Towards An Accurate Reliability, Availability and Maintainability Analysis Approach for Satellite Systems Based on Probabilistic Model Checking

Khaza Anuarul Hoque, Otmane Ait Mohamed
Concordia Univeristy
Montreal, Canada
Email: {k_hoque,ait}ece.concordia.ca

Yvon Savaria
Polytechnique Montréal,
Montreal, Canada
Email: yvon.savaria@polymtl.ca

***Abstract*—From navigation to telecommunication, and from weather forecasting to military, or entertainment services - satellites play a major role in our daily lives. Satellites in the Medium Earth Orbit (MEO) and geostationary orbit have a life span of 10 years or more. Reliability, Availability and Maintainability (RAM) analysis of a satellite system is a crucial part at their design phase to ensure the highest availability and optimized reliability. This paper shows the formal modeling and verification of RAM related properties of a satellite system. In a previously reported approach, time between possible failures and time between repairs are assumed to follow an exponential distribution, which does not represent a realistic scenario. In contrast, in our work, discrete time delays in the classical Continuous Time Markov Chain (CTMC) are approximated using the Erlang distribution. This is done by approximating nonexponential holding time with several intermediate states based on a phase type distribution. The RAM properties are then verified using the PRISM model checker. We present and compare modeling results with those obtained with a previously reported approach that demonstrate an improved modeling accuracy.**

I. Introduction

With the increasing need of efficient communication and navigation, satellites have become crucial parts of our day-to-day lives. The emerging market of micro, nano and pico satellites, with their low manufacturing cost, has enabled new space entrepreneurs, researchers and universities to launch their own satellites for commercial and research purposes. Apart form research, commercial and government satellites are often used for remote sensing, weather forecast, emergency communication and military purposes. Reliability, Availability and Maintainability (RAM) analysis of satellite systems is an important part in their design phase. An early RAM analysis of these safety-critical systems could allow their designer to adopt proper maintainability strategies to achieve a high reliability and maximum availability satisfying the requirements. This paper proposes a means to help designers achieve this goal.

Formal verification technique is a powerful approach for verification of complex systems. Model checking [1] is an automatic verification technique for finite state concurrent systems. In model checking, the system specifications are written in terms of propositional temporal logic and the verification procedure is an exhaustive search of the state space of the design. This paper proposes a means by which formal verification methods can be applied for the reliability, availability and maintainability evaluation of satellite systems. In particular, the focus is on *Probabilistic model checking* [2]. Probabilistic model checking is used to verify the systems whose behaviors are stochastic in nature. It is mainly based on the construction and analysis of a probabilistic model, typically a Markov chain or a Markov process. These models are constructed in an exhaustive fashion. Indeed, the model explores all possible states that might occur in a system. Probabilistic model checking can be used to analyze a wide range of dependability properties. By contrast, in discrete-event simulations, approximate results are generated by averaging results obtained from large number of random samples. Probabilistic model checking applies numerical computation to provide exact and accurate results.

The goal of the paper is to show how accurately a satellite system can be modeled and how their RAM related properties are verified using probabilistic model checking. Probabilistic model checking has already been successfully applied to many areas for trustworthy hardware and software designs in the area of avionics and aerospace [3], [4]. However, verification of large and complex satellite systems is still a challenge. In [5], authors proposed a probabilistic model checking approach to perform RAM analysis of satellite systems using PRISM [6] and claimed this to be the first in this area. Their model is specified as a Continuous Time Markov Chain (CTMC) model, and the failure and repair rates of a satellite are assumed to be constant. As a result, the time to failure and repair are both defined as random variables with an exponential distribution. However, this does not represent the real life scenario. As the PRISM model checker tool, by default, supports only exponential distributions, authors mentioned it as a main limitation of their approach.

According to the definition of a classical CTMC, the transition delay between the states are exponentially distributed. However, it is still possible to approximate discrete delays in CTMCs using special techniques, such as those leading to Erlang distributions [7]. Our work is motivated by this goal. In this paper, we model a satellite system as a CTMC with an improved accuracy. In our model, the failure transitions are modeled using exponential distribution, but unlike the approach adopted in [5], for modeling the maintenance and repair related transitions, we approximate the discrete time delay using the Erlang distribution. Such modeling approach more accurately models periodic preventive maintenance or

978-3-9815370-4-8/DATE15/©2015 EDAA

test and diagnosis leading to such maintenance. Then we use the PRISM model checker tool to evaluate the RAM related properties showing significant differences in some of the system properties.

