Formal Probabilistic Analysis of Cyber-Physical Transportation Systems

Atif Mashkoor[1] and Osman Hasan[2]

[1] Software Competence Center Hagenberg,
Hagenberg, Austria
{atif.mashkoor}@scch.at
[2] School of Electrical Engineering and Computer Science,
National University of Sciences and Technology,
Islamabad, Pakistan
{osman.hasan}@seecs.nust.edu.pk

Abstract. Formal specification and verification of cyber-physical transportation systems is inherently a complex task. A fail-safe specification of such systems not only includes intricate formalizations of assumptions and requirements but also a fine-grained analysis of their unpredictable and random components, at times at different levels of abstraction. Traditional techniques of verification and validation, such as simulation or model checking, do not cope very well with the posed challenges. In fact, sometimes it becomes merely impossible to guarantee certain properties, such as liveness, under all possible scenarios. We propose an approach based on higher-order logic for formal modelling and reasoning of cyber-physical transportation systems. In this approach, we express the unpredictable elements of the model by appropriate random variables. Instead of guaranteeing absolute correctness, these randomized models can then be used to formally reason about the probability or expectation of the system meeting its required specification. For illustration purposes, the paper presents a simple analysis of a vehicle platoon control algorithm.

1 Introduction

Automation is widely being practised nowadays in all modes of transportation, be it aviation, railway or automotive. The automation of such systems involves extensive data acquisitions from their environment, communication mechanisms to interact with the other members of the domain and enormous computations within their real-time embedded components for generating the required control signals. These specifications make them one of the most complex cyber-physical systems to design and verify.

The complexity related to transportation systems coupled with the enormous involvement of continuous and unpredictable physical aspects makes their verification a great challenge. Traditionally, their analysis is done using computer-aided design tools like Matlab. The main idea is to create a software model, capturing all the continuous and unpredictable details of the system, expressed in terms of Matlab functions. However, computer arithmetic cannot support infinite precision so the continuous physical realities are approximated by their closest discrete counter parts in terms of floating or fixed point numbers. Similarly, true randomness cannot be attained in computers and thus the random physical realities are approximated using pseudo random numbers in the Matlab models of the system. Once the system is modelled, the Matlab functions are analysed using computer simulation techniques, which deduce a property to be true by checking it for a number of test cases. This kind of testing is mainly utilized because exhaustive simulation, or testing for all possible combinations, is not feasible in terms of computational time and complexity for cyber-physical transportation systems. Thus, the simulation based analysis of Matlab models cannot be termed as accurate due to the inherent nature of simulation, and the usage of floating or fixed point numbers and pseudo random number generator based random variables in the Matlab models.

Transportation systems are among the top most operational safety-critical systems in the modern world to date. A faulty transportation system could result in disastrous consequences and may lead to the loss of human lives in worst cases. For example, the 2002 mid-air collision in Überlingen caused due to the human-controller interaction flaw, the 2008 frontal train collision caused due to an error in the warning system, and the 2009 crash of Air France Flight 447 that resulted in killing all 216 passengers and 12 crew members happened due to a flaw in the automatic reporting system, are the historic events that no one would like to be repeated. Therefore, given such a safety-critical nature of cyber-physical transportation systems, inaccurate analysis techniques, like simulation, should not be completely relied upon for their verification.

Formal methods are capable of conducting precise system analysis and thus can overcome the above mentioned limitations of simulation in the context of analysing cyber-physical transportation systems. The main principle behind formal analysis of a system is to construct a computer based mathematical model of the given system and formally verify, within a computer, that this model meets rigorous specifications of intended behaviour using mathematical reasoning.

Two of the most commonly used formal verification methods are model checking [4] and higher-order logic theorem proving [20]. Model checking is an automatic verification approach for systems that can be expressed as a finite-state machine. Higher-order logic theorem proving, on the other hand, is an interactive approach but is more flexible in terms of tackling a variety of systems. Both model checking and theorem proving have been successfully used for the precise functional correctness of a broad range of engineering and scientific systems, including some aspects of cyber-physical transportation systems. These days, certification authorities explicitly demand transportation systems to be safe, dependable and correctly implemented. In this realm, formal approaches to assure system safety, dependability and correctness are finding their way into being used as a certification argument (cf. DO-178C, IEC-61508, SIL-4).

However, in most of the existing work in the formal verification of cyber-physical transportation systems, a high level model of the actual system is considered such that the continuous and random physical realities are abstracted away. These analysis, even though are accurate due to the inherent soundness of the formal methods yet, cannot be considered as complete as the left out details could lead to potential failures. One of the main reasons for not considering the complete models of the cyber-physical systems is that most of the formal methods based analysis in this domain have been conducted by using either model checking or theorem proving with a decidable logic. These techniques, even though are automatic and thus user friendly yet, are not expressive enough to completely capture the continuous and random physical realities.

