Formalization of Birth-Death and IID Processes in Higher-order Logic

Liya Liu, Osman Hasan and Sofiène Tahar

Department of Electrical and Computer Engineering,
Concordia University, Montreal, Canada
Email: {liy_liu, o_hasan, tahar}@ece.concordia.ca

Technical Report

November, 2016

Abstract

Markov chains are extensively used in modeling and analysis of engineering and scientific problems. Usually, paper-and-pencil proofs, simulation or computer algebra software are used to analyze Markovian models. However, these techniques either are not scalable or do not guarantee accurate results, which are vital in safety-critical systems. Probabilistic model checking has been recently proposed to formally analyze Markovian systems, but it suffers from the inherent state-explosion problem and unacceptable long computation times. To overcome these limitations, in this paper, we develop a framework to formally analyze discrete-time Markov models with finite state-space using higher-order logic theorem proving. The core component of the proposed framework is a higher-order-logic formalization of Discrete-Time Markov Chains (DTMC) and classified DTMCs. The proposed framework is generic enough to be used for extending the library of formal reasoning support for Markovian models and analyzing many real-world stochastic problems. In order to demonstrate the usefulness of our framework to formalize other Markov chain related concepts, we present the formalization of a Discrete-time Birth-Death process and the discrete Independent and Identically Distributed (IID) random process. Moreover, for illustrating the practical utilization of our framework, we use it to formally analyze the performance of a program, which controls a CPU and its connected devices through the system bus, as well as the performance of a data structure in a program.

1 Introduction

In probability theory, Markov chains are used to model time varying random phenomena that exhibit the memoryless property [3]. In fact, most of the randomness that we encounter in engineering and scientific domains has some sort of time-dependency. For example, noise signals vary with time, the duration of a telephone call is somehow related to the time it is made, population growth is time dependent and so is the case with chemical reactions. Therefore, Markov chains have been extensively investigated and applied for designing systems in many branches of science and engineering, including hardware circuits [31], software testing [49], Internet page ranking [12] and statistical mechanics [4].

Mathematically, a DTMC can be divided into two main categories. It may be *time homogeneous*, which refers to the case where the underlying Markov chains exhibit the constant transition probabilities between the states, or *time inhomogeneous*, where the transition probabilities between the states are not constant and are time dependent. Furthermore, DTMCs are also classified in terms of the characteristics of their state-space. For example, some states can be reached from all other states and some others once entered, cannot be left. In practice, these reachable states are the most attractive states in the dynamic analysis of Markovian systems. Regarding the features of the states in their state-space, DTMCs are categorized into different classes, such as *irreducible DTMC*, *aperiodic DTMC*, *absorbing DTMC*, etc. These classified Markov chains [46] are widely used to simplify the analysis of long-term behaviors for most applications. Among them, Discrete-time Birth-Death process is an important aperiodic and irreducible DTMC, which is mainly used in modeling and analyzing software performance.

Traditionally, engineers have been using paper-and-pencil proof methods to perform probabilistic and statistical analysis of Markov chain systems. Nowadays, real-world systems have become considerably complex and the behaviors of some critical subsystems need to be analyzed accurately. However, due to the increasing complexity, it becomes practically impossible to analyze a complex system precisely by paper-and-pencil methods due to the risk of human errors. Therefore a variety of computer-based techniques, such as simulation, computer algebra systems and probabilistic model checking have been used to analyze Markovian models.

The simulation based analysis is irreverently inaccurate due to the usage of pseudo random number generators and the sampling based nature of the approach. To improve the accuracy of the simulation results, Markov Chain Monte Carlo (MCMC) methods [39], which involve sampling from desired probability distributions by constructing a Markov chain with the desired distribution, are frequently applied. The major limitation of MCMC is that it generally requires hundreds of thousands of simulations to evaluate the desired probabilistic quantities and becomes impractical when each simulation step involves extensive computations.

Computer Algebra Systems (CAS) provide automated support for analyzing Markovian models and symbolic representations of Markovian systems using software tools, such as

Mathematica [42] and *Maple* [40]. However, the usage of huge symbolic manipulation algorithms, which have not been verified, in the cores of CASs also makes the analysis results untrustworthy. In addition, the computations in CAS cannot be completely trusted as well since they are based on numerical methods.

Due to the extensive usage of Markov chains for safety-critical systems, and thus the requirement of accurate analysis in these domains, probabilistic model checking has been recently proposed for formally analyzing Markovian systems. However, some algorithms implemented in these model checking tools are based on numerical methods too. For example, the Power method [47], which is a well-known iterative method, is applied to compute the steady-state probabilities (or limiting probabilities) of Markov chains in PRISM [50]. Moreover, model checking cannot be used to verify generic mathematical expressions with universally quantified continuous variables for the properties of interest (i.e., variable values have to be bounded and discretized to avoid endless computation time). Finally, model checkers suffer from the state-exploration problems when the analyzed systems are complex.

Higher-order-logic interactive theorem proving [15] is a formal method that provides a conceptually simple formalism with precise semantics. It allows to construct a computer based mathematical model of the system and use mathematical reasoning to formally verify systems properties of interest. The formal nature of analysis allows us to solve the inaccuracy problem mentioned above. Due to the highly expressive nature of higher-order logic and the inherent soundness of theorem proving, this technique is capable of conducting the formal analysis of various Markov chain models including hidden Markovian models [4], which, to our best knowledge, probabilistic model checking cannot cater for. Moreover, interactive theorem proving using higher-order logic is capable of verifying generic mathematical expressions and it does not suffer from the state-exploration problem of model checking.

Leveraging upon the above-mentioned strengths of higher-order-logic theorem proving and building upon a formalization of probability theory in HOL [44], we have formalized the definitions of a DTMC [36][38] and classified DTMCs [34] in higher-order logic. These definitions have then been used to formally verify classical properties of DTMCs, such as the joint probability theorem, Chapman-Kolmogorov Equation and Absolute probability [38], as well as Classified DTMCs, such as the stationary properties [34]. The above-mentioned formalization has been successfully used to verify some real-world applications, such as a binary communication channel [36], an Automatic Mail Quality Measurement protocol [38], a generic LRU (least recently used) stack model [34] and a memory contention problem in a Multiprocessor System [37], and to also formalize Hidden Markov Models (HMMs) [35].

These previous works clearly indicate the great potential and usefulness of higher-order-logic theorem proving based formal analysis of Markovian models. In the current paper, we utilize the core formalizations of DTMCs [38] and classified DTMCs [34] to propose a generic framework for the formal analysis of Markovian models using higher-order-logic theorem proving. The proposed framework, depicted in Figure 1, not only allows the formal modeling of systems that exhibit the discrete time Markovian behaviors but also allows its users to formalize more advanced discrete time random processes that are based on DTMCs

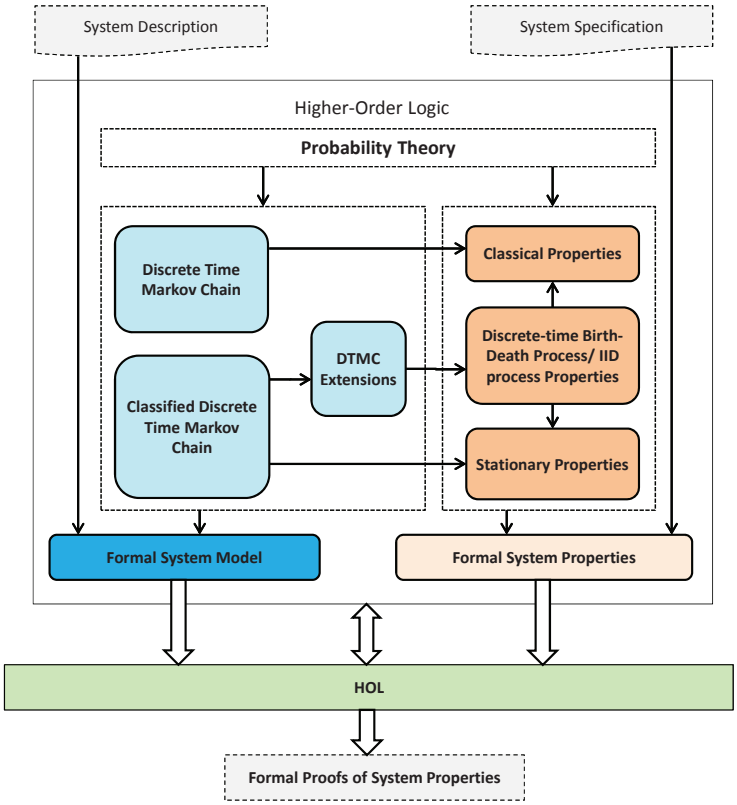

Figure 1: DTMC Formalization Framework

and classified DTMCs. These advanced random processes, such as the discrete-time birth-death and Independent and Identically Distributed (IID) processes, can in turn be used in our framework to analyze the systems that exhibit the corresponding behaviors.

The first step in analyzing a system that exhibits the discrete-time Markovian behavior in the proposed framework is to construct a formal system model of the given system as a function in higher-order logic. This can be done using the blue-colored boxes that contain the formal mathematical definitions of Markov chain foundations including discrete-time Markov chain, classified states and the classified discrete-time Markov chain. The second step is to formally express the system properties, which are given as a set of characteristics (system behaviors), as higher-order logic goals utilizing the formal system model developed in the first step. For the formal verification of system properties (proving these goals), the pre-verified general Markov chain properties (shown in the boxes colored as light brown), including the joint probability theorem, Chapman-Kolmogorov Equation, Absolute probability and the stationary properties, play a vital role and tend to minimize the user guidance efforts in the interactive proofs. Finally, the output of the theorem prover in this framework is the formal proofs of system properties, which is represented by the rectangular box with

dashed edges. The output certifies that the given system specifications are valid for the given Markovian system. In a similar way, the behavior of a discrete Markov chain can also be expressed in terms of the DTMC and classified DTMC and then their properties can be verified by building upon the existing properties of DTMCs, classified DTMCs or any other existing DTMC extensions. Once a DTMC extension is verified, it becomes part of the core formalization and can be used to formally analyze real-world systems.

In order to illustrate the usefulness of the proposed methodology, we first verify two discrete-time random processes with discrete state-space, i.e., the discrete-time birth-death process, which is the fundamental of a queue model applied in various telecommunication systems and the classical result that a random walk [14] is an Independent and Identically Distributed (IID) random process. Besides extending the capability of formal analysis of Markovian models, these formalizations also demonstrate the integrity and completeness of our formal definitions. Moreover, we utilize the proposed framework to analyze two software engineering applications, i.e., a simple program and a data-structure.

The rest of this paper is organized as follows. Section 2 presents a brief overview about existing related work. In Section 3, we provide some preliminary information about the probability and Markov chain theories that are required to understand the formalization described in the rest of the paper. We describe the formalization of discrete-time Markov chain and classified DTMCs in Section 4. The formalizations of discrete-time Birth-Death Chain model and the discrete IID random process are described in Section 5. The illustrative applications are given in Section 6. Finally, Section 7 concludes the paper.