The remaining of the paper is organized as follows. Section 2 discusses the related works. Section 3 describes the preliminaries about probabilistic model checking and property specification in PRISM. The proposed modeling approach including modeling of discrete delays using Erlang distribution is discussed in detail in section 4, and in section 5, we show how qualitative analysis is performed using PRISM. We also present modeling results and compare them with those obtained using a traditional approach in section 5. Section 6 concludes the paper with future research directions.

II. Related works

Due to its accuracy and exhaustive nature, formal verification is widely applied in verification of safety critical systems. In [8], authors presented the Architecture Analysis and Design Language (AADL) centered on a component-based modelling approach to system-software co-engineering of real-time embedded systems for aerospace applications. In [9], the authors showed a model checking based approach to verify if an abstract spacecraft meets the mission requirements and how the model behaves if mission parameters change. A methodology using probabilistic model checking to analyze soft-error effects in FPGA-based aerospace systems was proposed in [4]. Authors showed how rescheduling data flow graphs help to achieve optimized reliability when compared to redundancy-based solutions.

In [10], the model checking tool Murϕ [11] was used to model the Entry, Descent and Landing phase of the Mars Polar Lander. An exhaustive search was performed using the model checker to find the states violating the Murϕ invariants. The results showed that the thrust of the descent engine, controlled by pulse width modulation, should only be used above a certain altitude. Model checking of mission critical flight software to explore all the possible behavior of software design to detect faults was presented in [12]. Authors in [13] implemented model checking to the flight software from NASA's Deep Space One (DS1) mission that was implemented using the C language and detected problems in that model. Recently, a symbolic model checking approach to verify the design of an embedded satellite software control system called Attitude and Orbit Control System (AOCS) was proposed in [14]. The Ada implementation of the system was modeled at a detailed level and verified using the NuSMV-2 model checker [15]. In [5], authors claimed the first use of probabilistic model checking to perform RAM analysis of satellite systems. However, use of exponential distribution to model both failure and maintenance is a major limitation of their work. Our work aims at overcoming this limitation and improving the accuracy of their proposed modeling approach.

III. Preliminaries

A. Probabilistic Model Checking

Probabilistic model checking is based on two main components: the construction and the analysis of a probabilistic model of the system, typically a Markov chain. In this paper, we specifically focus on Continuous-Time Markov chains (CTMCs), that are widely used for reliability and performance analysis. A CTMC involves a set of states S and a transition rate matrix $\mathbf{R} : S \times S \rightarrow \mathbb{R}_{\geq 0}$. The delay before which a transition between states s and s' takes place is specified by the rate $\mathbf{R}(s, s')$. The probability that a transition between the states s and s' might take place within time t is given by $1 - e^{-\mathbf{R}(s,s') \times t}$. Hence, no transitions will take place if $\mathbf{R}(s, s') = 0$. Exponentially distributed delays are appropriate for modelling electronic component lifetimes and inter-arrival times. The model-checking approach to performance and dependability analysis requires both a model of the system under consideration and the desired properties for performance/dependability evaluation. In the case of stochastic modelling, such models are typically CTMCs. The properties are usually expressed in some form of extended temporal logic such as Continuous Stochastic Logic (CSL) [16], a stochastic variant of the well-known Computational Tree Logic (CTL) [2].

B. PRISM modeling and Property Specification

PRISM [6] is a well known tool for the formal modeling and verification of stochastic systems. The current version of the tool supports four types of probabilistic models: discrete-time Markov chains (DTMCs), continuous-time Markov chains (CTMCs), discrete-time Markov decision processes (MDPs) and probabilistic timed automata (PTA).

```
module adder
 s : [0..2] init 0;
 [] (s = 0) -> lambda_1 : (s' = s + 1);
 [] (s = 1) -> lambda_2 : (s' = s + 1);
endmodule
```

