Uncertainty Analysis of Phased Mission Systems with Probabilistic Timed Automata

Zhaoguang Peng  
Reliability Research and Analysis Center  
China Ceprei Laboratory  
Guangzhou, China  
Email: ghuapeng@gmail.com

Yu Lu*  
School of Computing Science  
University of Glasgow  
Glasgow, United Kingdom  
* Email: y.lu.3@research.gla.ac.uk

Alice Miller  
School of Computing Science  
University of Glasgow  
Glasgow, United Kingdom  
Email: alice.miller@glasgow.ac.uk

Abstract—A phased mission is one in which the requirements may alter over time. We present a novel approach to analyse phased mission systems using probabilistic timed automata (PTA). We show how to construct PTA models which allow one to reflect system uncertainty, and how to analyse these models using the PRISM probabilistic model checker. We illustrate our approach via a simple case study, namely path planning for a Mars exploration rover, since the mission of the rover can be expected to be an instance of phased mission systems.

Keywords—phased mission systems; uncertainty analysis; formal methods; probabilistic timed automata; probabilistic model checking

I. INTRODUCTION

A phased mission [1] is one in which the requirements may alter over time. Mission success depends on the success of all of the individual phases. For example, consider an aircraft whose operation consists of a number of phases, including: taxiing, taking off, climbing to the correct altitude, cruising, descending, landing, and taxiing back to the terminal. Uncertainty analysis allows us to predict phase failure probabilities and thus compute the overall probability of mission failure.

In a phased mission system (PMS), different phases may have different logical structures. The system unit may be involved in different tasks, and vary in each phase. For example, space monitoring and control communication system refers to the systems which track, monitor and control the various stages of the spacecraft and its payload. Its reliability is directly related to successfully launching spacecraft, changing orbit and other tasks.

Many aerospace applications are PMSs. Communication monitoring varies between the stages but in order to understand the overall communication program, communication within each stage must be analysed.

In recent years the analysis of PMSs has developed into an important research area. Some researchers have focused on the problem of phase independence, mainly using Fault Tree Analysis and state-space-based analytical methods, including Markov methods and Petri Nets [2], [3]. But, with the growing number of the system construction, it is more difficult to be applied to these systems.

Other approaches for analysing PMSs involve simulation and testing. Simulation is insufficient for analysing this type of complex system since it only allows for partial coverage. We consider a numerical analysis technique based on Probabilistic Timed Automata (PTAs). PTAs allow us to model systems which have both probabilistic and timed behaviours, and to analyze them with respect to a range of associated properties.

In this paper, we propose a method based on PTAs to evaluate quantitative properties of PMSs. This method allows us to solve specific problems in the practical operation of PMS systems. In general, traditional analysis methods do not allow for the modelling of uncertainty within a system. For example, they do not allow for uncertainty in event ordering, or in the timing of events. It might not be possible to determine a priori the exact time at which an event will occur, it is more likely that a time interval during which the event would occur (with a given probability) could be predicted. Without the inclusion of this type of uncertainty, PMS planning is unreliable, and consequently PMSs may be unpredictable and unsafe.

This paper describes how PTAs can be used to model PMSs with uncertainty. The aim of the paper is to promote the adoption of such an approach for modelling PMSs. PTAs are analysed using probabilistic model checking. Probabilistic model checking involves creating a probabilistic model (based on a Markov chain) of a system. In this model events have probabilities and transitions may have associated rewards. Properties of the overall system can be verified using probabilistic stochastic logic. Examples of such properties involve the maximum or minimum probabilities of the likelihood of a situation occurring within a given time interval, or the expected cumulative reward.

The method is described in several steps as follows. First, we introduce the concept of a PMS and its technology. Second, we give technical background on traditional techniques and time dependent techniques. Third, we introduce probabilistic model checking technology, and present the definition and formal specification of quantitative properties of PTAs. Then, a case study is provided to illustrate the approach, and we use formal methods to analyse the PTA models. Finally we present our conclusions and outline directions for future research.

II. ANALYSIS TECHNIQUES

A. Traditional Techniques

In order to analyse PMSs, a number of analysis techniques have been developed, such as Event Tree Analysis (ETA),
Fault Tree Analysis (FTA) [4], and Binary Decision Diagrams (BDDs) [5], [6]. We outline some of these techniques below, for more details see [7].