To address the above mentioned expressiveness problem, we propose to use higher-order logic theorem proving [16] for analysing cyber-physical transportation systems in this paper. Higher-order logic is a system of deduction with a precise semantics and due to its high expressiveness it can be used to model any system that can be expressed in a closed mathematical form. Higher-order logic has also been successfully used to develop some of the classical mathematical theories. Interactive theorem proving is the field of computer science and mathematical logic concerned with computer based formal proof tools that require some sort of human assistance.

The core of theorem provers usually consists of a handful of axioms and primitive inference rules. Soundness is assured as every newly added construct must adhere them. Powerful mathematical techniques such as induction and abstraction are the strengths of theorem proving and make it a very flexible verification technique. These distinguishing characteristics make higher-order logic theorem proving a very flexible verification technique that can be used to analyse any system with all of its continuous and physical details. For example, higher-order logic theorem proving has been used to successfully analyse continuous systems, such as optical waveguides [22], fractional order systems [35], real-time systems such as Stop-and-Wait protocol [7], systems with unpredictable and random elements such as reconfigurable memory arrays in the presence of stuck-at and coupling faults [23], and the wireless sensor network scheduling [12].

We propose to leverage upon the expressiveness of higher-order logic theorem proving to analyse cyber-physical transportation systems in this paper. In particular, the paper describes a framework for analysing their randomized aspects. The main idea is to model the randomness found in such systems in terms of formalized random variables in the higher-order logic model. These models can then be used to formally reason about the interesting probabilistic and statistical characteristics of the given system within the sound core of a theorem prover. Due to the undecidable nature of the underlying logic, the proofs would involve interaction, a cost that we pay to attain accurate probabilistic analysis results. However, these added efforts are justifiable given the safety-critical nature of transportation systems. Moreover, from past experiences, we expect the formal reasoning efforts to decrease with the availability of higher-order logic formalizations of cyber-physical systems as these available formal models and theorems can be re-utilized to formalize other variants of the same domain.

We have utilized HOL [21], a higher-order logic theorem prover for our work. Main reasons of its utilization are the availability of the underlying mathematical theories of probability and a possibility of a real analysis using its libraries.

The rest of the paper is organized as follows: After reviewing the related work in Section 2, we proceed by presenting some preliminaries including a brief introduction to higher-order logic theorem proving and the HOL theorem prover in Section 3. Next, Section 4 describes the proposed theorem proving based probabilistic analysis approach for the cyber-physical systems. This is followed by the simple analysis of a vehicle platoon control algorithm in Section 5. Finally, Section 6 concludes the paper.

2 Related Work

Due to inaccuracies introduced by the simulation based analysis methods, many researchers around the world are exploring the usage of formal methods for probabilistic analysis. Generally, probabilistic model checking is employed to asses the quantitative aspects of systems' safety and reliability. For example, probabilistic model checker PRISM [29] has been quite frequently used to evaluate the dependability and safety features of various systems (e.g., [28,15]). Probabilistic model checking involves the construction of a precise state-based mathematical model of the given probabilistic system. It is then subject to exhaustive analysis to formally verify that it satisfies a set of formally represented probabilistic properties. However, it can be used to analyse systems that can be expressed as probabilistic finite state machines only. Another major limitation of the probabilistic model checking approach is state space explosion. The state space of a probabilistic system can be very large, or sometimes even infinite. Thus, at the outset, it is impossible to explore the entire state space with limited resources of time and memory. Similarly, we cannot reason about mathematical expressions in probabilistic model checking. Thus, probabilistic model checking cannot be used to formally verify probability distributions or statistical characteristics, which are widely used parameters to assess the probabilistic correctness of a property.

A statistical model checker has been recently utilized to analyze some aspects of cyber-physical systems [9]. However, this approach also suffers from the classical model checking issues, like the state-space explosion and inability to reason about mathematical relations. Thus, the probabilistic model checking approach, even though is capable of providing exact solutions, is quite limited in terms of handling a variety of probabilistic analysis problems. Whereas higher-order logic theorem proving is capable of overcoming all the above mentioned problems, though at the cost of significant user interaction.

Formal methods, specifically B [1] and Event-B [2], have been extensively used in the development of transportation systems (e.g. [5,3]). Recently, these methods have been extended to allow the modelling and verification of probabilistic features. These extensions primarily use a probabilistic choice operator to probabilistically reason about certain termination conditions [17] and provide semantics of a Markov process to reason about some reliability issues [36]. However, these initiatives have a very limited scope. For example, they cannot be used to reason about generic mathematical expressions for probabilistic or statistical properties. Similarly, such formalisms are not mature enough to model and reason about all different kinds of continuous probability distributions. Given the continuous random nature of the transportation systems, both the probabilistic model checking and Event-B based techniques cannot be used to capture their behaviour and thus, the use of probabilistic and quantitative assessment for the formal verification of transportation systems is still few and far between.

