



Peng, Z., Lu, Y., Miller, A., Johnson, C., and Zhao, T. (2013) *A probabilistic model checking approach to analysing reliability, availability, and maintainability of a single satellite system*. In: 7th European Symposium on Computer Modelling and Simulation (EMS2013), 20-22 Nov 2013, Manchester, UK.

Copyright © 2013 IEEE

A copy can be downloaded for personal non-commercial research or study, without prior permission or charge

Content must not be changed in any way or reproduced in any format or medium without the formal permission of the copyright holder(s)

<http://eprints.gla.ac.uk/89809/>

Deposited on: 16 January 2014

A Probabilistic Model Checking Approach to Analysing Reliability, Availability, and Maintainability of a Single Satellite System

Zhaoguang Peng^{1,2}, Yu Lu^{2,*}, Alice Miller², Chris Johnson² and Tingdi Zhao¹

¹ *School of Reliability and Systems Engineering, Beijing University of Aeronautics and Astronautics, Beijing, China*

² *School of Computing Science, University of Glasgow, Glasgow, United Kingdom*

Abstract—Satellites now form a core component for space based systems such as GPS and GLONAS which provide location and timing information for a variety of uses. Such satellites are designed to operate in-orbit and have lifetimes of 10 years or more. Reliability, availability and maintainability (RAM) analysis of these systems has been indispensable in the design phase of satellites in order to achieve minimum failures or to increase mean time between failures (MTBF) and thus to plan maintainability strategies, optimise reliability and maximise availability. In this paper, we present formal modelling of a single satellite and logical specification of its reliability, availability and maintainability properties. The probabilistic model checker PRISM has been used to perform automated quantitative analyses of these properties.

Keywords—satellite systems; reliability, availability and maintainability (RAM) analysis; probabilistic model checking; continuous time Markov chains (CTMCs)

I. INTRODUCTION

With the emergence of efficient, high-performance, and low cost satellites, earth orbiting satellites are often deployed in satellite constellations and space systems to ensure reliable and dependable missions. These kinds of satellites have played an essential part in both civil and military contexts, and support a wide range of applications ranging from satellite navigation to space stations. Reliability, availability and maintainability (RAM) analysis has been indispensable in the design phase of satellites in order to achieve minimum failures or to increase mean time between failures (MTBF) and thus to plan maintainability strategies, optimise reliability and maximise availability. The question of how to select optimal configurations and maintenance plans and underlying resources, to satisfy requirements and improve efficiency is a key research question.

This concern calls for effective solutions to the challenges of verifying large and complex satellite systems. Formal verification is a well-established technique in Computer Science for either detecting errors, or for providing increased confidence in the reliability of a system. Until now, attempts to verifying satellite systems has been piecemeal. Verification largely depends on more brute force approaches, such as simulation and testing. Generally, simulation is the common

validation approach used for verification of such systems and protocols applied in them. However, simulation has been unable to keep pace with the growth in satellite design complexity. It is therefore timely to apply formal verification techniques to this domain.

Model checking is a formal verification technique that involves defining a model of a system from a formal specification. The model is then used to check desired properties of the system. This involves exploring the underlying state space of the model, and specifying properties via some formal logic such as temporal logic. In this context, the effects of proposed changes to an in-orbit system can be first checked via a model, rather than via expensive prototypes. The required reliability, availability, and maintainability properties of satellite systems can be expressed in temporal logic, and so lend themselves very well to proof via model checking.

The goal of the paper is to adopt probabilistic model checking to cope with the verification demand introduced by satellite systems. Probabilistic model checking is a formal method for specifying quantitative properties of a system model. Models obtained by this technique are normally extensions or variants of Markov chains or automata, extended with costs and rewards that estimate resources and their usage during operation. Properties to be verified or analysed are specified in temporal logic with auxiliary operators such as probability and reward. We present an automated quantitative analysis of single satellite availability with the probabilistic model checker PRISM [1].

Our paper is organised as follows. In Section II we give technical background on probabilistic model checking, while in Section III we present our formal specification of a single satellite and its associated continuous-time Markov chain (CTMC) model. Then, we perform RAM analysis using PRISM in Section IV. In Section V we report related work. Finally, in Section VI we conclude and outline directions for future research.

II. PROBABILISTIC MODEL CHECKING

In this section we introduce some formal notations that are relevant to probabilistic model checking. Note that our definitions are from [2], from which further details can be found.