Sample PRISM code

A PRISM model is defined as the composition of one or more elementary systems known as `modules`. The state of each module can be represented by a set of finite ranged `variables`. The global state of the model at any point can be determined by evaluating the values of those module (state) variables. A set of guarded commands specifies the behaviour of an individual module. For the case of a CTMC, it can be represented as: `[] <guard>` $\rightarrow$ `<rate> : <action>`. The `guard` is a predicate over the variables of all the modules in the model [17]. The update is specified by `rate` and `action`. A `rate` is an expression which evaluates to a positive real number. The term `action` describes a transition of the module specifying how its variables should be updated. The interpretation of the command is that if the guard is satisfied, then the module is allowed to make the corresponding transition with that associated rate. A very simple command for a module with only one variable z might be expressed as: `[] <z = 0>` $\rightarrow$ `7.5 : <z' = z + 1>`, which states that, if z is equal to 0, then it will be incremented by one and this action that occurs with rate 7.5.

Let us assume that we have a system with error detection capability. The Markov model of such a system as shown in Fig. 1, can be built with three discrete states (S0: fully operational, S1: faulty but with fault undetected, and S2: fault detected, failed) representing the system's status. The number of states can be easily changed, depending on the model specificity. The failure rates λ_1 and λ_2 are constant

Fig. 1: A simple Markov chain to illustrate failure occurrence

between states. This system can be described using the PRISM modeling language as shown in the sample PRISM code.

To analyze a system, it is required to specify one or more properties. The property specification language in PRISM is mainly based on temporal logic, which offers an unambiguous means of describing a broad range of properties [18]. The language includes operators from PCTL [19], CSL [16] and its extensions [20]. PRISM emphasizes on quantitative properties. For example, PCTL allows the expression of logical states such as "*what is the probability that the system will eventually fail ?*", expressed as `P =? [F fail]`. Here `fail` is the label that refers to the faulty states. Property expressed as `P=? [G<=20.0 !"failed"]` uses a bounded form of the operator `G` to make a query about failure probability for a bounded mission time. This property is satisfied by all paths that spend at least 20 time units in the fault-free states, that is, all paths where the first fault occurs after time 20. Rather than asking PRISM to compute probabilities of interest, one can also assert bounds on the probability of certain sets of paths and verify with PRISM whether these bounds actually hold. The property `P>0.99 [G<=20.0 (state!=3)]` is true in a state `s` of the Markov chain if the set of paths that start from `s` and do not reach state 3 in the first 20 time units has a probability of at least 0.99. Then this property holds for the system if the initial state satisfies the probability constraint.

The PRISM property specification language also provides an operator R to evaluate the expected value of a random variable defined by a reward structure. For example, let us assume a system that provides a specific service when it is in the operational state. So, a reward structure *operational* can be defined by assigning a state reward of 1 to all the operational states in the model. The PRISM property specification language allows us to reason about the amount of reward accumulated over a specific period of time, that can be expressed as `R {"operational"} = ? [C < T]`, which refers to the expected value of the cumulative operational time of the system in the time interval [0, T].

IV. Formal Modeling of Satellite Systems

A. Markov Modeling of Failure and Deterministic Delay

Markov models are widely used for reliability and availability analysis of complex systems. A Markov model is a directed graph where the nodes represent system states and the arcs represent transition rates between the states. The probability of a state transition for a classic Markov model is assumed to depend only on the current state. This is equivalent to assuming that failure rates are constant and that failure occurrence is a Poisson process. The satellite failure rates (and unscheduled interruptions as well, more detail in next section) are assumed for this work to follow the Poisson process. This implies that the time between two consecutive failures or unscheduled interruption events is exponentially distributed.

A satellite exposed to a harsh radiation environment will eventually fail, requiring repair or replacement. Hence, it is important to have a model that can represent maintenance effects on a system's condition as well as the deterioration process. However, many systems subject to maintenance and degradation may involve state transitions, which depend explicitly on time, or occur discretely. For these reasons, maintenance actions cannot generally be modeled by a simple exponential distribution within the Markov process. For example, a significant repair time, which may reflect realistic maintenance activities, does not generally follow the exponential distribution. Therefore, we have to develop a more accurate methodology to allow the Markov processes to model significant holding times. The concept of a phase-type distribution [21] can be used to approximate a time delay until absorption to one of the states in the Markov chain. It is also known that the Erlang process (i.e., summation of identical exponential distributions as displayed in Fig. 3) minimizes the variance among any phase-type distributions [7]. In other words, non-exponential holding time distributions can be approximated by inserting multiple intermediate states between any two conventional degradation states. In Fig. 3, τ is the total transition interval between S_0 and S_m, and the total number of intermediate stages used to approximate it, is equal to $m-1$. The rate at which transitions happen is proportional to m to provide a same total transition time. This Erlang process approximation of a constant time delay in a Markov process enables the incorporation of various maintenance activities into the equipment deterioration model.