The foremost criteria for conducting the formal probabilistic analysis in a theorem prover is to be able to express probabilistic notions, such as probability of an event and random variables, in higher-order logic and reason about the probability distribution and statistical properties of random variables in a higher-order logic theorem prover. A formalized probability theory provides the foundations for expressing probabilistic notions.

A number of authors, including Hurd [27], Mhamdi [32] and Hölze [26], reported higher-order logic based formalizations of probability theory. The recent works by Mhamdi [32] and Hölzl [26] are based on extended real numbers (including $\pm\infty$) and provide the formalization Lebesgue integral for reasoning about advanced statistical properties. This way, they are more mature than Hurd's [27] formalization of measure and probability theories, which is based on simple real numbers. However, these recent formalizations do not support a particular probability space like the one presented in Hurd's work. Due to this distinguishing feature, Hurd's formalization [27] has been utilized to verify sampling algorithms of a number of commonly used discrete [27] and continuous random variables [24] based on their probabilistic and statistical properties [24]. Due to the availability of a particular probability space as well as the formalization of probability and statistical properties, we build upon Hurd's formalization of measure and probability theories in this paper to analyse cyber-physical transportation systems.

Recently, a probabilistic kernels based mathematical approach [25] has been proposed for formalizing certain probabilistic safety claims. The mathematical framework has been illustrated using an example of a conflict detection system for an aircraft. Our proposed measure theoretic framework for probabilistic reasoning about cyber-physical systems is much more powerful in terms of handling a larger set of problems. Moreover, to the best of our knowledge, the mathematical framework of [25] has not been formalized in a theorem prover yet and thus the corresponding system analysis cannot be considered as completely sound. Our proposed formalization is based on the higher-order-logic formalization of measure and probability theories in HOL and thus the analysis are carried within the sound core of HOL theorem prover.

3 Preliminaries

In this section, we give a brief introduction to theorem proving in general and the HOL theorem prover in particular. The intent is to introduce the main ideas behind this technique to facilitate the understanding of this paper for the cyber-physical system community.

3.1 Theorem Proving

Theorem proving [16] is a widely used formal verification technique. The system that needs to be analysed is mathematically modelled in an appropriate logic and the properties of interest are verified using computer based formal tools. The use of formal logics as a modelling medium makes theorem proving a very flexible verification technique as it is possible to formally verify any system that can be described mathematically. The core of theorem provers usually consists of some well-known axioms and primitive inference rules. Soundness is assured as every new theorem must be created from these basic or already proved axioms and primitive inference rules.

The verification effort of a theorem in a theorem prover varies from trivial to complex depending on the underlying logic [18]. For instance, first-order logic [13] is restricted to propositional calculus and terms (constants, function names and free variables) and is semi-decidable. A number of sound and complete first-order logic automated reasoners are available that enable completely automated proofs. More expressive logics, such as higher-order logic [6], can be used to model a wider range of problems than first-order logic, but theorem proving for these logics cannot be fully automated and thus involves user interaction to guide the proof tools. For probabilistic analysis, we need to formalize (mathematically model) random variables as functions and their characteristics such as probability distribution properties and expectation, by quantifying over random variable functions. Henceforth, first-order logic does not support such formalization and we need to use higher-order logic to formalize probabilistic analysis.

3.2 HOL Theorem Prover

HOL is an interactive theorem prover developed by Mike Gordon at the University of Cambridge for conducting proofs in higher-order logic. It utilizes the simple type theory of Church [8] along with Hindley-Milner polymorphism [33] 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.

Secure Theorem Proving In order to ensure secure theorem proving, the logic in the HOL system is represented in the strongly-typed functional programming language ML [34]. An ML abstract data type is used to represent higher-order logic theorems and the only way to interact with the theorem prover is by executing ML procedures that operate on values of these data types. The HOL core consists of only 5 basic axioms and 8 primitive inference rules, which are implemented as ML functions.

Terms There are four types of HOL terms: constants, variables, function applications, and lambda-terms (denoted function abstractions). Polymorphism, types containing type variables, is a special feature of higher-order logic and is thus supported by HOL. Semantically, types denote sets and terms denote members of these sets. Formulas, sequences, axioms, and theorems are represented by using terms of Boolean types.

Theories A HOL theory is a collection of valid HOL types, constants, axioms and theorems, and is usually stored as a file in computers. Users can reload a HOL theory in the HOL system and utilize the corresponding definitions and theorems right away. The concept of HOL theory allows us to build upon existing results in an efficient way without going through the tedious process of regenerating these results using the basic axioms and primitive inference rules.

HOL theories are organized in a hierarchical fashion. Any theory may inherit types, definitions and theorems from other available HOL theories. The HOL system prevents loops in this hierarchy and no theory is allowed to be an ancestor and descendant of a same theory. Various mathematical concepts have been formalized and saved as HOL theories by the HOL users. These theories are available to a user when he first starts a HOL session. We utilized the HOL theories of Booleans, lists, sets, positive integers, *real* numbers, measure and probability in our work. In fact, one of the primary motivations of selecting the HOL theorem prover for our work was to benefit from these built-in mathematical theories.

