Logical *negation*
::	*cons*	Adds a new element to a list
++	*append*	Joins two lists together
`HD L`	*head*	Head element of list L
`TL L`	*tail*	Tail of list L
`EL n L`	*element*	n^{th} element of list L
`MEM a L`	*member*	True if a is a member of list L
λx.t	$\lambda x.t$	Function that maps x to $t(x)$
`SUC n`	$n+1$	Successor of a *num*
lim(λn.f(n))	$\lim_{n\to\infty} f(n)$	Limit of a *real* sequence f

Table 1: HOL Symbols and Functions

The probability that a random variable X is less than or equal to some value x, $Pr(X \leq x)$ is called the cumulative distribution function (CDF) and it characterizes the distribution of both discrete and continuous random variables. The CDF has been formalized in HOL as follows [4]:

```
⊢ ∀ p X x.  CDF p X x = distribution p X {y | y ≤ Normal x}
```

where the variables p, X and x 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 values of the given set in probability space p.

Now, reliability $R(t)$ is stated as the probability of a system or component performing its desired task over a certain interval of time t.

$$R(t) = Pr(X > t) = 1 - Pr(X \leq t) = 1 - F_X(t) \tag{1}$$

where $F_X(t)$ is the CDF. The random variable X, in the above definition, models the time to failure of the system and is usually modeled by the exponential random variable with parameter λ, which corresponds to the failure rate of the system. Based on the HOL formalization of probability theory [16], Equation (1) has been formalized as follows [4]:

```
⊢ ∀ p X x.  Reliability p X x = 1 - CDF p X x
```

The series RBD, presented in [4], is based on the notion of mutual independence of random variables, which is one of the most essential prerequisites for reasoning about the mathematical expressions for all RBDs. If N reliability events L_i are mutually independent then

$$Pr(\bigcap_{i=1}^{N} L_i) = \prod_{i=1}^{N} Pr(L_i) \tag{2}$$

This concept has been formalized as follows [4]:

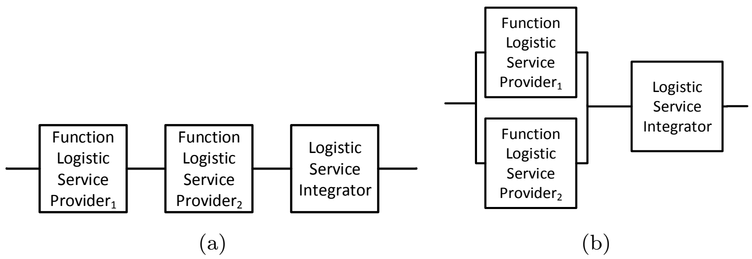

Figure 1: RBDs for the (a) Scenario with Different Types of Capacity (b) Scenario with the Same Type of Capacity

```
⊢ ∀ p L. mutual_indep p L = ∀ L1 n.  PERM L L1 ∧
1 ≤ n ∧ n ≤ LENGTH L ⇒
 prob p (inter_list p (TAKE n L1)) = list_prod (list_prob p (TAKE n L1))
```

The function `mutual_indep` accepts a list of events L and probability space p and returns $True$ if the events in the given list are mutually independent in the probability space p. The predicate `PERM` ensures that its two lists as its arguments form a permutation of one another. The function `LENGTH` returns the length of the given list. The function `TAKE` returns the first n elements of its argument list as a list. The function `inter_list` performs the intersection of all the sets in its argument list of sets and returns the probability space if the given list of sets is empty. The function `list_prob` takes a list of events and returns a list of probabilities associated with the events in the given list of events in the given probability space. Finally, the function `list_prod` recursively multiplies all the elements in the given list of real numbers. Using these functions, the function `mutual_indep` models the mutual independence condition such that for any 1 or more events n taken from any permutation of the given list L, the property $Pr(\bigcap_{i=1}^{N} L_i) = \prod_{i=1}^{N} Pr(L_i)$ holds.

4 Formalization of LSSC in HOL

A LSSC is essentially a service supply chain based on the ability logistics cooperation, which is generally required when the logistics service integrators face shortage in their capacity to deliver services to customers. At this stage, service integrators need to buy the logistics service capacity with functional logistics service providers. There could be a possible scenario where the type of capacity provided by the functional logistics service providers is of multiple (different) nature, such as transport and storage capacity. This scenario is modeled by using a series RBD configuration as shown in Figure 1(a) [15]. In case if the capacity type is the same then this scenario is modeled by using the series-parallel RBD configuration as depicted in Figure 1(b) [15].