Fig. 3: Markov chain for Erlang process

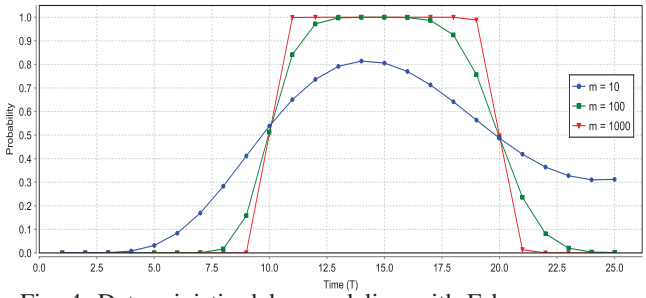

Fig. 4: Deterministic delay modeling with Erlang process

Fig. 4 illustrates the results of implementing the Markov chain of Fig. 3 to approximate a constant delay of 10 hours [22]. Fig. 4 shows the probability distribution of delay times for different values of *m*. This is how we implement constant time delays. This is useful for modeling repairs or scheduled maintenance, while preserving the Markov property. There is a clear and obvious trade-off here between the accuracy (how close it is to modelling a deterministic time delay) and the resulting expansion in the size of the model.

B. System modeling

A simplified model of a satellite system taken in [5] is shown in Fig. 2. A description of that model using the PRISM language was developed using two approaches: our approach and the approach in [5]. Initially, the system starts from its normal mode. During its operation, the satellite can encounter both, a scheduled/unscheduled interruption, or a catastrophic

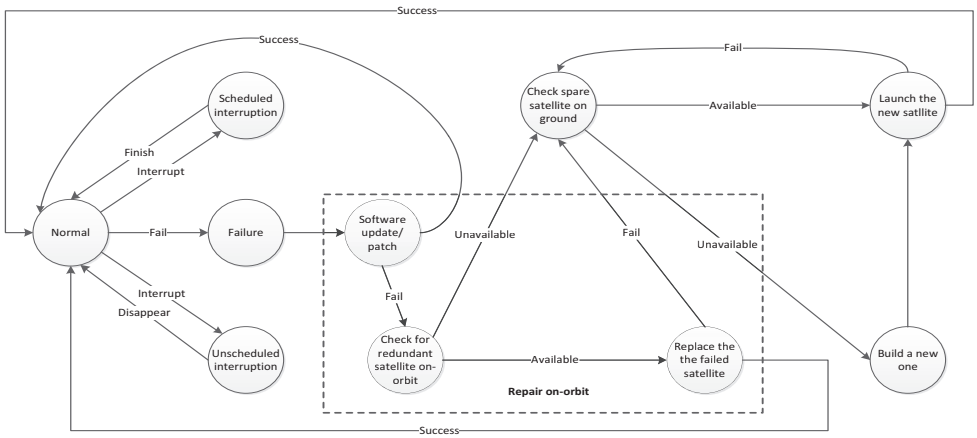

Fig. 2: Satellite system model

TABLE I: Parameters for the PRISM model of satellite system

R_e	m	MTBF	MTTR	t_a	p_b	t_y	t_d	p_n	t_k
0.80	200	15 yr	24h	4320h	80%	24h	1440h	90%	24h

failure leading to a permanent failure. An unscheduled interruption can be caused by unpredictable instances such as by cosmic radiation induced Single Event Upsets (SEUs). The scheduled interruption are routine procedures necessary for regular satellite operations and maintenance. This includes orbit correction maneuvers, hardware and software updates, and payload maintenance. Both of these scheduled and unscheduled interruptions may cause non-negligible downtime, however these interruptions are temporary in nature. On the other hand, a permanent failure in the solar cells, thermal protection system or even a collision with small orbital debris may cause the satellite a permanent failure during its lifetime, or the satellite can simply reach due to its End-Of-Life (EOL). The *unscheduled interruptions* are modeled using exponential distribution, e.g. the transition delay between the state *normal* and state *unscheduled interruptions* is exponentially distributed, whereas the *scheduled interruptions* are modeled using the Erlang distribution.