Writing Proofs HOL supports two types of interactive proof methods: forward and backward. In forward proof, the user starts with previously proved theorems and applies inference rules to reach the desired theorem. In most cases, the forward proof method is not the easiest solution as it requires the exact details of a proof in advance. A backward or a goal directed proof method is the reverse of the forward proof method. It is based on the concept of a *tactic*; which is an ML function that breaks goals into simple sub-goals. In the backward proof method, the user starts with the desired theorem or the main goal and specifies tactics to reduce it to simpler intermediate sub-goals. Some of these intermediate sub-goals can be discharged by matching axioms or assumptions or by applying built-in decision procedures. The above steps are repeated for the remaining intermediate goals until we are left with no further sub-goals and this concludes the proof for the desired theorem.

The HOL theorem prover includes many proof assistants and automatic proof procedures [18] to assist the user in directing the proof. The user interacts with a proof editor and provides it with the necessary tactics to prove goals while some of the proof steps are solved automatically by the automatic proof procedures.

4 Proposed Approach

A cyber-physical transportation system is composed of several interacting components which effect the travel demands within a given area and the services to satisfy these demands. A vast majority of the underlying components of transportation are nondeterministic. For example, the number of passengers, traffic at a given time and the speed of the individual vehicles are all random quantities. In the higher-order logic theorem proving based analysis, we propose to formalize the behaviour of the given transportation system including its randomized and unpredictable components in higher-order logic. The randomized behaviours would be captured in these formal models by using appropriate random variables.

The second step in theorem proving based probabilistic analysis is to utilize the formal model of the cyber-physical transportation system to express desired system properties as higher-order logic goals. The prerequisite for this step is the ability to express probabilistic and statistical properties related to both discrete and continuous random variables in higher-order logic. All probabilistic properties of discrete and continuous random variables can be expressed in terms of their *Probability Mass Functions* (PMFs) and *Cumulative Distribution Function* (CDFs), respectively. Similarly, most of the commonly used statistical properties can be expressed in terms of the expectation and variance characteristics of the corresponding random variable.

We require the formalization of mathematical definitions of PMF, CDF, expectation and variance for both discrete and continuous random variables in order to be able to express the given system's reliability characteristics as higher-order logic theorems. The third and the final step for conducting the formal probabilistic analysis in a theorem prover is to formally verify the higher-order logic goals developed in the previous step using a theorem prover. For this verification, it would be quite handy to have access to a library of some pre-verified theorems corresponding to some commonly used properties regarding probability distribution functions, expectation and variance. Since, we can build upon such a library of theorems and thus speed up the verification process.

Next, the higher-order logic formalization details associated with the above mentioned prerequisites are briefly described.

4.1 Discrete Random Variables and the PMF

A random variable is called discrete if its range, i.e., the set of values that it can attain, is finite or at most countably infinite. Discrete random variables can be completely characterized by their PMFs that return the probability that a random variable X is equal to some value x, i.e., $Pr(X = x)$.

Discrete random variables can be formalized in higher-order logic as deterministic functions with access to an infinite Boolean sequence B^∞; an infinite source of random bits with data type ($natural \rightarrow bool$) [27]. These deterministic functions make random choices based on the result of popping bits in the infinite Boolean sequence and may pop as many random bits as they need for their computation. When the functions terminate, they return the result along with the remaining portion of the infinite Boolean sequence to be used by other functions. Thus, a random variable that takes a parameter of type α and ranges over values of type β can be represented by the function

$$\mathcal{F} : \alpha \rightarrow B^{\infty} \rightarrow (\beta \times B^{\infty})$$

For example, a $Bernoulli(\frac{1}{2})$ random variable that returns 1 or 0 with probability $\frac{1}{2}$ can be modelled as

$\vdash$ `bit = ` λ`s. (if shd s then 1 else 0, stl s)`

where the variable s represents the infinite Boolean sequence and the functions `shd` and `stl` are the sequence equivalents of the list operations *'head'* and *'tail'*. A function of the form λ`x.t` represents a lambda abstraction function that maps x to $t(x)$. The function `bit` accepts the infinite Boolean sequence and returns a pair with the first element equal to either 0 or 1 and the second element equal to the unused portion of the infinite Boolean sequence.

The higher-order logic formalization of probability theory [27] also consists of a probability function $\mathbb{P}$ from sets of infinite Boolean sequences to *real* numbers between 0 and 1. The domain of $\mathbb{P}$ is the set $\mathcal{E}$ of events of the probability. Both $\mathbb{P}$ and $\mathcal{E}$ are defined using the Carathéodory's Extension theorem, which ensures that $\mathcal{E}$ is a σ-algebra: closed under complements and countable unions. The formalized $\mathbb{P}$ and $\mathcal{E}$ can be used to formally verify all basic axioms of probability. Similarly, they can also be used to prove probabilistic properties for random variables. For example, we can formally verify the following probabilistic property for the function `bit`, defined above,

$\vdash \mathbb{P}$ {`s | fst (bit s) = 1`} `=` $\frac{1}{2}$

where the function `fst` selects the first component of a pair and $\{x|C(x)\}$ represents a set of all elements x that satisfy the condition C.