ETA involves determining a trace of events that lead to a failure/accident (but not the specific event that led to it). On the other hand, FTA is a method that examines relationships between the subsystems and components of a system in order to find the root cause of a failure [8]. A graphical representation of the different combinations of events that lead to failure is used to identify failure causes, be they due to software error, user error, or otherwise [9]–[11]. ETA analysis is carried out in two ways. One of these involves the identification of sets of component failures that can, when combined, lead to system failure. These sets are known as cut sets, and FTA involves logically determining minimal cut sets MCSs that lead to a specific error event (the Top Event). It is also possible to determine the probability of each node in the fault tree [12], [13].

Binary Decision Diagrams (BDDs) have been used in the context of ETA in order to allow for the efficient verification of more expressive properties [14]. BDDs are an effective method for describing Boolean expressions, applied in system design and certification. In most cases, the use of BDDs to describe Boolean expressions uses less storage space than other methods. Whereas any BDD can be translated to an ETA model, the converse is not true. ETA involves more factors than can be modelled using a BDD, such as human factors and environmental factors. BDDs have often been used to analyse fault trees. The combination of ETA, with its rigorous logical structure, tree pattern, and ability to reveal fault causes, and BDDs which allow for the quantitative calculation of failure probability, is a powerful technique when applied in the context of reliability engineering. When BDDs are used in the context of ETA, the fault tree is not analysed directly, but rather an abstracted BDD representation. For this reason translation errors may be introduced [15].

B. Real Time Techniques

There have been efforts in the past to introduce the notion of real time to ETA. For example, in [16], a time-dependent methodology for fault tree analysis is proposed in which time is expressed as a function of the available information.

In other work, ETA has been augmented using temporal formulas, time dependencies and temporal fault trees [17]–[19]. The last two of these are extensions of traditional fault trees and designed for the safety analysis of safety critical systems.

Another popular methods that allow one to model time is Stochastic Petri Nets (SPNs). SPNs allow one to calculate aggregated performance values (in a similar way to rewards in probabilistic model checking). Typical values that are measured are the average number of tokens and average token delay [20]. Transition delays are assumed to be random variables from a negative exponential distribution.

Markov models (such as Discrete Time Markov Chains (DTMCs), Markov Decision Processes (MDPs), and Continuous Time Markov chains (CTMCs)) consist of a countable set of states together with a probabilistic transition matrix which determines the probability of moving between states (or the associated rate, drawn from a probability distribution, for CTMCs). MDPs allow us to also model non-deterministic transitions. Markov models allow us to verify a range of probabilistic properties, what is the maximum (minimum) probability that the system reaches a given state within a certain time period? Or: what is the expected fuel consumption in the first 10 seconds? Markov models cannot address all properties that can be verified using TTBs. An example of such a property is: how long does it take for an error to occur, after the initial failure has occurred?

III. Probabilistic Model Checking

In this paper, we present the analysis of probabilistic timed automata using the PRISM probabilistic model checker [21].

A. Probabilistic Timed Automata

Full details about probabilistic timed automata (PTAs) can be found in [22], [23]. We outline the important aspects in this section.

PTAs allow us to use the real-valued clocks of timed automata, together with the discrete probabilistic choice of MDPs. PTAs have real-valued clocks and, like MDPs [24] and therefore allow us to model systems with a range of different characteristics (non-determinism, probability, and real time).

![Fig. 1. An example of a probabilistic timed automaton.](image-url)

In this section, we illustrate a number of basic PTA concepts using the example in Fig. 1. The Figure illustrates a probabilistic timed automaton, with clock $t$ and integer variable $try$, modelling a simple probabilistic communication protocol. In the protocol, a sender repeatedly attempts to transmit a message over an unreliable channel. The probability that the sender’s transmission fails due to that the channel is unreliable is 0.05, and the sender successfully transmit the message to the receiver with probability 0.95. If message data from the
sender is lost, the sender suspends its activity, and there is a delay (of between 4 and 6 time units) before the sender tries to retransmit its message (up to \( M - 1 \) times).

The control states of the automaton model, “\( \text{state} = 0 \)”, “\( \text{state} = 1 \)”, “\( \text{state} = 2 \)”, “\( \text{state} = 3 \)”, have the meaning of “transmit”, “wait”, “quit”, and “finish” respectively. They are depicted as the nodes (circles) of the underlying graph, and the available transmissions between these control states are indicated as the edges (with arrow) of the graph. In the initial state, “\( \text{state} = 0 \)”, shown as the extra border, a communication is being initialised by the sender along the transmission channel. After between 2 and 3 time units, the sender attempts to send the message, and with probability 0.95 message is sent correctly, meantime with probability 0.05 message is lost. In “\( s = 1 \)”, when 4 to 6 time units have elapsed from whenever the message is lost, the sender tries to retransmit the message.