In order to formalize the LSSC scenarios in HOL, we firstly present the formalization of series RBD and series-parallel RBD configurations, which are essentially utilized to conduct the reliability analysis of the LSSC, in HOL. If $A_i(t)$ is a mutually independent event that represents the reliable functioning of the i^{th} component of a serially connected system with N components at time t, then the overall reliability of the complete system is [6]:

$$R_{series}(t) = Pr(\bigcap_{i=1}^{N} A_i(t)) = \prod_{i=1}^{N} R_i(t) \quad (3)$$

The above equation can be utilized, by specifying $N = 3$, to evaluate the reliability of the LSSC for the first scenario by modeling it with a series RBD configuration consisting of three reliability blocks, as shown in Figure 1(a). Mathematically, it can be expressed as follows:

$$R_{LSSC\_fst\_scen} = R_{logis\_provdr1} * R_{logis\_provdr2} * R_{logis\_integr} \quad (4)$$

We formalized the corresponding LSSC first scenario series RBD configuration in HOL as:

Definition 1: `⊢ ∀ p logis_provdr1 logis_provdr2 logis_integr.`
```
LSSC_series_RBD p [logis_provdr1;logis_provdr2;logis_integr] =
inter_list p [logis_provdr1;logis_provdr2;logis_integr]
```

The function `LSSC_series_struct` takes a list of events corresponding to the failure of LSSC system components, i.e., $logis\_provdr1$, $logis\_provdr2$ and $logis\_integr$, and the probability space p and returns the series structure event of the complete LSSC system. The function `inter_list` returns the intersection of all of the elements of the given list and the whole probability space, if the given list is empty.

We formally verified the reliability expression for the first scenario, given in Equation 4, representing different capacity types, shown in Figure 1(a), in HOL as follows:

Theorem 1: `⊢ ∀ p logis_provdr1 logis_provdr2 logis_integr.  prob_space p ∧`
```
(∀ x'.  MEM x' [logis_provdr1;logis_provdr2;logis_integr] ⇒
x' ∈ events p) ∧
mutual_indep p [logis_provdr1;logis_provdr2;logis_integr] ⇒
 prob p (LSSC_series_struct p [logis_provdr1;logis_provdr2;logis_integr] =
 list_prod (list_prob p [logis_provdr1;logis_provdr2;logis_integr])
```

The first assumption ensures that p is a valid probability space based on the probability theory in HOL [16]. The next two assumptions guarantee that the list of events, representing the reliability of LSSC components, must be in the events space p and the reliability events are mutually independent. The conclusion of Theorem 1 models the series RBD configuration of LSSC first scenario with different capacity.

Similarly, in the series-parallel RBD configuration, if $A_{ij}(t)$ is the event corresponding to the reliability of the j^{th} component connected in a i^{th} subsystem at time t, then the reliability of the complete system can be expressed as follows:

$$R_{series-parallel}(t) = Pr(\bigcap_{i=1}^{N} \bigcup_{j=1}^{M} A_{ij}(t)) = \prod_{i=1}^{N}(1 - \prod_{j=1}^{M}(1 - R_{ij}(t))) \quad (5)$$

The above equation can be used to obtain the reliability of LSSC for the second scenario, which is modeled by a series-parallel RBD configuration, as shown in Figure 1(b). Mathematically, the reliability of this second scenario is as follows:

$$R_{LSSC\_snd\_scen} = (1 - (1 - R_{logis\_provdr1}) * (1 - R_{logis\_provdr2})) * (1 - (1 - R_{logis\_integr})) \quad (6)$$

The HOL formalization of Equation 6 is as follows:

Definition 2: ⊢ ∀ p logis_provdr1 logis_provdr2 logis_integr.
LSSC_series_parallel_struct p [[logis_provdr1;logis_provdr2];logis_integr]=
series_struct p (parallel_struct_list
[[logis_provdr1;logis_provdr2];logis_integr])

The function `LSSC_series_parallel_struct` accepts a two dimensional list, i.e., a list of lists, along with a probability space p and returns the corresponding reliability event of the system constituted from the series connection of the parallel stages. The function `series_struct` is used to model the series connection while the function `parallel_struct_list` is used to model the parallel stages. The function `parallel_struct_list` takes a two dimensional list of events along with probability space p and returns a single dimensional list of events by mapping the `inter_list` function, already explained in Definition 1, on each element of the given two dimensional event list.