When a satellite encounters a permanent failure, the ground station analyze the failure and decides if the failure is repairable by a software patch or requires replacement by a redundant satellite on orbit. If a redundant satellite is not available, then a new satellite needs to be manufactured and launched. If the satellite is launched safely, the system is back to normal state. In the worst case, the launch can also fail with a certain probability which is not uncommon in space industry.

Note that *reward* operator *R* from PRISM property specification language and R_e that was also called *R* in [5] should not be confused. The permanent satellite failure is assumed to follow a constant failure rate λ, hence follows an exponential distribution. The reliability R_e, failure rate λ and the Mean Time Between Failure (MTBF) can be related with the following equation [5]:

$$\lambda = \frac{-lnR_e}{MTBF} \tag{1}$$

The parameters that we used to populate our model and analysis are summarized in Table I. These parameter values

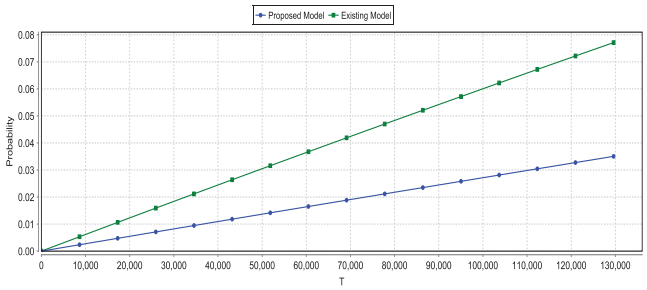

Fig. 5: Reliability (property 1)

are adopted from [5] to facilitate the comparison of their modeling approach to ours. These parameters are inspired by the US GPS Block III satellites or similar. The reliability of the satellite is denoted by R_e. The time to scheduled interruption t_{si} and unscheduled interruption t_{ui}, are both set to t_a, where $t_a = t_{si} = t_{ui}$. When a satellite fails, the probability that the satellite is replaced by an redundant satellite on orbit is p_b. If that is not possible, then the decision time to make a new satellite and the manufacturing time is represented by t_y and t_d respectively. The probability of successful launch is expressed by p_n. In the case of a successful launch, the time required for orbit correction and for contacting the ground is expressed by t_k. The parameters that require discrete time delays, such as Mean Time To Repair (MTTR), t_a, t_y, t_d and t_k, are modeled using Erlang distribution with the value m = 200.

V. Quantitative Analysis using PRISM

The PRISM tool is able to evaluate a wide range of dependability properties. We are interested to evaluate the reliability, availability, and maintainability properties of a satellite system. For every satellite mission, these evaluations need to meet a certain quantitative target to ensure that the satellite system is able to meet the mission requirements. Using formal analysis, it is possible to ensure some target availability, reliability and maintainability at early design stage. Such analysis at early design stage may reduce the overall cost, effort and risk of the project. All the RAM related properties that are evaluated with our model, are summarized in Table II.

In property 1, for reliability evaluation, we calculated the probability that the satellite will fail in such a way that repair

TABLE II: RAM Properties

Type	Properties	Figure
Reliability	1. Probability that a satellite will need to be replaced by a redundant one in 15 years over the time: P = ? [F <= T s = "check for redundant satellite on-orbit"]; R_e = 0.80; T = 0 : 129600 : 8640	Fig. 5
	2. Number of times a satellite will need to be replaced by a redundant one over different radiabilities in 15 years: R {"num_replace"} =?[C <= T]; T = 129600; R_e = 0.01 : 0.99 : 0.05	Fig. 6
Maintainability	3. Number of times that satellites need to be repaired over different radiabilities in 15 years: R {"num_repair"}[C <= T]; T = 129600; R_e = 0.01 : 0.99 : 0.05	Fig. 7
	4. Number of times that satellites need to be repaired over different MTBFs from 1 to 15 years: R {"num_repair"}[C <= T]; R_e = 0.80; MTBF = 1 : 129600 : 8640	Fig. 8
Availability	5. Availability of the satellite in 15 years, over different reliability: (R {"available"}=?[C <= T])/T ; T = 129600; R_e = 0.01 : 0.99 : 0.05	Fig. 9
	6. Relationship between satellite availability and its maintenance time (f) taken for scheduled interruption: (R {"available"}=?[C <= T])/T ; T = 129600; R_e = 0.80, f = 1 :48 : 3	Fig. 10