*Corresponding author. (✉) School of Computing Science, University of Glasgow, 17 Lilybank Gardens, Glasgow G12 8RZ, United Kingdom. (✉) y.lu.3@research.gla.ac.uk.

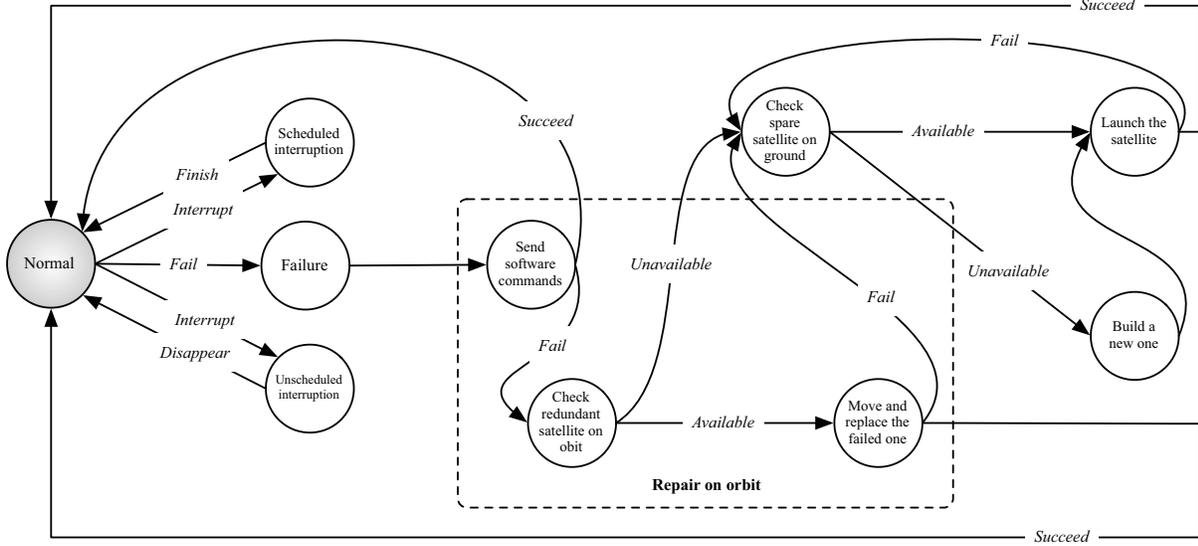


Figure 1. A failure model and maintainability plan of a single satellite

A. Continuous-time Markov Chains

Let AP be a fixed, finite set of atomic propositions. Formally, a continuous-time Markov chain (CTMC) \mathcal{C} is a tuple (S, s_{init}, R, L) where:

- $S = \{s_1, s_2, \dots, s_n\}$ is a finite set of states.
- $s_{init} \in S$ is the initial state.
- $R : S \times S \rightarrow \mathbb{R}_{\geq 0}$ is the transition rate matrix.
- $L : S \rightarrow 2^{AP}$ is a labelling function which assigns to each state $s_i \in S$ the set $L(s_i)$ of atomic propositions $a \in AP$ that are valid in s_i .

Intuitively, $R(s_i, s_j) > 0$ if and only if there is a transition from state s_i to state s_j . Furthermore, $R(s_i, s_j)$ specifies that the probability of moving from s_i to s_j within t time units is $1 - e^{-R(s_i, s_j) \cdot t}$, an exponential distribution with rate $R(s_i, s_j)$. If $R(s_i, s_j) > 0$ for more than one state s_j , a competition between the transitions originating in s_i exists, known as the race condition.

B. Continuous Stochastic Logic

Let $\mathcal{C} = (S, s_{init}, R, L)$ be a continuous time Markov chain. In this section, we introduce Continuous Stochastic Logic (CSL) [3], [4]. CSL is inspired by the logic Computation Tree Logic (CTL) [5], and its extensions to discrete time stochastic systems (PCTL) [6], and continuous time non-stochastic systems (TCTL) [7]. There are two types of formulae in CSL: state formulae, which are true or false in a specific state, and path formulae, which are true or false along a specific path.

Let $a \in AP$ be an atomic proposition, $p \in [0, 1]$ be a real number, $\bowtie \in \{\leq, <, >, \geq\}$ be a comparison operator, and $I \subseteq \mathbb{R}_{\geq 0}$ be a non-empty interval. The syntax of CSL

formulae 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 Φ and Ψ are state formulas, then so are $\neg\Phi$ and $\Phi \wedge \Psi$.
- If Φ is state formula, then so is $\mathcal{S}_{\bowtie p}(\Phi)$.
- If φ is a path formula, then $\mathcal{P}_{\bowtie p}(\varphi)$.
- If Φ and Ψ are state formulas, then $\mathcal{X}_I\Phi$ and $\mathcal{U}_I\Psi$ are path formulas.