Now, the reliability expression for the series-parallel RBD configuration of the LSSC, which corresponds to the second scenario with same capacity type, given in Equation 6, can be verified as the following HOL theorem:

Theorem 2: ⊢ ∀ p logis_provdr1 logis_provdr2 logis_integr. (prob_space p) ∧
mutual_indep p FLAT([[logis_provdr1;logis_provdr2];logis_integr]) ∧
(∀x'. MEM x' [logis_provdr1;logis_provdr2;logis_integr]) ⇒
x' ∈ events p) ⇒
prob p
(LSSC_series_parallel_struct p [[logis_provdr1;logis_provdr2];logis_integr] =
list_prod (one_minus_list
(list_compl_rel_list_prod p [[logis_provdr1;logis_provdr2];logis_integr]))

where $logis\_provdr1$, $logis\_provdr2$ and $logis\_integr$ are the reliability events associated with the logistic service providers and integrator, respectively. The function `list_compl_rel_list_prod` accepts a two-dimensional list of events, representing the time to failure of individual components connected in a series-parallel structure along with the probability space p and returns a list, which is the product of complement reliabilities of the components connected in parallel. The functions `list_prod`, `one_minus_list` and `list_prob` are used to model the product of reliabilities, complement of reliabilities, and the events corresponding to the component functioning reliably at the desired time, respectively. The assumptions of Theorem 2 are similar to the ones used in Theorem 1.

5 Case Study: A Three Node Corporation LSSC

In order to formally verify the reliability expression of a LSSC used in a typical three node corporation, we first need to formally model the reliability events that are associated with its logistic service providers and integrator. A reliability event list constructed from the list of random variables can be formalized in HOL is as follows:

Definition 3:
```
⊢ ∀ p x. rel_event_list p [] x = [] ∧
∀ p x h t.rel_event_list p (h::t) x =
PREIMAGE h {y | Normal x < y} ∩ p_space p :: rel_event_list p t x
```

The function `rel_event_list` accepts a probability space p, a list of random variables, representing the failure time of individual components, and a real number x, which represents the time index at which the reliability is desired. It returns a list of events, representing the proper functioning of all individual components at time x.

Definition 4:
```
⊢ ∀ p L x. List_rel_event_list p L x =
        MAP (λa. rel_event_list p a x) L
```

The function `List_rel_event_list` accepts a probability space p, a list of random variables, representing the failure time of individual components, and a real number x, which represents the time index at which the reliability is desired. It returns a two dimensional list of events by mapping the function `rel_event_list` on every element of the given two dimensional list of random variables, which in turn models the proper functioning of all individual components at time x.

We consider that the reliability of each LSSC component connected in RBD configurations, as shown in Figure 1, is exponential distributed. The HOL formalization of the exponential distribution predicate, which models the failure behavior of LSSC components, is as follows:

Definition 5:
```
⊢ ∀ p X l. exp_dist p X l =
∀ x. (CDF p X x = if 0 ≤ x then 1 - exp (-l * x) else 0)
```

The function `exp_dist` guarantees that the CDF of the random variable X is that of an exponential random variable with a failure rate l in a probability space p. We classify a list of exponentially distributed random variables based on this definition as follows:

Definition 6:
```
⊢ ∀ p L. list_exp p [] L = T ∧
∀ p h t L. list_exp p (h::t) L = exp_dist p (HD L) h ∧ list_exp p t (TL L)
```

The function `list_exp` accepts a list of failure rates, a list of random variables L and a probability space p. It guarantees that all elements of the list L are exponentially distributed with the corresponding failure rates, given in the other list, within the probability space p. For this purpose, it utilizes the list functions `HD` and `TL`, which return the *head* and *tail* of a list, respectively. Next we model a two dimensional list of exponential distribution functions to model nodes connected in a series-parallel RBD as follows:

Definition 7:
```
⊢ (∀ p L. list_list_exp p [] L = T) ∧
   ∀ h t p L. list_list_exp p (h::t) L =
              list_exp p h (HD L) ∧ list_list_exp p t (TL L)
```

The function `list_list_exp` accepts two lists, i.e., a two dimensional list of failure rates and random variables L, corresponding to the components at each stage of a series-parallel RBD. It calls the function `list_exp` recursively to ensure that all elements of the list L are exponentially distributed with the corresponding failure rates, given in the other list, within the probability space p.

The reliability of the first scenario of LSSC, modeled by a series RBD configuration and each component reliability is represented by exponential distribution, can be expressed as:

$$R_{LSSC\_fst\_scen}(t) = e^{(\lambda_{logis\_provdr1} + \lambda_{logis\_provdr2} + \lambda_{logis\_integr})t} \tag{7}$$

where the λ terms in the above equation represent the failure rates of logistic service providers and integrators.