The above mentioned infrastructure can be utilized to formalize most of the commonly used discrete random variables and verify their corresponding PMF relations [27]. For example, the formalization and verification of Bernoulli and Uniform random variables can be found in [27] and of Binomial and Geometric random variables can be found in [24].

4.2 Continuous Random Variables and the CDF

A random variable is called continuous if it ranges over a continuous set of numbers that contains all real numbers between two limits. Continuous random variables can be completely characterized by their CDFs that return the probability that a random variable X is exactly less than or equal to some value x, i.e., $Pr(X \leq x)$.

The sampling algorithms for continuous random variables are non-terminating and hence require a different formalization approach than discrete random variables, for which the sampling algorithms are either guaranteed to terminate or satisfy probabilistic termination, meaning that the probability that the algorithm terminates is 1. One approach to address this issue is to utilize the concept of the non-uniform random number generation [11], which is the process of obtaining arbitrary continuous random numbers using a Standard Uniform random number generator. The main advantage of this approach is that we only need to formalize the Standard Uniform random variable from scratch and use it to model other continuous random variables by formalizing the corresponding non-uniform random number generation method.

Based on the above approach, a methodology for the formalization of all continuous random variables for which the inverse of the CDF can be represented in a closed mathematical form is presented in [24]. The first step in this methodology is the formalization of the Standard Uniform random variable, which can be done by using the formalization approach for discrete random variables and the formalization of the mathematical concept of limit of a *real* sequence [19]:

$$\lim_{n \to \infty} (\lambda n. \sum_{k=0}^{n-1} (\frac{1}{2})^{k+1} X_k) \tag{1}$$

where X_k denotes the outcome of the k^{th} random bit; $True$ or $False$ represented as 1 or 0, respectively. The formalization details are outlined in [24].

The second step in the methodology for the formalization of continuous probability distributions is the formalization of the CDF and the verification of its classical properties. This is followed by the formal specification of the mathematical concept of the inverse function of a CDF. This definition along with the formalization of the Standard Uniform random variable and the CDF properties, can be used to formally verify the correctness of the Inverse Transform Method (ITM) [11]. The ITM is a well known non-uniform random generation technique for generating non-uniform random variables for continuous probability distributions for which the inverse of the CDF can be represented in a closed mathematical form. Formally, it can be verified for a random variable X with CDF F using the Standard Uniform random variable U as follows [24].

$$Pr(F^{-1}(U) \leq x) = F(x) \tag{2}$$

The formalized Standard Uniform random variable can now be used to formally specify any continuous random variable for which the inverse of the CDF can be expressed in a closed mathematical form as $X = F^{-1}(U)$. Whereas, its CDF can be verified based on simple arithmetic reasoning, using the formally verified ITM, given in Equation (2). This approach has been successfully utilized to formalize and verify Exponential, Uniform, Rayleigh and Triangular random variables [24].

4.3 Statistical Properties for Discrete Random Variables

In probabilistic analysis, statistical characteristics play a major role in decision making as they tend to summarize the probability distribution characteristics of a random variable in a single number. Due to their widespread interest, the computation of statistical characteristics has now become one of the core components of every contemporary probabilistic analysis framework.

The expectation for a function of a discrete random variable, which attains values in the positive integers only, is defined as follows [30]

$$Ex\_fn[f(X)] = \sum_{n=0}^{\infty} f(n) Pr(X = n) \tag{3}$$

where X is the discrete random variable and f represents a function of X. The above definition only holds if the associated summation is convergent, i.e., $\sum_{n=0}^{\infty} f(n)Pr(X = n) < \infty$. The expression of expectation, given in Equation (3), has been formalized in [24] as a higher-order logic function using the probability function $\mathbb{P}$. The expected value of a discrete random variable that attains values in positive integers can now be defined as a special case of Equation (3)

$$Ex[X] = Ex\_fn[(\lambda n.n)(X)] \tag{4}$$

when f is an identity function. In order to verify the correctness of the above definitions of expectation, they are utilized in [24] to formally verify various properties of expectation like its linearity and Markov's inequality. These properties not only verify the correctness of the above definitions but also play a vital role in verifying the expectation characteristics of discrete random components of probabilistic systems.

Variance of a random variable X describes the difference between X and its expected value and thus is a measure of its dispersion.

$$Var[X] = Ex[(X - Ex[X])^2] \tag{5}$$

The above definition of variance has been formalized in higher-order logic in [24] by utilizing the formal definitions of expectation, given in Equations (3) and (4). This definition is then formally verified to be correct by proving its classical properties like linearity and Chebyshev's inequality.