2 Related Work

Markov Analyzers, such as *MARCA* [41] and *DNAmaca* [28], which contain numerous matrix manipulation and numerical solution procedures, are powerful autonomous tools for analyzing large-scale Markovian models. Many reliability evaluation software tools, such as *Möbius* [45] and *SHARPE* [53], integrate simulation and numerical analyzers for modeling and analyzing the reliability, maintainability or safety of systems using Markov methods, which offer simplistic modeling approaches and are more flexible compared to traditional approaches, e.g., Fault Tree [26]. Some prevalent software tools for evaluating performance, e.g., *MACOM* [51] and *HYDRA* [11], take the advantages of a popular Markovian algebra, i.e., *PEPA* [48], to model systems and efficiently compute passage time densities and quantities in large-scale Markov chains. Although these software packages can be successfully applied to analyze large scale Markovian models, the results cannot be guaranteed to be accurate because the underlying iterative methods [54] are not 100% precise.

Another technique, *Stochastic Petri Nets* (*SPN*) [17], has been found as a powerful method for modeling and analyzing Markovian systems because it allows local state modeling instead of global modeling. SPNs are utilized to model the stochastic systems and offer the capability of analyzing large and complex models. The Markov chain of a SPN is modeled

by means of a reachability graph [30]. The prevailing software tools of stochastic petri nets are *SPNP* [6] and *GreatSPN* [16]. These tools can model, validate, and evaluate distributed systems and analyze the dynamic events of the models by means of embedded Markov chain theory. For example, the quantitative analysis of Generalized Stochastic Petri Nets (GSPNs) [18] mainly depends on a Markovian solution, in which the models are described as semi-Markov processes in order to calculate the steady state distributions of stochastic systems. The calculations are based on numerical methods, which is the main limiting factor of the application of SPN for analyzing safety-critical system models. Another key limiting factor of the application of SPN models using this approach is the complexity of their analysis.

Probabilistic model checking tools, such as *PRISM* [50], *VESTA* [52] and *Ymer* [59], provide precise system analysis by modeling the stochastic behaviors, including its random components, using probabilistic state machines and exhaustively verifying their probabilistic properties. However, these tools use numerical methods for calculating probabilities and suffer from the inherent state-space explosion problems. Moreover, model checking tools cannot be used to verify generic expressions involving universally quantified continuous variables, which are frequently encountered in Markovian systems.

Higher-order-logic theorem proving is capable of overcoming the above-mentioned inaccuracy limitations. The foremost requirement of using theorem proving for analyzing Markovian models is the higher-order-logic formalization of the probability theory. Hurd [25] formalized a measure space as a pair (Σ, μ) in HOL. The sample space, on which this pair is defined, is implied from the higher-order-logic definitions to be equal to the universal set of the appropriate data-type. Building upon the formalization of measure space, the probability space was also defined in HOL as a pair $(\mathcal{E}, \mathbb{P})$, where the domain of $\mathbb{P}$ is the set $\mathcal{E}$, which is a set of subsets of infinite Boolean sequences $\mathbb{B}^\infty$. 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. As a consequence, the space is implicitly a universal set. This fact limits its scope considerably.

Later, Coble [7] formalized the measure space as the triple (X, Σ, μ), which allows to define an arbitrary space X and overcomes the disadvantage of Hurd's work. Coble's probability theory is built upon finitely-valued (standard real numbers) measures and functions. Specifically, the Borel sigma algebra cannot be defined on open sets and this constrains the verification of some applications.

More recently, Mhamdi [43] improved the development based on the axiomatic definition of probability proposed by Kolmogorov [29]. Mhamdi's theory provides a mathematical consistent for assigning and deducing probabilities of events. Hölzl [23] has also formalized three chapters of measure theory in Isabelle/HOL. However, this progressing work lacks Radon Nikodym derivative which is mainly used in analyzing expectation properties and it does not support signed measures or functions taking negative values. Affeldt [1] formalized some probability theory in Coq [9], with the main motivation of mechanizing the proof of Shannon's source coding and channel theorems.

The first formalization of time-homogeneous DTMC with finite state-space [36] was based on Hurd's formalization of probability theory [25]. However, the definition of DTMC is not general enough due to the above-mentioned limitations of the probability theory formalization [25] and the verified theorems are not rich enough to deal with various DTMC models. Our work, in this paper, mainly utilizes the most recent and general formalization of probability theory [44] and all the formalizations of discrete-time Markov chain are done using the theorem prover HOL4 [15].

Recently, the Isabelle/HOL theorem prover has also been used for the formalization of a time-homogeneous Markov chain [24] based on the corresponding probability theory formalization [23]. The aim of this work was to verify Probabilistic Computation Tree Logic (PCTL) in probabilistic model checkers, hence, to the best of our knowledge, a generalized formalization of DTMC theory has not been provided. Furthermore, this work has not been used for formalizing time-inhomogeneous Markov chains, which we tackle in the current paper.

3 Probability Theory in HOL4

In this section, we present an overview of the probability theory and the higher-order-logic formalizations of probability theory in HOL. Our formalization of DTMC is based on this work so it is a prerequisite to understand some of its technical details.

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 $(\Omega, \Sigma, \mathcal{P}r)$ such that the measure, referred to as the probability and denoted by $\mathcal{P}r$, of the sample space is 1.

In probability and statistical theory, an essential concept is *random variable*, which is a function from a probability to a *measurable space*. A measurable space refers to a pair (S, Σ), where S denotes a set and Σ represents a nonempty collection of subsets of S. Especially, if the set S is a *discrete set*, which contains only isolated elements, then this random variable is called a *discrete random variable*. The probability that a discrete random variable X is exactly equal to some value i is defined as the *probability mass function* (PMF) and it is mathematically expressed as $\mathcal{P}r(X = i)$.

Random process, which denotes a collection of random variables X_t $(t \in T)$, is another widely used concept in probability theory. If the indices (t) of the random variables X_t are real numbers, then this random process is a *continuous-time random process*. If the indices (t) of the random variables X_t are natural numbers, then it is a *discrete-time random process*.

One of the crucial concepts in the random process study is the *conditional probability*, which basically reflects the dependency between the events which happen at different times in a process. The formal definition of conditional probability in HOL can be found in [22], which is based on Hurd's work [25]. In order to make use of the most advanced probability

theory in our work, we improved the formalization of conditional probability as:

Definition 3.1 *(Conditional Probability)*

The conditional probability of the event A given the occurrence of the event B, when the probability of the occurrence of the event B is greater than 0, is

$$\mathcal{P}r(A|B) = \mathcal{P}r(A \cap B)/\mathcal{P}r(B)$$

```
⊢ ∀ A B. cond_prob p A B = prob p (A ∩ B) / prob p B
```

where `cond_prob` represents the conditional probability, and `prob` denotes the probability. They are different functions of probability space p in HOL. In this paper, we utilize the symbol $\mathbb{P}$ to denote both the HOL function `cond_prob p` and the function `prob p` in HOL code and the argument of $\mathbb{P}$ would clarify if we want to use it in the context of `cond_prob` or `prob`.

In order to facilitate the formalization of DTMC, we verified various classical properties of conditional probability based on Definition 3.1. Some of the prominent ones are listed below:

$$\mathcal{P}r(A \cap B) = \mathcal{P}r(A|B)\mathcal{P}r(B) \tag{1a}$$

$$\mathcal{P}r(A) = \sum_{i \in \Omega} \mathcal{P}r(B_i)\mathcal{P}(A|B_i) \tag{1b}$$

$$\mathcal{P}r(A) = \sum_{i \in \Omega} \mathcal{P}r(A)\mathcal{P}r(B_i|A) \tag{1c}$$

$$\sum_{i \in \Omega} \mathcal{P}r(B_i|A) = 1 \tag{1d}$$

where A, B and C are events in an event space, and the finite events set $\{B_i\}_{i \in \Omega}$ contains mutually exclusive and exhaustive events. The first theorem is based on Definition 3.1. The second one is the Total Probability Theorem (1b) and the third one is a lemma of the Total Probability Theorem. The last theorem is the Additivity Theorem.

A random variable is formally defined (formalized) as a measurable function `X` between a probability space `p` and a measurable space `s`. It is written as `random_variable X p s` in HOL. The definition of random variables is general enough to formalize both discrete and continuous random variables. Now, utilizing the formalization of random variables, the random process $\{X_t\}_{t \geq 0}$ can be easily written in HOL as `∀t.random_variable (X t) p s`.

4 Formalization of DTMCs and Classified DTMCs

In this section, we describe the formalization of discrete-time Markov chain and the formal verification of some of its most important properties.

4.1 Formalization of DTMCs

Given a probability space, a stochastic process $\{X_t : \Omega \to S\}$ is a sequence of random variables X, where t represents the time that can be discrete (represented by non-negative integers) or continuous (represented by real numbers) [3]. The set of values taken by each X_t, commonly called states, is referred to as the *state-space*. The *sample space* Ω of the process consists of all the possible state sequences based on a given state-space S. Now, based on these definitions, a *Markov process* can be defined as a stochastic process with *Markov property* [5]. If a Markov process has finite or countably infinite state-space Ω, then it is called a *Markov chain* and satisfies the following Markov property: For $0 \leq t_0 \leq \cdots \leq t_n$ and $f_0, \cdots, f_{n+1}$ in the state-space, then:

$$\mathcal{P}r\{X_{t_{n+1}} = f_{n+1} | X_{t_n} = f_n, \ldots, X_{t_0} = f_0\} = \mathcal{P}r\{X_{t_{n+1}} = f_{n+1} | X_{t_n} = f_n\} \quad (2)$$

where f_{n+1} represents the future state, f_n denotes the current state and f_0 refers to the initial state. $t_0 \cdots t_n$ indicates a consecutive time sequence, which can have values like $t_0 = 0$, $t_1 = 1, \cdots, t_n = n$ or $t_0 = 1$, $t_1 = 3, \cdots, t_n = n + 2$. This equation describes the important property of a discrete-time Markov chain (DTMC): the future state only depends on the current state and is independent of all the other past states.

In a DTMC, the space of its states $f_0, \cdots, f_{n+1}$ can be finite or infinite. In this paper, we mainly focus on the DTMC with a finite state-space, which is the fundamental of Markov chain theory. A *DTMC with finite state- space* is usually expressed by specifying: an initial distribution p_0 which gives the probability of initial occurrence $\mathcal{P}r(X_0 = s) = p_0(s)$ for every state; and transition probabilities $p_{ij}(t)$ which give the probability of going from i to j for every pair of states i, j in the state-space [46], at time t.

For states i, j and a time t, the *transition probability* $p_{ij}(t)$ is defined as $\mathcal{P}r\{X_{t+1} = j | X_t = i\}$, which can be easily generalized to *n-step transition probability*.

$$p_{ij}^{(n)}(t) = \begin{cases} \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases} & n = 0 \\ \mathcal{P}r\{X_{t+n} = j | X_t = i\} & n > 0 \end{cases} \quad (3)$$

In analyzing the stationary behaviors of Markovian models, it is quite common to categorize Markov chains into different classes depending on the properties exhibited by their states [3]. Some commonly used classes include *reducible*, *irreducible*, *periodic*, *aperiodic*, *regular* and *absorbing Markov chains*. Classified Markov chains are very useful for the dynamic analysis of systems as their properties allow us to judge long-term characteristics of Markovain systems, such as if a system will re-visit a particular state or to determine the time of the first visit to a state. Some of the widely used application areas of classified Markov chains are reliability analysis, performance analysis and validation of models.