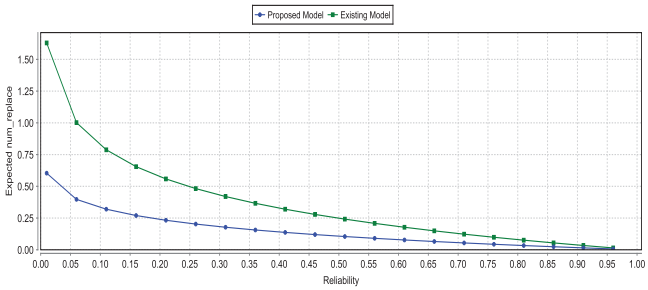

Fig. 6: Reliability (property 2)

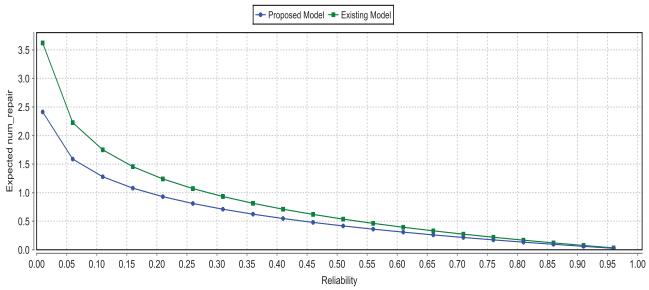

Fig. 7: Maintainability (property 3)

on-orbit would not be possible, hence will need to be replaced by a redundant one. We evaluate this property for different mission times (T) from 1 to 15 years. The obtained results are shown in Fig. 5. We observe a significant difference between the results obtained from our approach and the approach in [5]. The gap between reliability results from both modeling approaches gets wider and the trend continues. It gives us a hint that for longer mission times, the results from these two approaches will vary, i.e. the larger the mission time, the larger is the difference. For instance, according to our model, the probability that the satellite will fail in 15 years is 0.03. In contrast, according to the approach in [5], the probability is 0.07. This huge gap is due to the fact that the repair on-orbit and the repair via software patches was not directly modeled in the approach in [5]. In our case, we have included those in our model and also instead of using an exponential distribution, we have approximated the discrete repair transitions using the Erlang distribution.

Property 2 evaluates, the number of times the satellite will need to be replaced with a redundant one in 15 years for different values of R_e in (1). We vary the value of R_e from 0.01 to 0.99, which is the same range explored in [5], so that we can compare the results easily. This statement is applicable for all the modeling results we present in this paper. After evaluating property-2, the results are plotted in Fig. 6. We observe that there is a wide gap between the results calculated using the two models. If R_e = 0.01, according to [5], the number of times it will need to be replaced is 1.63, whereas with our modeling, the number is around 0.60. The gap is wider when the R_e value in (1) is smaller, and as the value of R_e increases, results from both approaches get closer leading to the same asymptotic behavior.

To analyze maintainability properties, the number of times that the satellite will need to be repaired in 15 years for different values of R_e is evaluated via property 3 and shown in Fig. 7. For property 4, we vary the MTBF values ranging

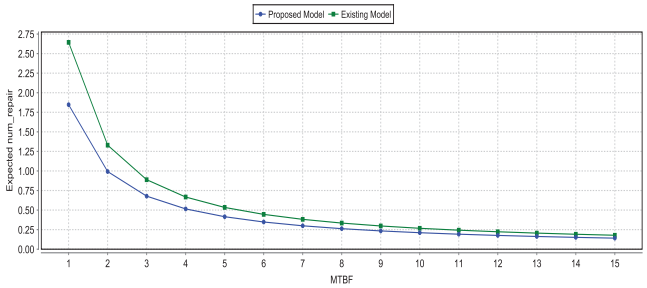

Fig. 8: Maintainability (property 4)

from 1 to 15 years to obtain the number of repair and show the results in Fig. 8. Fig. 7 shows a similar trend as reliability property 2, the difference gets wider if the parameter R_e in (1) is small. If parameter R_e is increased, then the number of satellite repairs is decreased. In Fig. 8, when the MTBF is between 10-15 years, the number of repairs required is fairly close, however a convergence was not observed up to this point. If the MTBF value is smaller than that, the difference gets wider.