Now, based on Equation (7), we carried out the formal reliability analysis of the first scenario of LSSC, given in Figure 1(a), in HOL and the resulting theorem is as follows:

Theorem 3: ⊢ ∀ X_logis_provdr1 X_logis_provdr2 X_logis_integr C_logis_provdr1 C_logis_provdr2 C_logis_integr p t.

```
0 ≤ t ∧ prob_space p ∧
(∀x'.  MEM x'
  rel_event_list p [X_logis_provdr1;X_logis_provdr2;X_logis_integr] t ⇒
x' ∈ events p) ∧
mutual_indep p
  (rel_event_list p [X_logis_provdr1;X_logis_provdr2;X_logis_integr] t) ∧
list_exp p [C_logis_provdr1;C_logis_provdr2;C_logis_integr]
  [X_logis_provdr1;X_logis_provdr2;X_logis_integr] ⇒
prob p (series_struct p
  (rel_event_list p [X_logis_provdr1;X_logis_provdr2;X_logis_integr] t) =
exp (-list_sum [C_logis_provdr1;C_logis_provdr2;C_logis_integr]*t)
```

where the function `list_sum` returns the sum of all the elements of the given failure rate list. The first assumption ensures that the variable `t` models time as it can acquire positive integer values only. The next assumption ensures that `p` is a valid probability space based on the probability theory in HOL [16]. The next two assumptions ensure that the events corresponding to the failures modeled, by the random variables $X\_logis\_provdr1$, $X\_logis\_provdr2$ and X_logis_integr are valid events from the probability space `p` and they are mutually independent. Finally, the last assumption assigns the random variables $X\_logis\_provdr1$, $X\_logis\_provdr2$ and X_logis_integr, as exponential random variables with failure rates $C\_logis\_provdr1$, $C\_logis\_provdr2$ and C_logis_integr, respectively. The conclusion of Theorem 3 represents the desired reliability expression.

Similarly, the reliability of the second scenario of LSSC with exponential failure distribution, shown in Figure 1(b), can be expressed as:

$$R_{LSSC\_snd\_scen}(t) = (1 - (1 - e^{(\lambda_{logis\_provdr1}t)} * (1 - e^{(\lambda_{logis\_provdr1}t)})) * (1 - (1 - e^{\lambda_{logis\_integr}t})) \tag{8}$$

We formally verified the above equation in HOL as follows:

Theorem 4: ⊢ ∀ X_logis_provdr1 X_logis_provdr2 X_logis_integr C_logis_provdr1 C_logis_provdr2 C_logis_integr p t.

```
(0 ≤ t) ∧ (prob_space p) ∧
mutual_indep p (FLAT
 (List_rel_event_list p [[X_logis_provdr1;X_logis_provdr2];X_logis_integr] t)) ∧
list_list_exp p([[C_logis_provdr1;C_logis_provdr2];C_logis_integr])
 ([[X_logis_provdr1;X_logis_provdr2];X_logis_integr]) ⇒
prob p (LSSC_series_parallel_struct p
 (list_rel_event_list p [[X_logis_provdr1;X_logis_provdr2];X_logis_integr] t)) =
list_prod (one_minus_list
 (list_exp_func_list ([[C_logis_provdr1;C_logis_provdr2];C_logis_integr]) t)
```

where the functions `list_prod` and `list_exp_func_list` accept a two-dimensional list of failure rates and return a list with products of one minus exponentials of every sub-list. For example, `list_exp_func_list [[c1; c2; c3]; [c4; c5]; [c6; c7; c8] x =` `[1 - exp -(c1+c2+c3) x; 1 - exp -(c4+c5) x; 1 - exp -(c6+c7+c8) x]`. The assumptions of Theorem 4 are quite similar to the ones used in Theorem 3. The proofs of Theorems 3 and 4 involves Theorems 1 and 2 and some basic probability theory axioms and some properties of the exponential function `exp`. The reasoning process took about 2000 lines of HOL script [2] with dedicated probability-theoretic guidance. The first LSSC scenerio reliability analysis is mainly carried out by using the series RBD formalization, which is presented in [4]. However, the major part of the effort was put into the formalization of generic series-parallel RBD configurations. This formalization facilitated the formalization of second scenario of LSSC, considerably as the analysis only took about 650 of HOL code.

The distinguishing features of the formally verified Theorems 3 and 4, compared to the reliability analysis of the LSSC scenarios of Figure 1 using Petri Nets [15], includes its generic nature, i.e., all the variables are universally quantified and thus can be specialized to obtain the reliability of any number of logistic providers and integrators for any given failures rates. The guaranteed correctness of the theorems is due to the involvement of a sound theorem prover in their verification, which ensures that all the required assumptions for the validity of the result are accompanying the theorems. To the best of our knowledge, the above-mentioned benefits are not shared by any other computer based reliability analysis approach.