Now the Markov property, given in Equation (2), can be formalized as follows:

Definition 4.1 *(Markov Property)*

```
⊢ ∀ X p s.  mc_property X p s =
  (∀ t.  random_variable (X t) p s) ∧
  ∀ f t n.  increasing_seq t ∧
        ℙ(⋂_{k∈ [0,n−1]}{x | X t_k x = f k}) ≠ 0 ⇒
        (ℙ({x|X t_{n+1} x = f (n + 1)}|{x|X t_n x = f n} ∩
             ⋂_{k∈ [0,n−1]}{x|X t_k x = f k}) =
        ℙ({x|X t_{n+1} x = f (n + 1)}|{x|X t_n x = f n}))
```

where `increasing_seq t` is defined as `∀ i j.  i < j ⇒ t i < t j`, thus formalizing the notion of increasing sequence, which denotes the time indices. The first conjunct indicates that the Markov property is based on a random process $\{X_t : \Omega \to S\}$. The quantified variable `X` represents a function of the random variables associated with time `t` which has the type `num`. This ensures that the process is a *discrete time* random process. The random variables in this process are the functions built on the probability space `p` and a measurable space `s`.

We also have to explicitly mention all the usually implicit assumptions stating that the states belong to the considered space. The assumption $\mathbb{P}(\bigcap_{k\in \mathtt{ts}}\{x \mid X\ k\ x = f\ k\}) \neq 0$ ensures that the corresponding conditional probabilities are well-defined, where `f k` returns the k^{th} element of the state sequence. In fact, the assumption $\mathbb{P}(\bigcap_{k\in [0,n-1]}\{x \mid X\ t_k\ x = f\ k\}) \neq 0$ ensures that the corresponding conditional probabilities are well-defined, i.e.,

$$\mathtt{prob\ p}\ (\bigcap_{k\in [0,n-1]}\{x \mid X\ t_k\ x = f\ k\}) \neq 0$$

The term `x ∈ p_space p` ensures that `x` is in the samples space in the considered probability space `p_space p`.

Another important concept in DTMC is the transition probability corresponding to the expression in Equation (3). The transition probability is formalized as a predicate in HOL as follows:

Definition 4.2 *(Transition Probability)*

```
⊢ ∀ X p s t n i j.  Trans X p s t n i j =
  if i ∈ space s ∧ j ∈ space s then
    if n = 0 then
      if (i = j) then 1
      else 0
    else
      ℙ({x | X (t + n) x = j}|{x | X t x = i})
  else 0
```

It is easy to understand that the probability of an event is zero, when this event is not in the event space. For instance, i is not in the state-space implies that event $\{X_t = i\} = \emptyset$. In this case, the conditional probability related to an empty set is zero.

Now, the discrete-time Markov chain (DTMC) can be formalized as follows:

Definition 4.3 *(Discrete-Time Markov Chain)*

```
⊢ ∀ X p s p₀ pᵢⱼ.  dtmc X p s p₀ =
  mc_property X p s ∧ (∀i.  i ∈ space s ⇒{i} ∈ subsets s) ∧
  (∀i.  i ∈ space s ⇒ (p₀ i = ℙ{x | X 0 x = i})) ∧
  (∀t i j.  ℙ{x | X t x = i} ≠ 0 ⇒ (pᵢⱼ t i j = Trans X p s t 1 i j))
```

where the first three variables are inherited from Definition 4.1, p_0 and p_{ij} refer to the functions expressing the given initial status and transition matrix associated with this random process, respectively. The first condition in this definition describes the Markov property presented in Definition 4.1 and the second one ensures the events associated with the state-space (`space s`) are discrete in the event space (`subsets s`), which is a *discrete space*. The last two conditions assign the functions $\mathtt{p_0}$ and $\mathtt{p_{ij}}$ to initial distributions and transition probabilities.

It is important to note that `X` is polymorphic, i.e., it is not assigned to a particular data type, which is a very useful feature of our definition.

In Definition 4.3, if the function p_{ij} depends on `t`, then this discrete-time Markov chain is referred to as the time-inhomogeneous Markov chain. However, most of the applications actually make use of *time-homogenous DTMCs*, i.e., DTMCs with finite state-space and time-independent transition probabilities [2]. The time-homogenous property refers to the time invariant feature of a random process. Thus, the one-step transition probability of the random process can be simplified as $p_{ij} = \mathcal{P}r\{X_{t+1} = j | X_t = i\} = p_{ij}(t)$, based on Equation (3). Now, the time-homogenous DTMC with finite state-space can be formalized as:

Definition 4.4 *(Time homogeneous DTMC)*

```
⊢ ∀ X p s p₀ pᵢⱼ.  th_dtmc X p s p₀ pᵢⱼ =
  dtmc X p s p₀ pᵢⱼ ∧ FINITE (space s) ∧
  ∀ t i j.  ℙ{x | X t x = i} ≠ 0 ∧ ℙ{x | X (t + 1) x = i} ≠ 0 ⇒
            (Trans X p s (t + 1) 1 i j = Trans X p s t 1 i j)
```

where the first and second conjuncts constraint this time-homogeneous DTMC to be a discrete-time Markov chain with the finite state-space, the last condition expresses the time-homogeneous property: $\forall\ t\ t'.\ p_{ij}(t) = p_{ij}(t')$ and thus $p_{ij}(t)$ is simply written as p_{ij} in the rest of this paper.

It is often the case that we are interested in the probability of some specific states as time tends to infinity under certain conditions. This is the main reason why stationary behaviors of stochastic processes are frequently analyzed in engineering and scientific domains. There is no exception for DTMCs.

Let $\{\mathrm{X}_t\}_{t\geq 0}$ be a Markov chain having state-space Ω and transition probabilities $\{p_{ij}\}_{i,j\in\Omega}$. If $\pi(i)$, $i \in \Omega$, are nonnegative numbers summing to one, then for all j, $j \in \Omega$, $\pi(j) = \sum_{i\in\Omega} \pi(i)p_{ij}$ is called a *stationary distribution*. The corresponding HOL definition is as follows.

Definition 4.5 *(Stationary Distribution)*

```
⊢ ∀f X p s.  stationary_dist f X p s =
  (SIGMA (λk.  f k) (space s) = 1) ∧
  ∀i. i ∈ space s ⇒
      0≤f i ∧ (∀t.  f i = SIGMA (λk.  f k * Trans X p s t 1 k i) (space s))
```

4.2 Verification of DTMC Properties

The *joint probability distribution* of a DTMC is the probability of a chain of states to occur. It is very useful in analyzing multi-stage experiments. In addition, this concept is the basis for the frequently used joint probability generating functions.

Theorem 4.1 (Joint Probability Distribution)

A joint probability distribution of n discrete random variables $X_0, \ldots X_n$ *in a finite DTMC* $\{X_t\}_{t\geq 0}$ *satisfies:*

$$\mathcal{P}r(X_t = L_0, \cdots, X_{t+n} = L_n) = \prod_{k=0}^{n-1} \mathcal{P}r(X_{t+k+1} = L_{k+1} | X_{t+k} = L_k) \mathcal{P}r(X_t = L_0)$$

```
⊢ ∀ X p s p L p₀ pᵢⱼ n.  dtmc X p s p₀ pᵢⱼ ⇒
   ℙ(⋂ⁿₖ₌₀{x | X (t + k) x = EL k L}) =
   PROD (0, n - 1) (λk.  ℙ({x | X (t + k + 1) x = EL (k + 1) L}|
                          {x|X (t + k) x = EL k L})
   ℙ{x | X t x = EL 0 L})
```

The proof of Theorem 4.1 is based on induction on the variable n, Definition 4.3 and some arithmetic reasoning.

The Chapman-Kolmogorov equation [3] is a widely used property of time homogeneous DTMCs. It basically gives the probability of going from state i to j in $m+n$ steps. Assuming the first m steps take the system from state i to some intermediate state k and the remaining n steps then take the system from state k to j, we can obtain the desired probability by adding the probabilities associated with all the intermediate steps.

Theorem 4.2 (Chapman-Kolmogorov Equation)

For a finite time homogeneous DTMC $\{X_t\}_{t\geq 0}$*, its transition probabilities satisfy the Chapman-Kolmogorov Equation*

$$p_{ij}^{(m+n)} = \sum_{k\in\Omega} p_{ik}^{(m)} p_{kj}^{(n)}$$

```
⊢ ∀ X p s i j t m n p₀ pᵢⱼ.  th_dtmc X p s p₀ pᵢⱼ ⇒
   Trans X p s t (m + n) i j =
   SIGMA (λk.  Trans X p s t m i k * Trans X p s t n k j) (space  s)
```

The proof of Theorem 4.2 again involves induction on the variables m and n and both of the base and step cases are discharged using the following lemma:

Lemma 4.1 (Multistep Transition Probability)

```
⊢ ∀ X p s i j t m p₀ pᵢⱼ.  th_dtmc X p s p₀ pᵢⱼ ⇒
   Trans X p s t (m + 1) i j =
   SIGMA (λk.  Trans X p s t 1 k j * Trans X p s t m i k) (space  s)
```

The proof of Lemma 4.1 is primarily based on Definitions 4.3 and 4.4 and the additivity property of probabilities.

The unconditional probabilities associated with a Markov chain are called *absolute probabilities*, which can be computed by applying the initial distributions and n-step transition probabilities. From now on, we write $p_j^{(n)}$ for the probability $\mathcal{P}r(X_n = j)$. We then have the following result:

Theorem 4.3 (Absolute Probability)
In a finite time homogeneous DTMC, the absolute probabilities $p_j^{(n)}$ *satisfy*

$$p_j^{(n)} = \mathcal{P}r(X_n = j) = \sum_{k \in \Omega} \mathcal{P}r(X_0 = k)\mathcal{P}r(X_n = j | X_0 = k)$$

```
⊢ ∀ X p s j n p₀ pᵢⱼ.  th_dtmc X p s p₀ pᵢⱼ ⇒
   ℙ{x | X n x = j} =
   SIGMA (λk.  ℙ{x|X 0 x = k}ℙ({x|X n x = j}|{x|X 0 s = k})) (space  s)
```

The proof of Theorem 4.3 is based on the Total Probability theorem (in Equation 1) along with some basic arithmetic and probability theoretic reasoning.

The main challenge of the work in this section is to describe the property of a DTMC using the predicates in higher-order logic.

4.3 Formalization of Classified DTMCs

In this section, the formalization of classified DTMCs will be introduced. We first formalize some foundational notions of classified states, which are categorized based on reachability, periodicity or absorbing features. Then, these results along with our formal definition of a DTMC are used to formalize classified Markov chains, such as aperiodic and irreducible DTMCs.

The foremost concept of states classification is the *first passage time* τ_j, or the *first hitting time*, which is defined as the minimum time required to reach a state j from the initial state i:

$$\tau_j = min\{t > 0 : X_t = j\}.$$

The first passage time can be defined in HOL as:

Definition 4.6 *(First Passage Time)*

$\vdash \forall$ `X x j.  FPT X x j = MINSET {t | 0 < t ∧ (X t x = j)}`

where `X` is a random process and `x` is a sample in the probability space associated with the random variable X_t. Note that the first passage time is also a random variable.