$\mathcal{S}_{\bowtie p}(\Phi)$ asserts that the steady-state probability for a Φ state meets the boundary condition $\bowtie p$. $\mathcal{P}_{\bowtie p}(\varphi)$ asserts that the probability measure of the paths satisfying φ meets the bound given by $\bowtie p$. The path formula $\mathcal{X}_I\Phi$ asserts that a transition is made to a Φ state at some time point $t \in I$. Operator \mathcal{U}_I is the timed variant of the until operator of CTL; the path formula $\Phi\mathcal{U}_I\Psi$ asserts that Ψ is satisfied at some time instant in the interval I and that at all preceding time instants Φ holds.

III. FORMAL MODELLING WITH A CTMC

PRISM [1] is a probabilistic model checker. It supports the analysis of several types of probabilistic models: discrete-time Markov chains (DTMCs), continuous-time Markov chains (CTMCs), Markov decision processes (MDPs), probabilistic automata (PAs), and also probabilistic timed automata (PTAs), with optional extensions of costs and rewards. PRISM allows us to verify properties specified in the temporal logics PCTL for DTMCs and MDPs and CSL for CTMCs. Models are described using the PRISM language, a simple, state-based language. The abstract model

of a single satellite is illustrated in Figure 1, parameters are omitted. We take a CTMC as our underlying PRISM model for our abstract model. The detailed PRISM model of the satellite system, the property specification and the analysis results are available in [8].

We specify our actual CTMC model with states, a transition rate matrix, and a labelling function. Initially, the satellite runs in the normal state. After a period of execution it could be interrupted by an scheduled or unscheduled interruption during its lifecycle. Scheduled interruptions are normally caused by certain types of Operations and Maintenance (O&M) for routine satellite. This can cause satellite signal unavailability due to the station keeps manoeuvres, atomic clock maintenance, software updates, and hardware maintenance. Unscheduled interruptions can be caused by solar radiation, the earth’s magnetic field cosmic rays, which result in a satellite Single Event Upset (SEU). However, both scheduled and unscheduled interruptions are usually temporary, lasting just several hours. An unscheduled interruption usually disappears automatically. The satellite can fail any time during its lifetime due to End-of-Life (EOL) outage or other vital failures.

When the satellite fails, staff on the ground must decide upon the best approach to repair it. It may be possible that failures can be resolved on orbit by giving specific software commands to the satellite. Otherwise it might be necessary to move a redundant satellite into position to replace the failed satellite. If no redundant satellite is available then a new satellite must be manufactured and launched. In the worst case, the new satellite does not launch successfully, due to a known probability of satellite launch failure.

In our paper, parameter values correspond to those latest U.S. GPS system, GPS Block III satellites. The GPS III series is the newest block of GPS satellites (SVN-74 and up). GPS III provides more powerful signals than previous versions in addition to enhanced signal reliability, accuracy, and integrity. The key improvement is the 15-year design lifespan [9]. Since not all of the actual data for the GPS III is available, in this paper we instead use some parameter values associated with similar satellite systems. All parameters used in our CTMC model and properties are specified in Table I.

Table I
PARAMETERS FOR THE CTMC MODEL AND ANALYSES

| R | $MTBF$ | $MTTR$ | t_α | p_β | t_γ | t_δ | t_ϵ | p_η | t_κ |
|------|--------|--------|------------|-----------|------------|------------|--------------|----------|------------|
| 0.80 | 15y | 24h | 4320h | 80% | 24h | 1440h | 4320h | 90% | 24h |

We use p to express probability and t for time, and the reliability of the satellite is R . If the satellite fails, we say that it moves from a “normal” state to a “failure” state. Both the mean time to unscheduled interruption and the mean time to the scheduled interruption are t_α . When the satellite fails, the probability of the failure being resolved in-orbit by

moving a redundant satellite to replace the failed one is p_β . If on orbit repair is not possible, a new satellite is needed. The times taken to decide to build a new satellite and for one to be manufactured are t_γ and t_δ respectively. If a new satellite is to be manufactured, the probability of successful launch is p_η . After successful launch, the time taken for the satellite to move to the right position and a normal signal sent from it to be received on ground is t_κ .