Another set of interesting results is found when we analyze the availability. The availability properties are evaluated and

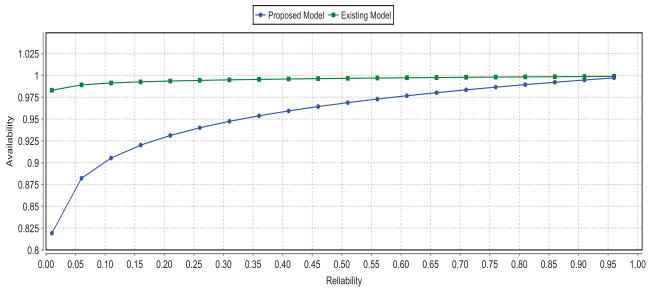

Fig. 9: Availability (property 5)

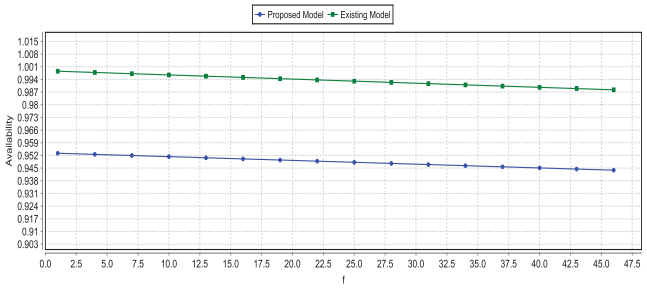

Fig. 10: Availability (property 6)

plotted in Fig. 9 and in Fig. 10. The comparison of availability reveals a significant difference between the two modeling approaches for some cases. In Fig. 9, there is a large difference between the two approaches, and this difference decreases, but it remains noticeable throughout the explored modeling interval. For instance, if R_e = 0.50, the availability of the satellite system is 0.998 according to the existing approach. By contrast, according to our modeling method, the availability is only 0.961. The difference gets quite wide for lower values of R_e. For instance, the availability is only 90% for R_e = 0.10 according to our modeling. In contrast, according to the other approach, the availability is 99%. The availability comparison of both approaches for different maintenance times due to scheduled interruption (1 to 48 hours), is plotted in Fig. 10. Here as well, we observe a large difference. For the maintenance time of 48 hours assuming scheduled interruption, the availability never goes below 99% in the first 15 years according to the traditional approach [5]. According to our modeling approach, the availability value is $\approx$ 95%, such a large difference in availability can be crucial for safety-critical and mission-critical applications. Our modeling shows that to maintain a 95% availability, the maintenance time must be less than 7 hours for the parameters mentioned in Table I. During the satellite design phase, such a crucial error can cause a serious problem in the real mission. For example, most of the communication satellites targets an availability of five nines, e.g. 99.999%. Even a single missing 'nine' will not fulfill the mission requirement and may cause serious damage related to financial and performance issues.

VI. Conclusion and Future Works

In this paper, we showed how probabilistic model checking can be used to accurately model satellite systems and perform RAM analysis on them. The authors in [5] claimed, to be the first one to use probabilistic model checking to perform RAM analysis of satellite systems. However, their modeling approach was limited to exponential distribution for both maintenance and failure transitions. By contrast, in our work, we show the use of Erlang distribution to model the discrete time delays for the maintenance related transitions and compare the modeling results. Approximation of discrete delay makes our modeling approach more accurate and closer to the real world scenario. As future work we propose modeling of a satellite constellation instead of a single satellite system using our approach.

References

[1] E. M. Clarke, O. Grumberg, and D. Peled, *Model checking*. MIT press, 1999.

[2] E. M. Clarke, E. A. Emerson, and A. P. Sistla, "Automatic verification of finite-state concurrent systems using temporal logic specifications," *ACM Transactions on Programming Languages and Systems*, vol. 8, pp. 244–263, 1986.

[3] N. Rungta, G. Brat, W. J. Clancey, C. Linde, F. Raimondi, C. Seah, and M. Shafto, "Aviation safety: modeling and analyzing complex interactions between humans and automated systems," in *Proceedings of the 3rd International Conference on Application and Theory of Automation in Command and Control Systems*. ACM, 2013, pp. 27–37.