The conditional distribution of τ_j defined as the probability of the events starting from state i and visiting state j at time n is expressed as $f_{ij}^{(n)} = \mathcal{P}r\{\tau_j = n | X_0 = i\}$. This definition can be formalized in HOL as follows:

Definition 4.7 *(Probability of First Passage Events)*

$\vdash \forall$ `X p i j n.  f X p i j n = ℙ({x | FPT X x j = n}|{x | X 0 x = i})`

Another important notion, denoted as f_{ij}, is the probability of the events starting from state i and visiting state j at all times n, is expressed as $f_{ij} = \sum_{n=1}^{\infty} f_{ij}^{(n)}$. It can be expressed in HOL as `(λn.  f X p i j n) sums` f_{ij}. Thus f_{jj} provides the probability of events starting from state j and eventually returning back to j. If $f_{jj} = 1$, then the *mean return time* of state j is defined as $\mu_j = \sum_{n=1}^{\infty} n f_{jj}^{(n)}$. The existence of this infinite summation can be specified as `summmable (λn.  n * f X p j j n)` in HOL.

A state j in a DTMC $\{\mathrm{X}_t\}_{t \geq 0}$ is called a *transient* state if $f_{jj} < 1$, and a *persistent* state if $f_{jj} = 1$. If the mean return time μ_j of a persistent state j is finite, then j is said to be the *persistent nonnull state* (or *positive persistent state*). Similarly, if μ_j is infinite, then j is termed as the *persistent null state*.

The Greatest Common Divisor (*GCD*) of a set is a frequently used mathematical concept in defining classified states. We formalize the GCD of a set as follows:

Definition 4.8 *(The GCD of a Set)*

$\vdash \forall$ `A. GCD_SET A = MAXSET {r|∀x.  x ∈ A ⇒ divides r x}`

where `MAXSET` is a function in the Set Theory of HOL4 such that `MAXSET s` defines the maximum element in the set `s`. A *period* of a state j is any n such that $p_{jj}^{(n)}$ is greater than 0 and we write $d_j = GCD\ \{n : p_{jj}^{(n)} > 0\}$ as the GCD of the set of all periods. If the period of a state is 1, then this state is called *aperiodic state*.

A state i is said to be *accessible* from a state j (written $i \rightarrow j$), if there exists a nonzero n-step transition probability of the events from state i to j. Two states i and j are called *communicating states* (written $i \leftrightarrow j$) if they are mutually accessible. A state j is an *absorbing state* if the one-step transition probability $p_{jj} = 1$. The formalization of some other foundational notions of the classified states is given in Table 1.

We build upon the above mentioned definitions to formalize classified DTMCs. Usually, a DTMC is said to be *irreducible* if every state in its state-space can be reached from any other state including itself in finite steps.

Table 1: Formalization of Classified States

Definition	**Condition**	**HOL Formalization**
Transient State	$f_{jj} < 1$	$\vdash \forall$ X p j. Transient_state X p j = $\forall$ x. {t $\mid$ 0 $<$ t $\wedge$ (X t x = j)} $\neq \emptyset \wedge$ ($\exists$ s. s $<$ 1 $\wedge$ (λ n. f X p j j n) sums s)
Persistent State	$f_{jj} = 1$	$\vdash \forall$ X p j. Persistent_state X p j = $\forall$ x. {t $\mid$ 0 $<$ t $\wedge$ (X t x = j)} $\neq \emptyset \wedge$ (λn. f X p j j n) sums 1
Persistent Nonnull State	$f_{jj} = 1$ $\mu_j < \infty$	$\vdash \forall$ X p j. Nonnull_state X p j = Persistent_state X p j $\wedge$ summable (λn. n * f X p j j n)
Persistent Null State	$f_{jj} = 1$ $\mu_j = \infty$	$\vdash \forall$ X p j. Null_state X p j = Persistent_state X p j $\wedge$ $\sim$summable (λn. n*f X p j j n)
Periods of a State	0 $<$ n $0 < p^n_{jj}$	$\vdash \forall$ X p s j. Period_set X p s j = {n $\mid$ 0 $<$ n $\wedge \forall$t. 0 $<$ Trans X p s t n j j}
GCD of a Period Set	d_j	$\vdash \forall$ X p s j. Period X p s j = GCD_SET (Period_set X p s j)
Periodic State	$d_j > 1$	$\vdash \forall$ X p s j. Periodic_state X p s j = (1 $<$ Period X p s j) $\wedge$ (Period_set X p s j $\neq \emptyset$)
Aperiodic State	$d_j = 1$	$\vdash \forall$ X p s j. Aperiodic_state X p s j = (Period X p s j = 1) $\wedge$ Period_set X p s j $\neq \emptyset$)
Accessibility	i $\rightarrow$ j	$\vdash \forall$ X p s i j. Accessibility X p s i j = $\forall$ t. $\exists$ n. 0 $<$ n $\wedge$ 0 $<$ Trans X p s t n i j
Communicating State	i $\leftrightarrow$ j	$\vdash \forall$ X p s i j. Communicating_states X p s i j = (Accessibility X p s i j) $\wedge$ (Accessibility X p s j i)
Absorbing State	$p_{jj} = 1$	$\vdash \forall$ X p s j. Absorbing_states X p s j = (Trans X p s t 1 j j = 1)

Definition 4.9 *(Irreducible DTMC)*

```
⊢ ∀ X p s p0 pij.  Irreducible_mc X p s p0 pij =
  th_dtmc X p s p0 pij ∧
  (∀i j.  i ∈ space s ∧ j ∈ space s ⇒ Communicating_states X p s i j)
```

where `th_dtmc` represents the time-homogeneous Markov chain, defined in Definition 4.4, and the second conjunct expresses that all the states in the state-space can communicate with each other.

If there exists a state in the state-space of a DTMC, which cannot reach some other states, then this DTMC is called *reducible*.

Definition 4.10 *(Reducible DTMC)*

```
⊢ ∀ X p s p0 pij.  Reducible_mc X p s p0 pij =
  th_dtmc X p s p0 pij ∧
  ∃i j.  i ∈ space s ∧ j ∈ space s ∧ ∼Communicating_states X p s i j
```

A DTMC is considered as *aperiodic* if every state in its state-space is an aperiodic state; otherwise it is a *periodic DTMC*.

Definition 4.11 *(Aperiodic DTMC)*

```
⊢ ∀ X p s p0 pij.  Aperiodic_mc X p s p0 pij =
  th_dtmc X p s p0 pij ∧ (∀i.  i ∈ space s ⇒ Aperiodic_state X p s i)
```

Definition 4.12 *(Periodic DTMC)*

```
⊢ ∀ X p s p0 pij.  Periodic_mc X p s p0 pij =
  th_dtmc X p s p0 pij ∧ (∃ i.  i ∈ space s ∧ Periodic_state X p s i)
```

If at least one absorbing state exists in a DTMC and it is possible to go to the absorbing state from every non-absorbing state, then such a DTMC is named as an *absorbing DTMC*.

Definition 4.13 *(Absorbing DTMC)*

```
⊢ ∀ X p s p0 pij.
  Absorbing_mc X p s p0 pij =
  th_dtmc X p s p0 pij ∧
  ∃ i.  i ∈ space s ∧ Absorbing_state X p s i ∧
        (∀ j.  j ∈ space s ⇒ Communicating_state X p s i j)
```

Finally, if there exists some n such that $p_{ij}^{(n)} > 0$ for all states i and j in a DTMC, then this DTMC is defined as a *regular DTMC*.

Definition 4.14 *(Regular DTMC)*

```
⊢ ∀ X p s p₀ p_ij.
  Regular_mc X p s p₀ p_ij =
  th_dtmc X p s p₀ p_ij ∧ ∃n.  ∀i j.  i ∈ space s ∧ j ∈ space s ⇒
  Trans X p s t n i j > 0
```

The main utility of the higher-order logic formalization of the classified Markov chains mentioned above is to formally specify and analyze the dynamic features of Markovian systems within the sound environment of a theorem prover as will be demonstrated in Section 6 of this paper.

4.4 Verification of Classified DTMC Properties

Among the classified DTMCs formalized in the previous section, aperiodic and irreducible DTMCs are considered to be the most widely used ones in analyzing Markovian systems because of their attractive stationary properties, i.e., their limit probability distributions are independent of the initial distributions. For this reason, we now focus on the verification of some key properties of aperiodic and irreducible DTMCs [19].

Theorem 4.4 (Closed Period Set)
In an aperiodic DTMC, the set of the times when state i has a non-null probability of being visited is closed under addition.

```
⊢ ∀ X p s p₀ p_ij i.
  Aperiodic_DTMC X p s p₀ p_ij ∧ i ∈ space s ⇒
  ∀a b.  a ∈ Period_set X p s i ∧ b ∈ Period_set X p s i ⇒
         (a + b) ∈ Period_set X p s i
```

We verified the above theorem by using Theorem 4.2 and some arithmetic and set theoretic reasoning.

Another key property of an aperiodic DTMC states that the transition probability $p_{ij}^{(n)}$ is greater than zero, for all states i and j in its state-space, after n steps. It is very useful in analyzing the stability or reliability of many real-world systems.

Theorem 4.5 (Positive Return Probability)
For any state i in the finite state-space S of an aperiodic DTMC, there exists an $N < \infty$ such that $0 < p_{ii}^{(n)}$, for all $n \geq N$.

```
⊢ ∀ X p s p₀ p_ii i t.
  Aperiodic_DTMC X p s p₀ p_ii ∧ i ∈ space s ⇒
  ∃N. ∀n.  N ≤ n ⇒ 0 < Trans X p s t n i i
```

The formal reasoning about the correctness of the above theorems involves Theorems 4.2 and 4.4 and the following lemmas, along with some arithmetic reasoning and set theoretic reasoning.

Lemma 4.2 (Positive Element in a Closed Set)
If an integer set S contains at least one nonzero element and S is closed under addition and subtraction, then $S = \{kc; k \in \mathbb{Z}\}$, where c is the least positive element of S.

```
⊢ ∀ s:int → bool.
  s ≠ ∅ ∧ (∀ a b.  a ∈ s ∧ b ∈ s ⇒ (a + b) ∈ s ∧ (a - b) ∈ s) ⇒
  0 < MINSET {r|0<r ∧ r∈s} ∧ (s = {r|∃k.  r = k*MINSET {r | 0<r ∧ r∈s}})
```

Lemma 4.3 (Linearity of Two Integer Sequences)
For a positive integer sequence a_0, a_1, a_2, $\cdots$, a_k, there exists an integer sequence n_0, n_1, n_2, $\cdots$, n_k, such that $d = \sum_{i=0}^{k} n_i a_i$, where d is the GCD of sequence a_0, a_1, a_2, $\cdots$, a_k.

```
⊢ ∀ a k.
  0 < k ∧ (∀ i.  i ≤ k ⇒ 0 < a i) ⇒
  (∃n.  GCD_SET {a i | i ∈ [0, k]} = SIGMA (λn.  n i * a i) [0, k])
```

Lemma 4.4 (Least Number)
If a set of positive integers A is nonlattice, i.e., its GCD is 1, and closed under addition, then there exists an integer $N < \infty$ such that $n \in A$ for all $N \leq n$.

```
⊢ ∀ (A:int → bool) a.
  (A = {a i | 0 < a i ∧ i ∈ UNIV(:num)}) ∧ (GCD_SET A = 1) ∧
  (∀a b.  a∈A ∧ b∈A ⇒ (a + b) ∈ s) ⇒ (∃N. {n | N ≤ n} ⊂ A)
```

The proofs of Lemmas 4.2, 4.3 and 4.4 are based upon various summation properties of integer sets. These properties are not available in the HOL libraries and thus had to be verified as part of our development.