IV. QUANTITATIVE ANALYSIS IN THE PRISM

We have identified the need to analyse reliability, availability, and maintainability properties of satellite based applications. We illustrate the use of probabilistic model checking in this domain by describing our PRISM model. The reliability for a satellite consists of scheduled interruptions, unscheduled interruptions, and failure states in the system. The probability of successful launch is the reliability for the satellite. “Repaired in-orbit” is the maintainability for the satellite. Reliability and maintainability are availability properties of a satellite. Reliability must be sufficient to support the mission capability needed in its expected operating environment.

If reliability and maintainability are not adequately designed into satellite and space based systems, there is risk that design will breach desired availability or performance requirements. System performance baseline thresholds with significantly higher design or development costs due to resulting corrective action costs; will cost more than anticipated to use and operate; or will fail to provide availability expected by the researchers or users.

Satellite will deteriorate with time due to failure mechanisms. We assume that time delay is a random variable selected from an exponential distribution, which is an assumption used in PRISM. According to the system reliability theory [10], the reliability of a satellite from $R(t)$ can be defined as

$$R(t) = Pr\{T > t\} = e^{-\lambda t}, \quad (1)$$

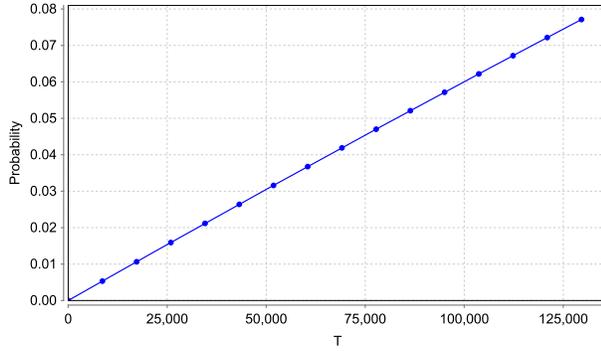
and, then we can obtain

$$\lambda(t) = \frac{-\ln R(t)}{E(s_i)}. \quad (2)$$

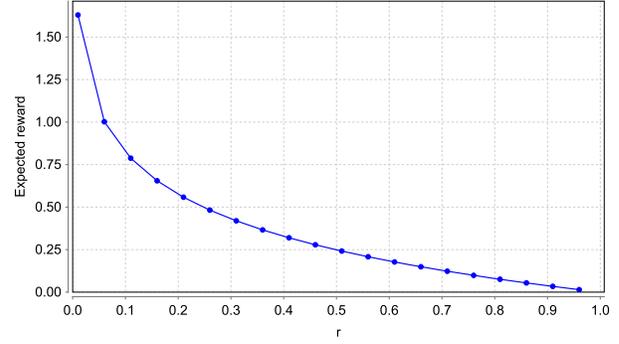
Satellite failures typically occur at some constant failure rate λ , failure probability depends on the rate λ and the exposure time t . Typically failure rates are carefully derived from substantiated historical data such as mean time between failure ($MTBF$). We have

$$\lambda = \frac{-\ln R}{T} \implies \lambda = \frac{-\ln R}{MTBF}, \quad (3)$$

where $t = T = MTBF$, where $MTBF$ is the design parameter or the statistics parameter. Referring to the latest characteristics of satellites used for Global Positioning



(a) Reliability property 2



(b) Reliability property 4

Figure 2. Analysis results of reliability properties.

Systems (GPSs), we assume the *MTBF* of the satellite to be 15 years. As a result, $R = 0.80$ and $MTBF = 15 \text{ years}$. Further, the mean time to repair (*MTTR*) is 24 hours.

PRISM provides support for automated analysis of a wide range of quantitative properties of these models, such as “what is the probability of a failure causing the satellite to stop working within 12 hours?”, “what is the worst-case probability of the satellite on-board system terminating due to an error, over all possible initial configurations?”, or “what is the worst-case expected time taken for the satellite signal to be received?”.