**Definition 1.** A probabilistic timed automaton PTA is a tuple of the form \( (L, l_0, \Sigma, \text{inv}, \text{prob}) \) where:

- \( L = \{l_0, l_1, l_2, \ldots, l_n\} \) is a finite set of locations;
- \( l_0 \in L \) is the initial location;
- \( \chi = \{x, y, z, \ldots\} \) is a finite set of clocks;
- \( \Sigma = \{a, b, c, \ldots\} \) is a finite set of events, of which \( \Sigma_u \subseteq \Sigma \) are declared as being urgent;
- the function \( \text{inv} : L \rightarrow CC(\chi) \) is the invariant condition;
- the finite set \( \text{prob} \subseteq L \times CC(\chi) \times \Sigma \times \text{Dist}(2^L \times L) \) is the probabilistic edge relation.

Note that clocks are real-valued. The values of the clocks synchronise and increase together over time. Transitions and states may have guards and invariants over clock variables and other variables which indicate when transitions can occur and how long can be spent in a state. In our example, the transition between states \( \text{state} = 0 \) and \( \text{state} = 1 \) (or \( \text{state} = 2 \)) has the clock guard \( t \geq 2 \). The state \( \text{state} = 0 \) and \( \text{state} = 3 \) have the invariant \( t < 3 \) and \( t < 6 \) respectively.

The semantics of PTAs are formally defined as an infinite state MDP. As clocks are real-valued the MDP will have an infinite state-space (both in terms of set of states, and the set of transitions). Since model-checking algorithms are designed to work on finite state spaces, the analysis of PTAs requires some form of abstraction, to a finite state representation. PTAs have been used to verify a variety of protocols, e.g. the CSMA/CD back-off protocol [25], the FireWire root contention protocol [26], and the IPv4 Zeroconf protocol [22].

**B. The Probabilistic Model Checker PRISM**

PRISM is a probabilistic model checker, which can be used to model, and verify systems that exhibit random or probabilistic behaviour based on different types of probabilistic models, such as DTMCs, CTMCs, MDPs, and PAs. PRISM has recently incorporated support for the analysis of probabilistic real-time systems, using PTAs. PRISM was developed and is maintained by researchers from University of Oxford, University of Birmingham, and University of Glasgow in UK. It has been used to analyse many different industrial applications, such as communication and multimedia protocols, security protocols, dynamic power management, biological systems, and autonomous systems.

In this paper our properties are expressed using Continuous Stochastic Logic (CSL) [28], [29]. This logic is an extension
pta
const int M;
module sender
    state : [0..3] init 0;
    try : [0..M] init 0;
    t : clock;
endmodule

invariant
    (state = 0 => t <= 3) & (state = 3 => t <= 6)
endinvariant

[transmit] state = 0 & t >= 1 & try < M -> 0.95 : (state’ = 1) +
0.05 : (state’ = 3) & (try’ = try+1) & (t’ = 0);

[retransmit] state = 3 & t >= 4 -> (state’ = 0) & (t’ = 0);

[terminate] state = 0 & try == M -> (state’ = 3);
endmodule

rewards
    "energy"
    (state=0) : 3;
endrewards

Fig. 3. The PRISM code of the probabilistic timed automaton in Fig. 1.

of: Computation Tree Logic (CTL) [30]; probabilistic CTL (PCTL), which is an extension of CTL for discrete time stochastic systems (PCTL) [31]; and Timed CTL, which is an extension of CTL for continuous time non-stochastic systems (TCTL) [32]. Formulas in CSL are either state formulas (which can be either true or false in any given state), and path formulas which apply to entire paths in a model.

Definition 2. Let \( a \in AP \) be an atomic proposition, \( p \in [0,1] \) be a real number, \( \in \{\leq, <, >, \geq\} \) be a comparison operator, and \( I \subseteq \mathbb{R}_{\geq 0} \) be a non-empty interval. The syntax of CSL formulas over the set of atomic propositions \( AP \) is defined inductively as follows:

- **true** is a state-formula.
- Each \( a \in AP \) is a state formula.
- If \( \Phi \) and \( \Psi \) are state formulas, then so are \( \neg\Phi \) and \( \Phi \land \Psi \).
- If \( \Phi \) is a state formula, then so is \( S_{\text{gap}}(\Phi) \).
- If \( \varphi \) is a path formula, then \( P_{\text{gap}}(\varphi) \).
- If \( \Phi \) and \( \Psi \) are state formulas, then \( X_I\Phi \) and \( \Phi U_I\Psi \) are path formulas.