Theorem 4.6 (Existence of Positive Transition Probabilities)
For any aperiodic and irreducible DTMC with finite state-space S, there exists an N, for all $n \geq N$, such that the n-step transition probability $p_{ij}^{(n)}$ is non-zero, for all states i and $j \in S$.

```
⊢ ∀ X p s p₀ pᵢⱼ i j t.
  Aperiodic_DTMC X p s p₀ pᵢⱼ ∧ Irreducible_DTMC X p s p₀ pᵢⱼ ∧
  i ∈ space s ∧ j ∈ space s ⇒
  ∃N. ∀n.  N ≤ n ⇒ 0 < Trans X p s t n i j
```

We proceed with the proof of Theorem 4.6 by performing case analysis on the equality of i and j. The rest of the proof is primarily based on Theorems 4.2 and 4.5, Definition 4.1 and Lemmas 4.3 and 4.4.

Theorem 4.7 (Existence of Long-run Transition Probabilities)
For any aperiodic and irreducible DTMC with finite state-space S and transition probabilities p_{ij}, there exists $\lim_{n\to\infty} p_{ij}^{(n)}$, for all states i and $j \in S$.

```
⊢ ∀ X p s p₀ pᵢⱼ i j t.
  Aperiodic_DTMC X p s p₀ pᵢⱼ ∧ Irreducible_DTMC X p s p₀ pᵢⱼ ⇒
  ∃u.  (λn.  Trans X p s t n i j) → u
```

We first prove the monotonic properties of M_j^n and m_j^n, which are the maximum and minimum values of the set $\{n \leq 1: p_{ij}^{(n)}>0\}$, respectively. Then, the proof is completed by verifying the convergence of the sequence $(M_j^n$ - $m_j^n)$ for all n by applying Theorem 4.2 and some properties of real sequences. It is important to note that we do not need to use the assumption $j \in$ *space* s here, like all other theorems, as $\forall\ n\ j.\ j \notin$ *space* $s \Rightarrow (p_j^{(n)} = 0)$, which in turn implies $\lim_{n\to\infty} p_j^{(n)} = 0$ and $\lim_{n\to\infty} p_{ij}^{(n)} = 0$. The long-run probability distributions are often considered in the convergence analysis of random variables in stochastic systems. It is not very easy to verify that the limit probability distribution of a certain state exists in a generic non-trivial DTMC, because the computations required in such an analysis are often tremendous. However, in the aperiodic and irreducible DTMCs, we can prove that all states possess a limiting probability distribution, by the following two theorems.

Theorem 4.8 (Existence of Long-run Probability Distributions)
For any aperiodic and irreducible DTMC with finite state-space S, there exists $\lim_{n\to\infty} p_i^{(n)}$, for any state $i \in S$.

```
⊢ ∀ X p s p₀ pᵢⱼ i.
  Aperiodic_DTMC X p s p₀ pᵢⱼ ∧ Irreducible_DTMC X p s p₀ pᵢⱼ ⇒
  ∃ u.  (λ n.  ℙ{x | X n x = i}) → u
```

We used Theorems 4.3 and 4.7, along with some properties of the limit of a sequence, to prove this theorem in HOL.

Theorem 4.9 (Existence of Steady State Probability)
For every state i in an aperiodic and irreducible DTMC, $\lim_{n\to\infty} p_i^{(n)}$ is a stationary distribution.

```
⊢ ∀ X p s p₀ pᵢⱼ.
  Aperiodic_DTMC X p s p₀ pᵢⱼ ∧ Irreducible_DTMC X p s p₀ pᵢⱼ ⇒
  (stationary_dist (λi.  lim (λn.  ℙ{x | X n x = i})) X p s)
```

The proof of Theorem 4.9 involves rewriting with Definition 4.5 and then splitting it into the following three subgoals:

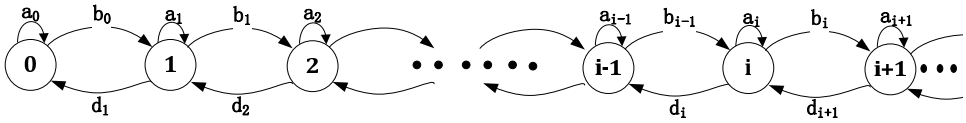

Figure 2: The State Diagram of Birth-Death Process

- $0 \leq \lim_{n\to\infty} p_j^{(n)}$
- $\sum_{i\in\Omega} \lim_{n\to\infty} p_i^{(n)} = 1$
- $\lim_{n\to\infty} p_j^{(n)} = \sum_{i\in\Omega} \lim_{n\to\infty} p_i^{(n)} p_{ij}$

Utilizing the probability bounds theorem, we can prove the first subgoal. The proof of the second subgoal is primarily based on the additivity property of conditional probability [22]. Then the last subgoal can be proved by applying the linearity of limit of a sequence and the linearity of real summation.

The major challenge of the work presented in this section, is to find a way to formally verify the theorems in HOL4. Many detailed proof steps are not available in textbooks. The proof scripts of all the theorems, presented in this section, contain about 8000 lines of HOL4 code and are available at [33]. These formally verified theorems facilitate the formal reasoning about DTMC system properties that can be modeled using classified Markov chains. For illustration purposes, we present a formalization of the Discrete-time Birth-Death process and the Independent and Identical Distribution Process in the next section.

5 DTMC Extensions

In this section, we show how we can apply the DTMC and classified DTMC properties to verify the discrete-time Birth-Death Process stationary features (such as limit probabilities and stationary distributions) and how to use the formal definition of DTMC to validate that a discrete-time IID random process with discrete state-space is a DTMC.

5.1 Discrete-time Birth-Death Process

Discrete-time Birth-Death process [56] is an important sub-class of Markov chains as it involves a state-space with nonnegative integers. Its remarkable feature is that all one-step transitions lead only to the nearest neighbor state. The discrete-time Birth-Death Processes are mainly used in analyzing software stability, for example, verifying if a data structure will have overflow problems.

The discrete-time Birth-Death Process, in which the states refer to the population, can be described as a state diagram depicted in Figure 2.

In this diagram, the states 0, 1, $\cdots$, i, $\cdots$ are associated with the population. The transition probabilities b_i represent the probability of a birth when the population is i, d_i denotes the probability of a death when the population becomes i, and a_i refers to the probability of the population in the state i. Considering $0 \leq a_i \leq 1$, $0 < b_i < 1$ and $0 < d_i < 1$ (for all i, $1 \leq i \leq n$), the Birth-Death process described here is not a pure birth or pure death process as the population is finite. Thus, the Birth-Death process can be modeled as an aperiodic and irreducible DTMC [56]. In this DTMC model, a_i, b_i and d_i should satisfy the additivity of probability axiom [49]. Then, in this DTMC model, the amount of population is greater than 1. Also, a_i, b_i and d_i should satisfy the additivity of probability axiom. Now, the discrete-time Birth-Death process can be formalized as:

Definition 5.1 *(Transition Probability of Discrete-Time Birth_Death Process)*

```
⊢ ∀ a b d t i j.
  DBLt a b d t i j =
   if (i = 0) ∧ (j = 0) then a 0
   else if (i = 0) ∧ (j = 1) then b 0
   else if (0 < i) ∧ (i-j=1) then d i
   else if (0 < i) ∧ (i = j) then a i
   else if (0 < i) ∧ (j-i=1) then b i
   else 0;
```

This definition leads to the following formalization of the discrete-time Birth-Death process:

Definition 5.2 *(Birth_Death Process Model)*

```
⊢ ∀ X p a b c d n p0.
   DB_MODEL X p a b d n p0 =
   Aperiodic_MC X p ([0,n], POW [0,n]) p0 (DBLt a b d) ∧
   Irreducible_MC X p ([0,n],POW [0,n])p0 (DBLt a b d) ∧
   1 < n ∧ (a 0 + b 0 = 1) ∧
   (∀j.  0 < j ∧ j < n ⇒ (a j + b j + d j = 1)) ∧
   (∀j.  j < n ⇒ 0 < a j ∧ 0 < b j ∧ 0 < d j)
```

In this definition, this process is formally described as an aperiodic and irreducible DTMC, in which the state-space is expressed as a pair `([0, n], POW [0,n])`. The set `[0, n]` represents the population and `POW [0,n]` is the sigma-algebra of the set `[0, n]`. Since the aperiodic and irreducible DTMC is independent of initial distribution, the parameter $\mathtt{p_0}$ in this model is a general function. The other conjunctions shown in Definition 5.2 are the requirements described in the specification of the discrete-time Birth-Death process mentioned above.

Now, we can prove that this discrete-time Birth-Death process has the limiting probabilities.

Theorem 5.1 (Birth-Death Process Exists Limit Probability)

```
⊢ ∀ X p a b d n p₀.
   DB_MODEL X p a b d n p₀ ⇒ (∃ u.  ℙ{x | X t x = i} → u)
```

This theorem can be verified by rewriting the goal with Definition 5.2 and then applying Theorem 4.8.

Now, we can prove that the limit probabilities are the stationary distributions and are independent of the initial probability vector as the following theorem.

Theorem 5.2 (Birth-Death Process Exists Stationary Distribution)

```
⊢ ∀ X p a b d n p₀.
   DB_MODEL X p a b d n p₀ ⇒ (∃ f.  stationary_dist p X f s)
```

We proved this theorem by first instantiating `f` to be the limiting probabilities, `lim (λt. ℙ{x | X t x = i}`, and then by applying Theorem 5.1.

The last two theorems verify that the Birth-Death process holds the steady-state probability vector $V_i = \lim_{t\to\infty} \mathbb{P}\{X_t = i\}$. The computation of the steady-state probability vector V_i is mainly based on the following two Equations (4a) and (4b):

$$v_0 = a_0 v_0 + d_1 v_1 \tag{4a}$$

$$v_i = b_{i-1} v_{i-1} + a_i v_i + d_{i+1} v_{i+1} \tag{4b}$$

Now, these two equations can be formally verified by the following two theorems.

Theorem 5.3 (First Steady-state Probability)

```
⊢ ∀ X p a b d n p₀.
   DB_MODEL X p a b d n p₀ ⇒
   (lim (λt.  ℙ{x | X t x = 0}) =
   a 0 * lim (λt.  ℙ{x | X t x = 0}) + d 1 * lim (λt.  ℙ{x | X t x = 1}))
```

The proof steps use Theorems 4.3 and 5.2 to simplify the main goal and the resulting subgoal can be verified by applying the conditional probability additivity theorem, along with some arithmetic reasoning.

Theorem 5.4 (General Steady-state Probability)

```
⊢ ∀ X p a b d n i p₀.
  DB_MODEL X p a b d n p₀ ∧ i + 1 ∈ [0, n] ∧ i - 1 ∈ [0, n] ⇒
  (lim (λt.  ℙ{x | X t x = i}) = b (i-1) * lim (λt.  ℙ{x|X t x = i-1}) +
                                 a i * lim (λt.  ℙ{x|X t x = i}) +
                                 d (i+1) * lim (λt.  ℙ{x|X t x = i+1}))
```

We proceed with the proof of this theorem by applying Theorems 4.3, 5.2, 5.3 and the total probability theorem along with some arithmetic reasoning.