A. Reliability Properties and Analysis

Reliability properties that we can analyse using PRISM include:

- 1) the probability that a satellite will need to be replaced by a new one in 15 years at the reliability 0.80:
 $P_{=?}[F \leq T \ s = 5]; T = 129600;$
- 2) the probability that a satellite will need to be replaced by a new one due to complete failure in 15 years at the reliability 0.80 over the time:
 $P_{=?}[F \leq T \ s = 5]; R = 0.80; T = 0 : 129600 : 8640;$
- 3) how many times a satellite will need to be replaced by a new one in 15 years at the reliability 0.80:
 $R_{=?}[C \leq T]; T = 129600; R = 0.80;$
- 4) how many times a satellite will need to be replaced by a new one over different reliabilities, in 15 years:
 $R_{=?}[C \leq T]; T = 129600; r = 0.01 : 0.99 : 0.05.$

As is shown in Figure 2(a), the probability that the satellite has a failure and is unable to be repaired during 15 years is 7.71%. From the analysis result in Figure 2(b), the number of times the satellite will have a failure and be unable to be repaired in 15 years is 0.08, under the precondition that the reliability is 0.80. If the reliability is set to 0.5, the number of vital failures will be smaller than 0.25 during 15 years. Using the property to calculate the number of unscheduled interruptions, the number of times will be 29.95 in 15 years.

B. Maintainability Properties and Analysis

Maintainability properties that we can analyse using PRISM include:

- 1) the number of times that satellites need to be repaired on the orbit in 15 years:
 $R_{=?}[C \leq T]; T = 129600; R = 0.80;$
- 2) the satellite maintenance times when the reliability from the 0.01 to 0.99 in 15 years:
 $R_{=?}[C \leq T]; T = 129600; R = 0.01 : 0.99 : 0.01;$
- 3) the satellite maintenance times when the *MTBF* from the 1st year to 15th years:
 $R_{=?}[C \leq T]; T = 129600; R = 0.01 : 0.99 : 0.01; MTBF = 1 : 129600 : 8640;$
- 4) the number of cases that a satellite needs to be repaired on orbit, but not eventually succeed in 15 years:
 $R_{=?}[C \leq T]; T = 129600; r = 0.80.$

The number of times the satellite needs to be repaired on orbit in 15 years is 0.18. The number of times the satellite needs to be repaired on orbit over time is shown in Figure 3(a). When the reliability of the satellite is increased to 0.5, the number of times the satellite needs to be repaired will decrease to 0.5. Figure 3(b) illustrates that the number of times to repair the satellite is below 1 when the *MTBF* is 2 years.

C. Availability Properties and Analysis

Availability properties that we can analyse using PRISM includes:

- 1) the availability of the satellite in 15 years, when the reliability is 0.80:
 $(R_{=?}[C \leq T])/T; T = 129600; R = 0.80,$ and
 $R_{=?}[C \leq T]; T = 129600; R = 0.01 : 0.99 : 0.01;$
- 2) the unavailability of a satellite over the satellite operation time:
 $(T - R_{=?}[C \leq T])/T; T = 0 : 129600 : 8640; R = 0.80;$
- 3) the relationship between satellite availability and its maintenance time taken for scheduled interruption:
 $(R_{=?}[C \leq T])/T; T = 129600; R = 0.80, f = 1 : 48 : 3.$

The availability of the satellite is 99.83% in 15 years when the reliability is 0.80. As is shown in Figure 4(a), if the reliability increases to 0.4, the availability of the satellite reaches 0.995. So if the required probability of the available satellite is 0.995, the reliability must have minimum value 0.4. Figure 4(b) presents the result of availability property

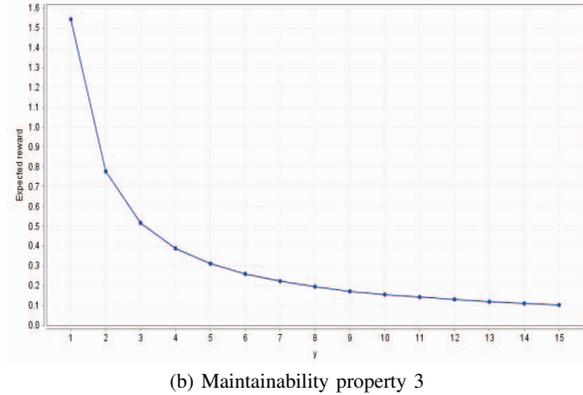
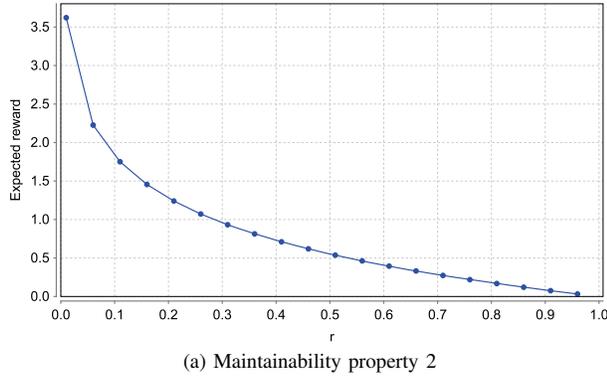


Figure 3. Analysis results of maintainability properties.