6 Conclusions

The accuracy of reliability analysis of LSSC is a dire need these days due to their extensive usage in safety-critical applications, where an incorrect reliability estimate may lead to disastrous situations including the loss of innocent lives. In this paper, we presented a higher-order-logic formalization of commonly used RBD configurations, i.e., series and series-parallel, to facilitate the formal reliability analysis of LSSC within a theorem prover. The commonly used LSSC RBDs are also formalized and we illustrated the usefulness of the proposed idea by considering a small application. In future, we plan to formally analyze the reliability of larger LSSC models.

Acknowledgments

This publication was made possible by NPRP grant # [5 - 813 - 1 134] from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the author[s].

References

[1] N. Abbasi, O. Hasan, and S. Tahar. An Approach for Lifetime Reliability Analysis using Theorem Proving. *Journal of Computer and System Sciences*, 80(2):323–345, 2014.

[2] W. Ahmad. Towards Formal Reliability Analysis of Logistics Service Supply Chains using Theorem Proving, Proof Script. http://save.seecs.nust.edu.pk/projects/rbd/LSSC, 2015.

[3] W. Ahmad and O.Hasan. Towards the Formal Fault Tree Analysis using Theorem Proving. In *Conferences on Intelligent Computer Mathematics*, LNAI, pages 39–54. Springer, 2015.

[4] W. Ahmed, O. Hasan, S. Tahar, and M. S. Hamdi. Towards the Formal Reliability Analysis of Oil and Gas Pipelines. In *Intelligent Computer Mathematics*, volume 8543 of *LNCS*, pages 30–44. Springer, 2014.

[5] ASENT. https://www.raytheoneagle.com/asent/rbd.htm, 2015.

[6] R. Bilintion and R.N. Allan. *Reliability Evaluation of Engineering System*. Springer, 1992.

[7] C.E. Brown. *Automated Reasoning in Higher-order Logic*. College Publications, 2007.

[8] A. Church. A Formulation of the Simple Theory of Types. *Journal of Symbolic Logic*, 5:56–68, 1940.

[9] O. Hasan and S. Tahar. Formalization of the Continuous Probability Distributions. In *Automated Deduction*, volume 4603 of *LNAI*, pages 3–18. Springer, 2007.

[10] O. Hasan and S. Tahar. Formal Verification Methods. In *Encyclopedia of Information Science and Technology*, pages 7162–7170. IGI Global, 2014.

[11] O. Hasan, S. Tahar, and N. Abbasi. Formal Reliability Analysis using Theorem Proving. *IEEE Transactions on Computers*, 59(5):579–592, 2010.

[12] J. Holzl and A. Heller. Three Chapters of Measure Theory in Isabelle/HOL. In *Interactive Theorem Proving*, volume 6172 of *LNCS*, pages 135–151. Springer, 2011.

[13] J. Hurd. *Formal Verification of Probabilistic Algorithms*. PhD Thesis, University of Cambridge, UK, 2002.

[14] A.T. Kearny. Improving the Medical Supply Chain. `http://www.atkearneypas.com/knowledge/publications/2004/Medicines_Monograph_S.pdf`, 2004.

[15] Y. Li and H. Yi. Research on the Inherent Reliability and the Operational Reliability of the Supply Chain. *u- and e-Service, Science and Technology*, 7(1):104–112, 2014.

[16] T. Mhamdi, O. Hasan, and S. Tahar. On the Formalization of the Lebesgue Integration Theory in HOL. In *Interactive Theorem Proving*, volume 6172 of *LNCS*, pages 387–402. Springer, 2011.

[17] R. Milner. A Theory of Type Polymorphism in Programming. *Journal of Computer and System Sciences*, 17:348–375, 1977.

[18] MIT Strategic Engineering. Interplanetary Supply Chain Network for Space Exploration. http://strategic.mit.edu/spacelogistics/, 2015.

[19] D. Mu and Z. Du. Research on the Inherent Reliability and the Operational Reliability of the Supply Chain. *Logistics Technology*, 12(37), 2004.

[20] ReliaSoft. http://www.reliasoft.com/, 2015.

[21] R. Robidoux, H. Xu, L. Xing, and M. Zhou. 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):337–351, 2010.

[22] K. Slind and M. Norrish. A Brief Overview of HOL4. In *Theorem Proving in Higher-order Logics*, volume 5170 of *LNCS*, pages 28–32. Springer, 2008.

[23] J. Soszynska. Reliability and Risk Evaluation of a Port Oil Pipeline Transportation System in Variable Operation conditions. *International Journal of Pressure Vessels and Piping*, 87(2-3):81–87, 2010.