The general solution of the linear Equations (4a) and (4b) are expressed as:

$$v_{i+1} = \prod_{j=1}^{i+1} \frac{\mathtt{b_{j-1}}}{\mathtt{d_j}} v_0 \tag{5a}$$

$$v_0 = \frac{1}{\sum_{i=0}^{n} \prod_{j=1}^{i+1} \frac{\mathtt{b\ (j-1)}}{\mathtt{d\ j}}} \tag{5b}$$

These two equations are the major targets of the long-term behavior analysis and can be verified in HOL as the following two theorems:

Theorem 5.5 (Equation *(5a)*)

$\vdash$ ∀ X p a b d n i Linit.
DB_MODEL X p a b d n Linit ∧ i + 1 ∈ [0, n] ⇒
(lim (λt. $\mathbb{P}${x | X t x = i + 1}) = lim (λt. $\mathbb{P}${x | X t x = 0}) *
PROD (1, i + 1) (λj. $\frac{b\ (j-1)}{d\ j}$))

The proof of this theorem starts by induction on the variable n. The base case can be verified by Theorem 5.3 and some arithmetic reasoning. The proof of the step case is then completed by applying a lemma that proves the following Equation (6) based on the DB_MODEL, which describes the discrete-time Birth-Death process model, of Definition 5.2:

$$v_{i+1} = \frac{b_i}{d_{i+1}} v_{i+1} \tag{6}$$

The formal proof of Equation (6) is mainly done by induction on the variable i. The base case is proved by applying Theorems 4.3, 5.2 and 5.3 as well as some arithmetic reasoning. The proof of the step case is completed by using Theorem 5.4 along with some arithmetic reasoning.

Theorem 5.6 (Equation *(5b)*)

$\vdash$ ∀ X p a b d n i Linit.
DB_MODEL X p a b d n Linit ∧ i + 1 ∈ [0, n] ⇒
(lim (λ t. $\mathbb{P}${x | X t x = 0}) = $\frac{1}{\mathtt{SIGMA}\ (\lambda i.\ \mathtt{PROD}\ (1,\ i+1)\ (\lambda j.\ \frac{b\ (j-1)}{d\ j})\ (0,\ n+1)}$)

The proof of this theorem begins by rewriting the goal as lim (λ t. $\mathbb{P}${x | X t x = 0}) * SIGMA (λi. PROD (1, i + 1) (λj. $\frac{b_{j-1}}{d_j}$)) (0, n + 1) = 1.

Then we split the summation into two terms: $\frac{b_0}{d_1}$ and SIGMA (1, n + 1) (λi. PROD (1, i + 1) (λj. $\frac{b_{j-1}}{d_j}$)) (0, n + 1). The proof is then concluded by applying Theorems 5.3 and 5.5 and the probability additivity theorem and some real arithmetic reasoning.

After these theorems are verified, the limit probabilities of any state in this model can be calculated by instantiating the parameter n and transition probabilities a, b and d. Thus, it becomes unnecessary for the potential users to employ any numerical arithmetic to analyze the long-term behaviors of this model. The solution, shown in Equations (5a) and (5b), is mainly used to predict safety properties in the development of the population in a long period, in various domains, such as statistics and biological.

Furthermore, when the birth-death coefficients are $b_i = \lambda$ and $d_i = \mu$ (λ and μ are constants) for all the i's in the state-space, the model described in Definition 5.2 represents a classical M/M/1 queueing system [27] (in this case, the average inter arrival time becomes $\frac{1}{\lambda}$ and the average service time is $\frac{1}{\mu}$). For this particular case, our formally verified theorems can be directly applied for analyzing the ergodicity of M/M/1 queueing.

5.2 IID Random Process

In this section, we formally validate that an *Independent and Identically Distributed (IID)* random process (model) is a DTMC.

In probability theory, a collection of random variables is called independent and identically distributed if all of the random variables have the same probability distribution and are mutually independent [10]. The *IID* random process plays an important role in modelling the repeated independent trials, such as Bernouli trails. In HOL, the IID random process can be formally defined as:

Definition 5.3 *(IID Random Process)*

$$
\begin{array}{l}
\vdash \texttt{p X s.}\ \ \textit{iid\_rp}\ \texttt{p X s =} \\
\quad \forall\ \texttt{i.}\ \ \textit{random\_variable}\ \texttt{(X i) p s} \wedge \texttt{FINITE (space s)} \wedge \\
\quad (\forall \texttt{i.}\ \ \texttt{x} \in \texttt{space s} \Rightarrow \{\texttt{x}\} \in \texttt{subsets s}) \wedge \\
\quad (\forall \texttt{i.}\ \ \texttt{i} \in \texttt{space s} \Rightarrow (\texttt{p}_0\ \texttt{i} = \mathbb{P}\{\texttt{x} \mid \texttt{X 0 x = i}\})) \wedge \\
\quad \forall\ \texttt{B st.}\ \ (\texttt{st} \subseteq \{(i,j) \mid \texttt{i} \in \texttt{univ(:num)} \wedge \{B\ j\} \in \texttt{subset s}\}) \Rightarrow \\
\qquad\quad (\mathbb{P}\ \bigcap_{\texttt{(i,j)} \in \texttt{st}}\ \{\texttt{x|X i x = B j}\} = \texttt{PROD}\ (\lambda\texttt{(i,j)}.\ \ \mathbb{P}\{\texttt{x|X i x = B j}\}\ \texttt{st}) \wedge \\
\quad \forall\ \texttt{a i j.}\ \ \mathbb{P}\{\texttt{x|X i x = a}\} = \mathbb{P}\{\texttt{x|X j x = a}\}
\end{array}
$$

where the first conjunction defines this random process as a collection of random variables $\{X_i\}$ (i is a natural number), the second condition defines this random process on a *finite state-space*, the third condition ensures the events associated with the state-space (space s) are in the event space (subsets s), which is a *discrete* space, the next conjunction $\forall$i. i $\in$ space s $\Rightarrow$ (p_0 i = $\mathbb{P}${x | X 0 x = i}) defines a general initial distribution p_0 for all the states in the state-space space s. The last two conditions define the mutual independence and identical distribution properties.

It is important to note that the notion of mutual independence, also called *stochastic mutual independence*, is different from *mutually exclusive* and *pairwise independence* [14]. It refers to the case when the random variables are measurable functions from the set of possible outcomes x to an event set `subsets s`, where events are represented by E_{i_k} and the random variables satisfy

$$\mathcal{P}r(E_{i_1}, \cdots, E_{i_k}) = \prod_{k=0}^{n-1} \mathcal{P}r(E_{i_1}) \cdots \mathcal{P}r(E_{i_k}).$$

Note that the events $E_{i_1} \cdots E_{i_k}$ do not have to be successive. Thus, in Definition 5.3, a set `st` is defined as a subset of a pair set `(i, j)`, {`(i, j) | i` $\in$ `univ(:num)` $\wedge$ {`B j`} $\in$ `subset s`}, in which the index of a random variable `i` can be any natural number, while the event {`B j`} is in the event set `subsets s`.

The last condition $\mathbb{P}${`x|X i x = a`} = $\mathbb{P}${`x|X j x = a`} refers to the property that the random variables in the process have an identical distribution for any event in the event set.

Now, we can prove that a discrete IID random process with finite space is a DTMC using Definitions 4.3 and 5.3 as follows.

Theorem 5.7 (A Finite Discrete IID Random Process is a DTMC)

```
⊢ ∀ X p s p0.
  iid_rp X p s p0 ⇒
  dtmc X p s p0 (λt i j.  ℙ({x | X (t + 1) x = j}|{x | X t x = i})
```

To prove that a finite discrete IID random process is a DTMC, we first have to prove

```
ℙ({x | X (t+1) x = j}|{x | X t x = i}) =
         if i ∈ space s ∧ j ∈ space s then
           ℙ({x | X (t+1) x = j}|{x | X t x = i})
         else 0
```

and then have to prove the second condition in *Markov Property* (Definition 4.1). This step can be executed for two cases: $n = 0$ and $n > 0$. The first case is to prove

```
ℙ({x|X (t+1) x = f j}|{x|X t x = f i}) = ℙ({x|X (t+1) x = f j})
```

which can be verified by applying the mutual independence property of Definition 5.3. The second case can be verified by using some properties of product and the mutual independence.

The verification of the above theorem is one of the prerequisites to formalize random walk and gambler's ruins, which are frequently applied in modelling many interesting systems, such as behavioral ecology[8], financial status prediction (modelling the price of a fluctuating stock as a random walk)[13], etc. The proof of above theorem also means that our formal definition of DTMC can be applied to validate DTMC models.

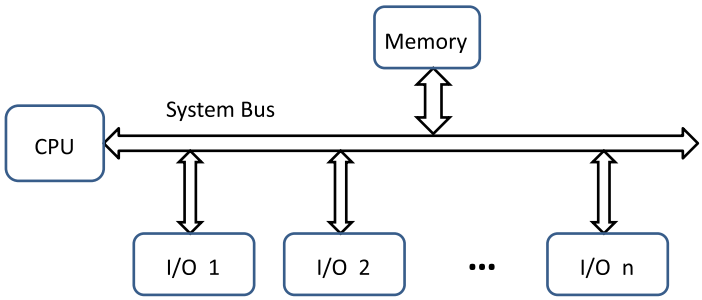

Figure 3: Basic Computer Architecture

6 Applications

In order to illustrate the usefulness of the developed Markov Chain formalization framework, we present in this section the formal performance analysis of two software applications, namely the formal analysis of a program performance and a data structure.

6.1 Formal Analysis of Program Performance

The basic architecture of a modern multi-processor based computer system can be illustrated by Figure 3. Each processor in such a system is usually connected with a memory module and several input/output (I/O) ports. Usually, a main program is designed to control the requests from the devices connected to these I/O ports. Requests from various devices at the end of a CPU burst are independent from the past behavior.

Consider a program that manages a CPU with n I/O devices. It is assumed that the program will finish the execution phase at the end of a CPU burst with probability q_0 and the probability of requests from the device connected with the i^{th} I/O is q_i. Moreover, all devices are assumed to be available, i.e., $0 < q_i < 1$ (for $i = 0, 1, \cdots, n$) and $\sum_{i=0}^{n} q_i = 1$, where n is the number of the I/Os or the devices connected to the CPU. In [56], the behavior of this program can be modeled as an aperiodic and irreducible discrete-time Markov chain, which is shown in Figure 4.