These results allow us to reason about expectation, variance and tail distribution properties of any formalized discrete random variable that attains values in positive integers, e.g., the formal verification for Bernoulli, Uniform, Binomial and Geometric random variables is presented in [24].

4.4 Statistical Properties for Continuous Random Variables

The most commonly used definition of expectation, for a continuous random variable X, is the probability density-weighted integral over the real line.

$$E[X] = \int_{-\infty}^{+\infty} x f(x) dx \tag{6}$$

The function f in the above equation represents the *Probability Density Function* (PDF) of X and the integral is the well-known Reimann integral. The above definition is limited to continuous random variables that have a well-defined PDF. A more general, but not so commonly used, definition of expectation for a random variable X, defined on a probability space (Ω, Σ, P) [14], is as follows:

$$E[X] = \int_{\Omega} X dP \tag{7}$$

This definition utilizes the Lebesgue integral and is general enough to cater for both discrete and continuous random variables. The reason behind its limited usage in the probabilistic analysis domain is the complexity of solving the Lebesgue integral, which takes its foundations from the measure theory that most engineers and computer scientists are not familiar with.

The obvious advantage of using Equation (6) for formalizing expectation of a continuous random variable is the user familiarity with Reimann integral that usually facilitates the reasoning process regarding the expectation properties in the theorem proving based probabilistic analysis approach. On the other hand, it requires extended real numbers, $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$, whereas all the foundational work regarding theorem proving based probabilistic analysis, outlined above, has been built upon the standard real numbers $\mathbb{R}$, formalized by Harrison [19]. The expectation definition given in Equation (7) does not involve extended real numbers, as it accommodates infinite limits without any ad-hoc devices due to the inherent nature of the Lebesgue integral. It also offers a more general solution. The limitation, however, is the compromise on the interactive reasoning effort, as it is not a straightforward task for a user to build on this definition to formally verify the expectation of a random variable.

We have formalized the expectation of a continuous random variable as in Equation (7) by building on top of a higher-order logic formalization of Lebesgue integration theory [?]. Starting from this definition, two simplified expressions for the expectation are verified that allow us to reason about expectation of a continuous random variable in terms of simple arithmetic operations [24]. The first expression is for the case when the given continuous random variable X is bounded in the positive interval $[a, b]$

$$E[X] = \lim_{n\to\infty} \left[\sum_{i=0}^{2^n-1} (a + \frac{i}{2^n}(b-a)) P\left\{ a + \frac{i}{2^n}(b-a) \le X < a + \frac{i+1}{2^n}(b-a) \right\} \right] \tag{8}$$

and the second one is for an unbounded positive random variable [14].

$$E[X] = \lim_{n\to\infty} \left[\sum_{i=0}^{n2^n-1} \frac{i}{2^n} P\left\{ \frac{i}{2^n} \le X < \frac{i+1}{2^n} \right\} + nP(X \ge n) \right] \tag{9}$$

Both of the above expressions do not involve any concepts from Lebesgue integration theory and are based on the well-known arithmetic operations like summation, limit of a real sequence, etc. Thus, users can simply utilize them, instead of Equation (7), to reason about the expectation properties of their random variables and gain the benefits of the original Lebesgue based definition. The formal verification details for these expressions are given in [24]. These expressions are further utilized to verify the expected values of Uniform, Triangular and Exponential random variables [24]. The above mentioned definition and simplified expressions will also prove to very helpful in formalizing variance and verifying its corresponding properties in a theorem prover, which to the best of our knowledge has not been done so far.

The above mentioned formalization plays a pivotal role for the formal probabilistic analysis of cyber-physical transportation systems. It is a general framework that can be built upon to formally reason about any kind of a system and property. Besides that, there are numerous other advantages of our proposed approach compared to the proof-based languages like Event-B. A couple of worth mentioning features include: (1) absolute correctness of results due to the inherent strong typed nature and soundness of higher-order logic theorem proving, and (2) the ability to verify all sorts of properties, including the temporal ones, due to the expressiveness of the underlying higher-order logic.

In the next section, we illustrate the usefulness and practical effectiveness of the proposed approach by analysing a control algorithm of autonomous vehicles moving as a platoon using HOL.

5 Vehicle Platoon Control Algorithm

A platoon is a set of self-operating vehicles moving in a convoy. It can be seen as a road-train where cars are linked by software, instead of hardware. Platooning has several potential uses in an urban mobility system: augmenting throughput, herding unused cars to stations, or running transient buses, for instance.

Homologous to other transportation systems, vehicle platoons also exhibit many unpredictable characteristics, like the platoon speed, inter-platoon gaps, intra-platoon headways and the inter-arrival time between consecutive platoons. In this study, we concentrate only on one randomized aspect, i.e., headways. The gaps between the vehicles in different platoons significantly impact the performance of an un-signalized intersection. Our formal modelling is based on a dichotomized headway model [10]. This model distinguishes the free vehicles (platoon leaders that are travelling without interacting with the vehicles ahead) and the bunched vehicles (platoon leader followers) in terms of their headway distribution. It assumes that the free vehicles with a proportion, α, of all platoons follow the displaced exponential headway distribution while the bunched vehicles $(1-\alpha)$ have the same constant headway t_m.