Full details of CSL can be found in [28], [29]. Two of the main type of CSL properties are:

- **\( S_{\text{gap}}(\Phi) \)**: the steady-state probability for a state satisfying \( \Phi \) satisfies \( \cdot \). An example property is \( S_{\leq 0.5}(x == 4) \).
- **\( P_{\text{gap}}(\varphi) \)**: the probability measure of the paths satisfying path property \( \varphi \) satisfies \( \preceq p \). An example property is \( P_{\geq 0.5}(p U q) \) where \( p \) and \( q \) are defined atomic propositions and \( U \) the until operator.

Another important class of property involves the **P** operator, which indicates the probability of an event. For example, in this paper we use this operator within PRISM to calculate the probability of a robot completing a task within 30 minutes and within 15 minutes respectively.

It is also possible to express properties to measure cumulative rewards using the **R**. We do not use properties of this form in this paper, but examples of such properties can be found on the Property Specification section of the PRISM website.

IV. FORMAL ANALYSIS

A. Formal Modelling of A Mars Exploration Rover

In this section, we present a case study to illustrate the role of PTAs and probabilistic model checking for the analysis of PMSs: a Mars exploration rover [33] exploring an unknown region. The mission of Mars exploration rover can be expected to be an instance of phased mission systems [34].

Fig. 4. NASA Mars exploration rover.

In Fig. 4, we illustrate the Mars exploration rover. The key purpose of the mission of the Mars exploration rover is to make it to pre-defined positions to undertake practical and systematic explorations. Specifically, the Mars exploration rover is planned to achieve different day-to-day conventional missions. Based on [34], a number of major missions can
summarised as follows: (1) waking up at a specific time, (2) receiving commands from the base on Earth, (3) navigating to a specific destination, (4) carrying out surface operations, (5) processing data and sending it to the base (downlink), and (6) entering a sleep mode. The routine operations can consist other phases to be performed according to individual mission, we only analyse the underlying 6 phases that are commonly used.

This example is common in the field of artificial intelligence. The rover attempts to reach a destination by following instructions from a controller. In this case, a rover moves to the destination by command. A region is divided into 12 parts. These consist of 11 numbered parts (numbered from 0 to 10) and one blocked region which cannot be passed by the rover (indicated by shading in Fig. 5).

![Energy Consumption of the Mars exploration rover in Each Area.](image)

Fig. 5. Energy Consumption of the Mars exploration rover in Each Area.

<table>
<thead>
<tr>
<th>Direction</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>East</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>South</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>7</td>
<td>8</td>
<td>4</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>West</td>
<td></td>
<td>4</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td></td>
<td></td>
<td>5</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>North</td>
<td>2</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td></td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>6</td>
<td>8</td>
</tr>
</tbody>
</table>

Table I shows the time spent by the robot in each area moving in each direction.

On collision with the boundary of the obstacle and the perimeter of the exploration area, the robot will be bounced back, and have to re-select the path of the route. For example, when the robot gets the command that it should move forward to area 4 from area 0, there is an 80% probability that it will complete the task, a 10% probability that it will hit the exploration perimeter and return to area 0, and a 10% probability that it will move to area 1.

B. Uncertainty Analysis

Using the model checker PRISM, we show that the probability of the robot completing the task within 30 minutes is 0.84134. The probability of the robot completing the task within 15 minutes is 0.05184. Fig. 6 shows the relationship between the probability and time to complete the task.

The PTA model in the case of minimal energy consumption is shown in Fig. 7(a). The route of the robot in this case is as shown in Fig. 7(a). The optimal route is: 0 → 1 → 5 → 8 → 9 → 10. The minimum energy consumption of the robot moving from area 0 to area 10 is 13.3W. If the direction of the robot deviates, the robot should reroute from its new position in such a way as to minimise energy consumption.

The PTA model in the case of minimal time consumption is as shown in Fig. 7(b). The shortest path is as shown in Fig. 9(b). The best route is: 0 → 4 → 7 → 8 → 9 → 10.
Similarly, the time the robot spends moving from area 0 to area 10 is at least 22.8 minutes. There is one best direction to take in each area. Once the robot enters an area, it should reroute from its new position in such a way as to minimise time consumption.

V. CONCLUSION AND FUTURE WORK

In this paper we consider an automated verification approach: probabilistic model checking, to analyse phased mission systems. A Probabilistic timed automaton is used to solve an uncertainty problem in PMS analysis. Using an example, we show how to construct PTAs for PMSs and use probabilistic model checking techniques to analyse the resulting models. The aim of the paper is to promote the adoption of PTAs and probabilistic model checking within this context. We have illustrated our approach using a simple case study involving a Mars explorer robot, for which we verify quantitative properties using the PRISM model checker.

REFERENCES


Fig. 9. Results of the Mars exploration rover to reach a given target.