From this diagram, we can obtain the transition probability matrix and formally express it as a function in HOL as:

$$P = \begin{pmatrix} q_0 & q_1 & q_2 & \cdot & \cdot & \cdot & q_n \\ 1 & 0 & 0 & \cdot & \cdot & \cdot & 0 \\ & & & & & & \\ 1 & 0 & \vdots & \cdot & \cdot & \vdots & 0 \\ 1 & 0 & \cdot & \cdot & \cdot & 0 & 0 \end{pmatrix}; \tag{7}$$

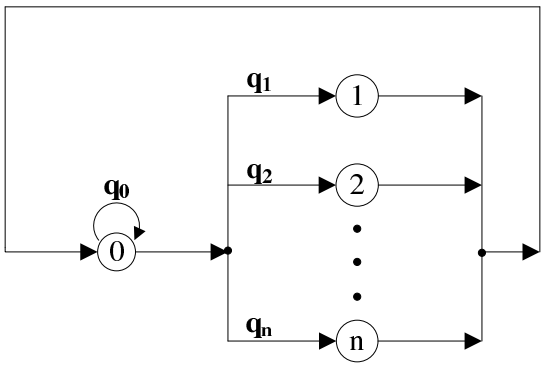

Figure 4: A Discrete-Time Markov Chain Model of a Program

Definition 6.1 *(Program Behavior Transition Probabilities)*

```
⊢ ∀ q t i j.  pmatrix q t i j =
 if (i = 0) then q j else if (j = 0) then 1 else 0
```

In order to evaluate the performance of this program, we can prove some interesting properties of this system. First of all, we verify that there exists a steady-state probability for every state in the state-space. Then, we can prove that the steady-state vector satisfies $v_j = v_0 q_j$ (for all $j = 1, 2, \cdots, m$). Furthermore, the following two equations, which are usually used to analyze the long-term behaviors of a multi-processor, can be verified:

$$v_0 = \frac{1}{2 - q_0} \tag{8}$$

$$v_j = \frac{q_j}{2 - q_0} \tag{9}$$

which are the steady-state probabilities of visiting the CPU (corresponding to Equation (8)) and different devices (corresponding to Equation (9)) in the system.

Now, we first define this model as a predicate in higher-order logic:

Definition 6.2 *(Program Behavior Model)*

```
⊢ ∀ X p q n p0.  PROGRAM_MODEL X p q n p0 =
    Aperiodic_MC X p ([0,n], POW [0,n]) p0 (pmatrix q) ∧
    Irreducible_MC X p ([0,n],POW [0,n]) p0 (pmatrix q) ∧
    (∀i.  i∈[0,n]⇒0 < q i ∧ q i < 1) ∧ (SIGMA (λi.  q i) [0, n] = 1)
```

where the first and second assumptions describe that the program's behavior can be modeled as an aperiodic and irreducible DTMC and the last two conjuncts constraint the probabilities.

Then, we prove that there exists steady-state probabilities for all states in the state-space:

Theorem 6.1 (Existence of Steady-state Probabilities of All States)

$$\vdash \forall\ \mathtt{X\ p\ q\ n\ p_0}.\ \ \mathtt{PROGRAM\_MODEL\ X\ p\ q\ n\ p_0} \wedge \mathtt{0<n} \Rightarrow (\forall \mathtt{j}.\ \ \exists \mathtt{u}.\ \ (\lambda \mathtt{t}.\ \ \mathbb{P}\{\mathtt{x|X\ t\ x\ =\ j}\}) \rightarrow \mathtt{u})$$

The properties expressed using Equations (8) and (9) can be verified as the following two theorems:

Theorem 6.2 (Steady-state Probabilities of Visiting the CPU)

$$\vdash \forall\ \mathtt{X\ p\ q\ n\ p_0}.\ \ \mathtt{PROGRAM\_MODEL\ X\ p\ q\ n\ p_0} \wedge \mathtt{0<n} \Rightarrow (\mathtt{lim}\ (\lambda \mathtt{t}.\ \ \mathbb{P}\{\mathtt{x|X\ t\ x\ =\ 0}\}) = \tfrac{1}{2\ -\ q\ 0})$$

Theorem 6.3 (Steady-state Probabilities of Visiting the j^{th} Devices)

$$\vdash \forall\ \mathtt{X\ p\ q\ n\ p_0}.\ \ \mathtt{PROGRAM\_MODEL\ X\ p\ q\ n\ p_0} \wedge \mathtt{0<n} \wedge \mathtt{j} \in \mathtt{[1,\ n]} \Rightarrow (\mathtt{lim}\ (\lambda \mathtt{t}.\ \ \mathbb{P}\{\mathtt{x|X\ t\ x\ =\ j}\}) = \tfrac{q\ \ j}{2\ -\ q\ 0})$$

If we use probabilistic model checking to analyze the performance of this system, then the steady-state probabilities of visiting each device can only be obtained by solving a group of linear equations. Thus, if the system involves n devices, then the computations would increase linearly. In the case of using simulation for analyzing this model, the final results will be obtained as a vector including many zeroes, which are not accurate enough (an event with very low probability will never be an impossible event). This is because if some q_i ($i \in [0, n]$) becomes very small (as the number of the devices increases) during the simulation process, the accuracy of the calculations is constrained by the underlying algorithms and the available computation resources.

As shown in Theorems 17 and 18, we were able to provide generic results. The HOL code for the above verification comprises of only around 300 lines and the reasoning was based on our foundational results, presented in the previous sections. Moreover, the verified generic results largely reduce the computation time for obtaining steady-state probabilities for the aperiodic and irreducible DTMCs. In fact, the steady-state probabilities computed based on the previous two theorems can also be used to interpret the average visiting time, for example, if the real-time interval is T, then the average number of visits to device j will be v_jT [56] in the long run.

6.2 Formal Analysis of a Data Structure

In software engineering, resource usage is one of the major quality attributes of a software. For example, the amount of memory consumptions by certain data structures, e.g., a linear list, being manipulated in a program is usually of interest in evaluating the performance of this program. The amount of the occupied memory units can be regarded as the population

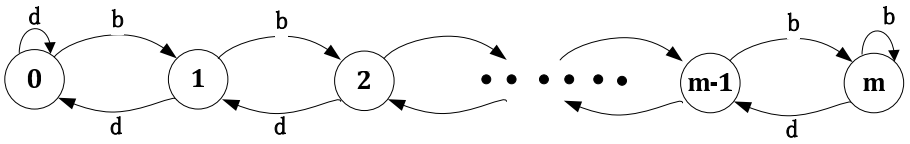

Figure 5: The State Diagram of Data Structure Behavior

in a discrete-time Birth-Death process, where the insertion of a data corresponds to the birth transmission, the release of a memory unit can be considered as the death transmission and the access of a memory unit represents that the system stays in a state. Assuming that this data structure in a program has a stable transition probability, which is independent of time, then the state diagram for this data structure can be depicted as Figure 5, where the transition probabilities are described as:

b_i = P("next operation is an insert" | "current i units of memory is occupied")
d_i = P("next operation is a delete" | "current i units of memory is occupied")

With the assumed stable transition probabilities, we have $b_i = b$ $(i \geq 0)$ and $d_i = d$ $(i \geq 1)$ in the process. We are interested in learning the probabilities of an overflow and underflow in a long run, which can be obtained by computing the steady state probabilities of the full-size of the accessible memory units. Also, we can predict the probability that all the usable memory units are released in a long-run. In order to formally reason about this steady-state probability, we proceed by first formally describing the data structure behaviors by instantiating the discrete-time birth-death process in higher-order logic.

Definition 6.3 *(Transition Probability Functions)*

```
⊢ ∀d.  ra d = λn.  if n=0 then d else 0;
⊢ ∀b.  rb b = λn.  b;
⊢ ∀d.  rd d = λn.  d
```

Then the following model can be used to describe the behavior of this data structure:

Definition 6.4 *(Data Structure Model)*

```
⊢ ∀ X p b d m p0.  Data_Struc_MODEL X p b d m p0 =
  DB_MODEL X p (ra d) (rb b) (rd d) m p0
```

as a discrete-time birth-death process where `b` and `d` are the *birth* and *death* transition probabilities, `m` denotes the amount of useable memory size, p_0 is a general initial distribution. Assuming that the potential number of the memory units (m) is very large $(m > 0)$ for allocation in this model and the parameters satisfy `b` < `d`, we can prove the existence of

steady-state probabilities (v_i, $1 < i$) of this system by applying Theorem 5.1. Then, using Theorem 5.6, it is easy to verify the steady-state probability that all the memories are released in the long-run, i.e, $v_0 = \lim_{t\to\infty} \mathcal{P}r\ (X_t = 0)$ is given by:

$$v_0 = \frac{1 - \frac{b}{d}}{1 - (\frac{b}{d})^m}$$

and it is verified as the following theorem in HOL:

Theorem 6.4 (Steady-state Probability of All Memories Released)

$\vdash \forall$ `X p d b m` $\mathtt{p_0}$. `Data_Struc_MODEL X p b d m` $\mathtt{p_0} \wedge$ `b` $<$ `d` $\wedge$ `0` $<$ `m` $\Rightarrow$
`(lim (`λ `t.` $\mathbb{P}${`x | X t x = 0`}`) =` $\frac{1-\frac{b}{d}}{1-(\frac{b}{d})^m}$`)`

The steady-state probability of i memory units required in such a model is

$$v_i = (\frac{b}{d})^i v_0$$

which can be proved as the following theorem in HOL.

Theorem 6.5 (Steady-state Probability of i Memory Units Required)

$\vdash \forall$ `X p d b m` $\mathtt{p_0}$. `Data_Struc_MODEL X p b d m` $\mathtt{p_0} \wedge$ `b` $<$ `d` $\wedge$ `0` $<$ `m` $\Rightarrow$
`(lim (`λ`t.` $\mathbb{P}${`x | X t x = i`}`) =` $(\frac{b}{d})^i$ `* lim (`λ`t.` $\mathbb{P}${`x | X t x = 0`}`))`

Then, the probability of an overflow is given by

$$bv_m = b(\frac{b}{d})^m \frac{1 - \frac{b}{d}}{1 - (\frac{b}{d})^{m+1}} = \frac{d^{m+1} * (d - b)}{d^{m+1} - b^{m+1}}$$

which means that all memory units available for allocation are used and the probability of a further insertion occurring in a long-run is $\frac{d^{m+1}*(d-b)}{b*(d^{m+1}-b^{m+1})}$. This property can be proved in a theorem as follows:

Theorem 6.6 (Overflow Probability)

$\vdash \forall$ `X p d b m` $\mathtt{p_0}$. `Data_Struc_MODEL X p b d m` $\mathtt{p_0} \wedge$ `b` $<$ `d` $\wedge$ `0` $<$ `m` $\Rightarrow$
`(b * lim (`λ`t.` $\mathbb{P}${`x | X t x = m`}`) =` $\frac{d^{m+1}*(d-b)}{d^{m+1}-b^{m+1}}$`)`

Similarly, the probability of underflow represents the probability that a delete operation will occur when all available memory units are occupied and it is proved as:

Theorem 6.7 (Underflow Probability)

$\vdash \forall$ X p d b m p_0. Data_Struc_MODEL X p b d m $p_0 \wedge$ b $<$ d $\wedge$ 0 $<$ m $\Rightarrow$
(lim (λ t. $\mathbb{P}${x | X t x = 0}) = $\frac{b^{m-1}*(d-b)}{d^m-b^m}$)

Using simulation or probabilistic model checking to analyze this kind of data structure model would involve an enormous amount of computation time and memory. It is also obvious (from Theorem 6.7) that the computation may encounter some errors, like dividing by zero with an increase in the value of m (d and b are both between 0 and 1 and the power of such a small positive number tends to zero). These features are unacceptable while analyzing safety-critical systems. The proposed approach shows quite promising results in this context as it is capable of overcoming the above mentioned limitations, with around 500 lines of HOL code to verify these interesting properties about the given data structure.