We formalized the underlying continuous random variable of the dichotomized headway model described above in HOL as follows:

Definition 1: *Dichotomized Headway Random Variable*
$\vdash \forall$ `l a t_m s. headway_rv l a t_m s =`
    $(\lambda$`x.` $-\frac{1}{l}$ `(ln (1 - x))/a+ t_m) (std_unif_cont s)`

where the variables `l`, `a`, `t_m`, and `s`, denote the decay constant, variable α, variable t_m, and the infinite boolean sequence, formalized in [27], respectively. The function `std_unif_cont` represents the standard uniform random variable and has been used to facilitate the inverse transform method based formalization of the given random variable, as described in the previous section.

Based on Definition 1, we also formally verified the following useful Cumulative Distribution Function (CDF) property.

Theorem 1: *CDF for the Dichotomized Headway Random Variable*
```
⊢ ∀ l a t_m t. (0 < l) ∧ (0 < a) ⇒
    cdf (λs. headway_rv l a t_m s) t =
      if t ≤ t_m then 0 else (1 - a (exp (-l (t - t_m))))
```

where `cdf` represents the HOL function for the CDF. The verification of Theorem 1 was done interactively and the formal reasoning relied heavily upon the probabilistic analysis related formalization [27,24], transcendental functions and real analysis. It is important to note that the above mentioned theorem is universally quantified for all the parameters, which means that these parameters can be specialized to obtain any specific results. Moreover, the CDF allows us to reason about any probabilistic property of the system. For example, we used it to reason about the probability of the headway being greater than $x + t_m$ seconds as follows:

Theorem 2: *Probabilistic Property of Platoon Headway*
```
⊢ ∀ l a t_m x. (0 < l) ∧ (0 < a) ∧ (0 < x) ⇒
    ℙ {s | (headway_rv l a t_m s) > x + t_m} = a (exp (-l x))
```

The above exercise illustrates the fact that interactive theorem proving is capable of conducting probabilistic analysis of cyber-physical transportation systems with at least the same degree of accuracy as the analytical proof techniques usually carried out using paper-and-pencil proof methods; a novelty that cannot be achieved by any other computer based techniques, such as simulation or model checking. As aforementioned, simulation-based techniques rely on many approximations and thus can never achieve accurate results. Similarly, due to the inherent limitations of the state-based formal methods, discussed in Section 2, they cannot evaluate the values as precisely as we have attained using the proposed approach.

6 Conclusions

In this paper, we have advocated the unification of formal analysis with quantitative reasoning for the verification of cyber-physical transportation systems. The main idea is that in addition to analysing the absolute correctness, which at times is not possible, a probabilistic analysis can provide a better insight about the model. In this fashion, it is easier for stakeholders to obtain a probability of occurrence of a hazard in terms of the likelihood of components failures. The paper presents an introduction to the available framework for formal probabilistic analysis in higher-order logic and proposes to use it to analyse the probability aspects of considered systems. To demonstrate our approach, we have used a small-scale example of inter-platoon headway property of a platooning algorithm. To the best of our knowledge, this is the first time that higher-order logic theorem proving has been proposed to be used in this context in the open literature.

In future, we intend to expand the boundaries of our work to a full-scale platooning system [37] and the transport domain model [31]. Our aim is to bring their Event-B specifications into higher-order logic in HOL and integrate them with associated random variable models, like the one given in Definition 1 and reason about more advanced probabilistic properties.