[4] K. A. Hoque, O. Ait Mohamed, Y. Savaria, and C. Thibeault, "Early analysis of soft error effects for aerospace applications using probabilistic model checking," in *Formal Techniques for Safety-Critical Systems*, ser. Communications in Computer and Information Science. Springer International Publishing, 2014, vol. 419, pp. 54–70.

[5] Z. Peng, Y. Lu, A. Miller, C. Johnson, and T. Zhao, "A probabilistic model checking approach to analysing reliability, availability, and maintainability of a single satellite system," in *Modelling Symposium (EMS), 2013 European*, Nov 2013, pp. 611–616.

[6] M. Kwiatkowska, G. Norman, and D. Parker, "PRISM 4.0: verification of probabilistic real-time systems," in *Computer aided verification*. Springer, 2011, pp. 585–591.

[7] A. David and S. Larry, "The least variable phase type distribution is erlang," *Stochastic Models*, vol. 3, no. 3, pp. 467–473, 1987.

[8] M. Bozzano, A. Cimatti, J.-P. Katoen, V. Y. Nguyen, T. Noll, and M. Roveri, "Safety, dependability and performance analysis of extended aadl models," *Comput. J.*, vol. 54, no. 5, pp. 754–775, May 2011.

[9] P. M. Fischer, D. Lüdtke, V. Schaus, O. Maibaum, and A. Gerndt, "Formal verification in early mission planning."

[10] Z. Shen, "Model checking for the mpl entry and descent sequence," *Iowa State University, Tech. Rep*, 2001.

[11] D. L. Dill, "The mur ϕ verification system," in *Computer Aided Verification*. Springer, 1996, pp. 390–393.

[12] P. J. Pingree, E. Mikk, G. Holzmann, M. Smith, and D. Dams, "Validation of mission critical software design and implementation using model checking [spacecraft]," in *Digital Avionics Systems Conference, 2002. Proceedings. The 21st*, vol. 1. IEEE, 2002, pp. 6A4–1.

[13] P. R. Gluck and G. J. Holzmann, "Using spin model checking for flight software verification," in *Aerospace Conference Proceedings, 2002. IEEE*, vol. 1. IEEE, 2002, pp. 1–105.

[14] X. Gan, J. Dubrovin, and K. Heljanko, "A symbolic model checking approach to verifying satellite onboard software," *Science of Computer Programming*, vol. 82, pp. 44–55, 2014.

[15] A. Cimatti, E. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, and A. Tacchella, "Nusmv 2: An opensource tool for symbolic model checking," in *Computer Aided Verification*. Springer, 2002, pp. 359–364.

[16] C. Baier, J.-P. Katoen, and H. Hermanns, "Approximate symbolic model checking of continuous-time markov chains (extended abstract)," 1999.

[17] M. Kwiatkowska, G. Norman, and D. Parker, "Controller dependability analysis by probabilistic model checking," *Control Engineering Practice*, vol. 15, no. 11, pp. 1427–1434, 2007.

[18] ——, "PRISM: Probabilistic model checking for performance and reliability analysis," *SIGMETRICS Performance Evaluation Review*, vol. 36, no. 4, pp. 40–45, 2009.

[19] H. Hansson and B. Jonsson, "A logic for reasoning about time and reliability," *Formal Aspects of Computing*, vol. 6, pp. 512–535, 1994.

[20] B. Haverkort, L. Cloth, H. Hermanns, J.-P. Katoen, and C. Baier, "Model checking performability properties," in *DSN 2002. Proceedings. International Conference on*. IEEE, 2002, pp. 103–112.

[21] T. Osogami and M. Harchol-Balter, "Closed form solutions for mapping general distributions to quasi-minimal ph distributions," *Performance Evaluation*, vol. 63, no. 6, pp. 524–552, 2006.

[22] K. A. Hoque, O. Mohamed, Y. Savaria, and C. Thibeault, "Probabilistic model checking based DAL analysis to optimize a combined TMR-blind-scrubbing mitigation technique for FPGA-based aerospace applications," in *Formal Methods and Models for Codesign (MEMOCODE), 2014 Twelfth ACM/IEEE International Conference on*, Oct 2014, pp. 175–184.