7 Conclusions

This paper presents a methodology to formally analyze Markovian systems based on the formalization of DTMCs and classified DTMCs with finite state-space. Due to the inherent soundness of theorem proving, our work guarantees to provide accurate results, which is a very useful feature while analyzing stationary or long-run behaviors of a system associated with safety or mission-critical systems. In order to illustrate the usefulness of the proposed approach, we formalize the Discrete-time Birth-Death process and validate that a discrete IID random process with finite state-space is a DTMC. Moreover, we use the definitions and verified properties of classified DTMCs in analyzing the performance of a couple of software applications, i.e., a program controlling the CPU interactions with its connected devices and a data structure used in a program.

The paper provides a new method to formally analyze DTMCs with finite-state-space and avoid the state-explosion problem or the unacceptable computation time issue which are commonly encountered problems of model checking and simulation, respectively, for analysing the stationary properties of a safety-critical system with a large number of states. Hence, the presented work opens the door to a new and very promising research direction, i.e., integrating HOL theorem proving in the domain of analyzing DTMC systems and validating DTMC models.

Our formalization of DTMCs can be built upon for formally verifying the properties of time-inhomogeneous discrete-time Markov chains and Markov Decision Process (MDP), which will enable us to formally analyze a wider range of systems. We also plan to build upon the formalization of continuous random variables [21] and statistical properties [20] to formalize Continuous-Time Markov Chains (CTMC) to be able to formally reason about statistical characteristics of more complex Markovian models. Furthermore, our work can be applied to validate/formalize various interesting random processes, such as random walk and gambler's ruins, which are widely used in diverse domains, i.e., biology[58], chemistry[57], computer science[55], ecology[8], economics[13], physics[57], psychology[32], etc..

References

[1] R. Affeldt and M. Hagiwara. Formalization of Shannon's Theorems in SSReflect-Coq. In *Interactive Theorem Proving*, volume 7406 of *Lecture Notes in Computer Science*, pages 233–249. Springer, 2012.

[2] C. Baier and J. Katoen. *Principles of Model Checking*. MIT Press, 2008.

[3] R. N. Bhattacharya and E. C. Waymire. *Stochastic Processes with Applications*. John Wiley & Sons, 1990.

[4] J. Bulla. *Application of Hidden Markov Models and Hidden Semi-Markov Models to Financial Time Series*. PhD. thesis, University Gottingen, Germany, 2006.

[5] K.L. Chung. *Markov Chains with Stationary Transition Probabilities*. Springer, 1960.

[6] G. Ciardo, J. K. Muppala, and K. S. Trivedi. SPNP: Stochastic Petri Net Package. In *Workshop on Petri Nets and Performance Models*, pages 142–151, 1989.

[7] A. R. Coble. *Anonymity, Information, and Machine-Assisted Proof*. PhD thesis, University of Cambridge, UK, 2010.

[8] E. A. Codling, M. J. Plank, and S. Benhamou. Random walk models in biology. *Journal of The Royal Society Interface*, 5(25):813–834, 2008.

[9] Coq. http://coq.inria.fr/, 2015.

[10] S. Datta. *Probabilistic Approximate Algorithms for Distributed Data Mining in Peer-to-peer Networks*. University of Maryland, Baltimore County, USA, 2008.

[11] N. J. Dingle, W. J. Knottenbelt, and P. G. Harrison. HYDRA - Hypergraph-based Distributed Response-time Analyser. In *International Conference on Parallel and Distributed Processing Technique and Applications*, pages 215 – 219, 2003.

[12] M. Eirinaki, M. Vazirgiannis, and D. Kapogiannis. Web Path Recommendations Based on Page Ranking and Markov Models. In *Web Information and Data Management*, pages 2–9. ACM Press, 2005.

[13] Eugene F. Fama. Random Walks in Stock-Market Prices. *Financial Analysts Journal*, 21:55–59, 1965.

[14] R. Goodman. *Introduction to stochastic models*. Benjamin/Cummings Pub. Co., 1988.

[15] M.J.C. Gordon. Mechanizing Programming Logics in Higher-0rder Logic. In *Current Trends in Hardware Verification and Automated Theorem Proving*, Lecture Notes in Computer Science, pages 387–439. Springer, 1989.

[16] GreatSPN. http://www.di.unito.it/∼greatspn/index.html, 2015.

[17] P. J. Haas. *Stochastic Petri Nets: Modelling, Stability, Simulation.* Springer, 2002.

[18] O.J. Haggarty, Knottenbelt W.J., and J.T. Bradley. Distributed Response Time Analysis of GSPN Models with MapReduce. In *International Symposium on Performance Evaluation of Computer and Telecommunication Systems*, pages 82–90, 2008.

[19] O. Häggström. *Finite Markov Chains and Algorithmic Applications.* Cambridge University Press, 2002.

[20] O. Hasan, N. Abbasi, B. Akbarpour, S. Tahar, and R. Akbarpour. Formal Reasoning about Expectation Properties for Continuous Random Variables. In *Formal Methods*, volume 5850 of *Lecture Notes in Computer Science*, pages 435–450. Springer, 2009.

[21] O. Hasan and S. Tahar. Formalization of continuous probability distributions. In *Automated Deduction*, volume 4603 of *Lecture Notes in Computer Science*, pages 3–18. Springer, 2007.

[22] O. Hasan and S. Tahar. Reasoning about Conditional Probabilities in a Higher-Order-Logic Theorem Prover. *Journal of Applied Logic*, 9(1):23 – 40, 2011.

[23] J. Hölzl and A. Heller. Three Chapters of Measure Theory in Isabelle/HOL. In *Interactive Theorem Proving*, volume 6898 of *Lecture Notes in Computer Science*, pages 135–151. Springer, 2011.

[24] J. Hölzl and T. Nipkow. Verifying pCTL Model Checking. In *Tools and Algorithms for the Construction and Analysis of Systems*, volume 7214 of *Lecture Notes in Computer Science*, pages 347–361. Springer, 2012.

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

[26] D. H. Jonassen, M. Tessmer, and W. H. Hannum. *Task Analysis Methods for Instructional Design.* Lawrence Erlbaum, 1999.

[27] L. Kleinrock. *Queueing Systems*, volume I: Theory. Wiley Interscience, 1975.

[28] W. J. Knottenbelt. Generalised Markovian Analysis of Timed Transition Systems. Master's thesis, University of Cape Town, South Africa, 1996.

[29] A. N. Kolmogorov. *Grundbegriffe der Wahrscheinlichkeitsrechnung. English translation (1950): Foundations of the Theory of Probability. Chelsea Publishing Co.* Springer, 1933.

[30] P. Kritzinger and F. Bause. *Introduction to Stochastic Petri Net Theroy*. Vieweg Verlag, 1995.

[31] J. A. Kumar. *Statistical Guarantees of Performance for RTL Designs*. PhD thesis, University of Illinois at Urbana-Champaign, USA, 2012.

[32] R. Lambiotte, V. Salnikov, and M. Rosvall. Effect of Memory on the Dynamics of Random Walks on Networks. *CoRR*, abs/1401.0447, 2014.

[33] L. Liu. Formal Reasoning about Classified Markov Chains in HOL, http://hvg.ece.concordia.ca/code/hol/cdtmc/, 2015.

[34] L. Liu, V. Aravantinos, O. Hasan, and S. Tahar. Formal Reasoning about Classified Markov Chains in HOL. In *Interactive Theorem Proving*, volume 7998 of *Lecture Notes in Computer Science*, pages 295–310. Springer, 2013.

[35] L. Liu, V. Aravantinos, O. Hasan, and S. Tahar. On the Formal Analysis of HMM Using Theorem Proving. In *Formal Methods and Software Engineering*, volume 8829 of *Lecture Notes in Computer Science*, pages 316–331. Springer, 2014.

[36] L. Liu, O. Hasan, and S. Tahar. Formalization of Finite-State Discrete-Time Markov Chains in HOL. In *Automated Technology for Verification and Analysis*, volume 6996 of *Lecture Notes in Computer Science*, pages 90–104. Springer, 2011.

[37] L. Liu, O. Hasan, and S. Tahar. Formal Analysis of Memory Contention in a Multiprocessor System. In *Formal Methods: Foundations and Applications*, volume 8195 of *Lecture Notes in Computer Science*, pages 195–210. Springer, 2013.

[38] L. Liu, O. Hasan, and S. Tahar. Formal Reasoning About Finite-State Discrete-Time Markov Chains in HOL. *Journal Computer Science and Technology*, 28(2):217–231, 2013.

[39] D.J.C. MacKay. Introduction to Monte Carlo Methods. In *Learning in Graphical Models, NATO Science Series*, pages 175–204. Kluwer Academic Press, 1998.

[40] Maple. http://www.maplesoft.com, 2015.

[41] MARCA. http://www4.ncsu.edu/∼billy/marca/marca.html, 2015.

[42] Mathematica. www.wolfram.com, 2015.

[43] T. Mhamdi, O. Hasan, and S. Tahar. On the Formalization of the Lebesgue Integration Theory in HOL. In *Interactive Theorem Proving*, volume 6172 of *Lecture Notes in Computer Science*, pages 387–402. Springer, 2010.

[44] T. Mhamdi, O. Hasan, and S. Tahar. Formalization of Entropy Measures in HOL. In *Interactive Theorem Proving*, volume 6898 of *Lecture Notes in Computer Science*, pages 233–248. Springer, 2011.

[45] Mobius. http://www.mobius.illinois.edu/, 2015.

[46] J. R. Norris. *Markov Chains*. Cambridge University Press, 1999.

[47] D. A. Parker. *Implementation of Symbolic Model Checking for Probabilitics Systems*. PhD thesis, University of Birmingham, UK, 2002.

[48] PEPA. http://www.dcs.ed.ac.uk/pepa/, 2015.

[49] K. G. Popstojanova and K. S. Trivedi. Failure Correlation in Software Reliability Models. volume 49, pages 37–48, 2000.

[50] PRISM. http://www.prismmodelchecker.org, 2015.

[51] M. Sczittnick. MACOM - A Tool for Evaluating Communication Systems. In *International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, pages 7–10, 1994.

[52] K. Sen, M. Viswanathan, and G. Agha. VESTA: A Statistical Model-Checker and Analyzer for Probabilistic Systems. In *IEEE International Conference on the Quantitative Evaluation of Systems*, pages 251–252, 2005.

[53] SHARPE. http://people.ee.duke.edu/∼chirel/irisa/sharpegui.html, 2015.

[54] W. J. Steward. *Introduction to the Numerical Solution of Markov Chain*. Princeton University Press, 1994.

[55] L.M. Surhone, M.T. Timpledon, and S.F. Marseken. *Random Walk: Computer Science, Lévy Flight, Markov Process, Integer, Pascal's Triangle, Stirling's Approximation, Factorial, Law of the Iterated Logarithm, Central Limit Theorem, Markov Chain*. Betascript Publishing, 2010.

[56] K. S. Trivedi. *Probability and Statistics with Reliability, Queuing, and Computer Science Applications*. John Wiley & Sons, 2002.

[57] N.G. Van Kampen. *Stochastic Processes in Physics and Chemistry*. North-Holland Personal Library. Elsevier Science, 2011.

[58] D.J. Wilkinson. *Stochastic Modelling for Systems Biology*. Chapman & Hall/CRC Mathematical and Computational Biology. CRC Press, 2011.

[59] YMER. http://www.tempastic.org/ymer/, 2015.