References

1. Abrial, J.R.: The B Book. Cambridge University Press (1996)
2. Abrial, J.R.: Modeling in Event-B: System and Software Engineering. Cambridge University Press (2010)
3. Badeau, F., Amelot, A.: Using B as a High Level Programming Language in an Industrial Project: Roissy VAL. In: 4th International Conference of Z and B Users (ZB'05). Springer, Guildford, UK (2005)
4. Baier, C., Katoen, J.: Principles of Model Checking. MIT Press (2008)
5. Behm, P., Benoit, P., Meynadier, J.M.: METEOR: A Successful Application of B in a Large Project. In: International Symposium on Formal Methods (FM'99). Springer, Toulouse, France (1999)
6. Brown, C.: Automated Reasoning in Higher-order Logic. College Publications (2007)
7. Cardell-Oliver, R.: The Formal Verification of Hard Real-time Systems. PhD Thesis, University of Cambridge, UK (1992)
8. Church, A.: A Formulation of the Simple Theory of Types. Journal of Symbolic Logic 5, pp. 56–68 (1940)
9. Clarke, E.M., Zuliani, P.: Statistical Model Checking for Cyber-Physical Systems. In: 9th International Conference on Automated Technology for Verification and Analysis (ATVA'11). Springer, Taipei, Taiwan (2011)
10. Cowan, R.J.: Useful Headway Models. Transportation Research 9, 371–375 (1975)
11. Devroye, L.: Non-Uniform Random Variate Generation. Springer (1986)
12. Elleuch, M., Hasan, O., Tahar, S., Abid, M.: Formal Analysis of a Scheduling Algorithm for Wireless Sensor Networks. In: 13th International Conference on Formal Engineering Methods (ICFEM'11). Springer, Durham, UK (2011)
13. Fitting, M.: First-Order Logic and Automated Theorem Proving. Springer (1996)
14. Galambos, J.: Advanced Probability Theory. Marcel Dekker Inc. (1995)
15. Gomes, A., Mota, A., Sampaio, A., Ferri, F., Buzzi, J.: Systematic Model-based Safety Assessment via Probabilistic Model Checking. In: 4th International Symposium On Leveraging Applications of Formal Methods, Verification and Validation (ISoLA'10), Springer, Crete, Greece (2010)
16. Gordon, M.: Mechanizing Programming Logics in Higher-0rder Logic. In: Current Trends in Hardware Verification and Automated Theorem Proving. pp. 387–439. Springer (1989)
17. Hallerstede, S., Hoang, T.S.: Qualitative Probabilistic Modeling in Event-B. In: 6th International Conference on Integrated Formal Methods (iFM'07). Springer, Oxford, UK (2007)
18. Harrison, J.: Formalized Mathematics. Technical Report 36, Turku Centre for Computer Science, Finland (1996)
19. Harrison, J.: Theorem Proving with the Real Numbers. Springer (1998)

20. Harrison, J.: Handbook of Practical Logic and Automated Reasoning. Cambridge University Press (2009)
21. Harrison, J., Slind, K., Arthan, R.: HOL. In: The Seventeen Provers of the World. pp. 11–19, Springer (2006)
22. Hasan, O., Afshar, S.K., Tahar, S.: Formal Analysis of Optical Waveguides in HOL. In: 22nd International Conference on Theorem Proving in Higher-Order Logics. Springer, Munich, Germany (2009)
23. Hasan, O., Tahar, S., Abbasi, N.: Formal Reliability Analysis using Theorem Proving. IEEE Transactions on Computers 59(5), 579–592 (2010)
24. Hasan, O., Tahar, S.: Formal Probabilistic Analysis: A Higher-Order Logic Based Approach. In: 2nd International Conference on Abstract State Machines (ASM), Alloy, B, and Z (ABZ'10). Springer, Orford, Canada (2010)
25. Herencia-Zapana, H., Hagen, G., Narkawicz, A.: Formalizing Probabilistic Safety Claims. In: 3rd NASA Formal Methods Symposium (NFM'11). Springer, Pasadena, CA, USA (2011)
26. Holzl, .J., Heller, A.: Three Chapters of Measure Theory in Isabelle/HOL. In: 2nd International Conference on Interactive Theorem Proving. Springer, Nijmegen, The Netherlands (2011)
27. Hurd, J.: Formal Verification of Probabilistic Algorithms. PhD Thesis, University of Cambridge, UK (2002)
28. Kwiatkowska, M., Norman, G., Parker, D.: Controller Dependability Analysis by Probabilistic Model Checking. Control Engineering Practice 15(11), pp. 1427–1434 (2007)
29. Kwiatkowska, M., Norman, G., Parker, D.: Prism: Probabilistic Symbolic Model Checker. Computer Performance Evaluation Modeling Techniques and Tools 2324(4), pp. 200–204 (2002)
30. Levine, A.: Theory of Probability. Addison-Wesley (1971)
31. Mashkoor, A., Jacquot, J.P.: Utilizing Event-B for Domain Engineering: A Critical Analysis. Requirements Engineering 16(3), pp. 191–207 (2011)
32. Mhamdi, T., Hasan, O., Tahar, S.: On the Formalization of the Lebesgue Integration Theory in HOL. In: 2nd International Conference on Interactive Theorem Proving. Springer, Nijmegen, The Netherlands (2011)
33. Milner, R.: A Theory of Type Polymorphism in Programming. Journal of Computer and System Sciences 17, pp. 348–375 (1977)
34. Paulson, L.: ML for the Working Programmer. Cambridge University Press (1996)
35. Siddique, U., Hasan, O.: Formal Analysis of Fractional Order Systems in HOL. In: Formal Methods in Computer Aided Design. pp. 163–170 (2011)
36. Tarasyuk, A., Troubitsyna, E., Laibinis, L.: Towards Probabilistic Modelling in Event-B. In: 8th International Conference on Integrated Formal Methods (iFM'10). Springer, Nancy, France (2010)
37. Yang, F., Jacquot, J.P.: Scaling up with Event-B: A Case Study. In: 3rd NASA Formal Methods Symposium (NFM'11). Springer, Pasadena, CA, USA (2011)