3). It shows that if the required availability is 0.995, the time taken for scheduled interruption for the satellite will be smaller than 16 hours.

V. RELATED WORK

There have been a number of notable attempts to use formal methods to address the problems of design exploration for a satellite system. The theorem prover PVS [11] was used to verify desired properties in system models of Ariane 5 where cost of failure is highest. The PICGAL project [12] has analysed ground-based software for launch vehicles similar to Ariane 5. In the NASA report [13], formal methods and their approaches to critical systems are explained to stakeholders from the aerospace domain. In [14], the potential role of formal methods in the analysis of software failures in space missions is discussed. Similarly, [15] explores how verification techniques, such as static analysis, model checking, and compositional verification, can be used to gain trust in model-based systems.

Model checking has been successfully applied to numerous computer systems and their applications, including both software and hardware systems [16], [17], [18], [19]. Historically, model checking has been considered to be a powerful extension of the traditional verification process such as emulation and simulation. It has also proved to be a suitable formal technique for exposing errors in satellites, mainly due to classical concurrency errors. Unforeseen interleavings between processes may cause undesired events to happen. In [20], the SPIN model checker [21] was used to formally analyse a multithreaded plan execution module. This module is a component of NASA's artificial intelligence-based spacecraft control system which launched in 1998 as part of the Deep Space 1 mission. Five previously undiscovered errors were identified in the spacecraft controller, in one case representing a major design flaw.

The model checking tool $Mur\psi$ [22] has been used in [23] to model the Entry, Descent and Landing phase of the Mars Polar Lander. The model checker was used to

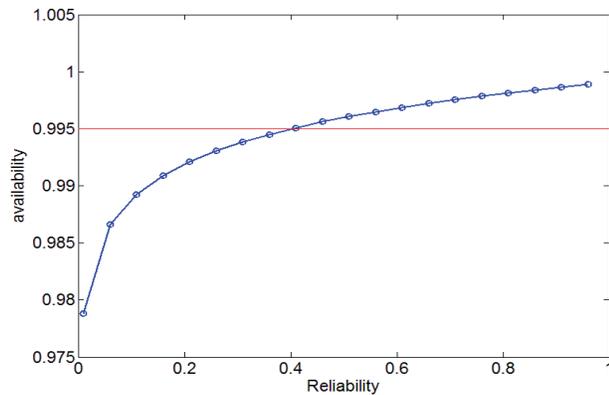
search for sequences of states that led to the violation of a $Mur\psi$ invariant. This stated that the thrust of the pulse-width modulation, which controls the thrust of the descent engines, should always be above a certain altitude. In [24] the model checker NuSMV [25] is used to model and verify the implementation of a mission and safety critical embedded satellite software control system. The control system is responsible for maintaining the attitude of the satellite and for performing fault detection, isolation, and recovery decisions, at a detailed level.

VI. CONCLUSION AND FUTURE WORK

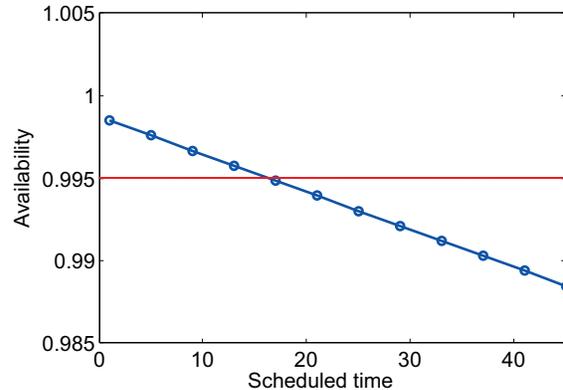
In this paper, a range of model checkers has been employed to represent and reason about design and failure of space based systems. The traditional approaches are not suitable for analysing system reliability and performance. In this paper, we consider an automated verification approach - probabilistic model checking, which extends classical model checking with quantitative analysis support.

Since parameter settings of our formal models are based on GPS Block III which is newest generation of GPS systems, our analysis results can be compared to existing GPS statistical analysis. In [9], the availability of the GPS Block III is given as 99.9%. The availability we evaluate in this paper is close to the actual data. This indicates that our approach is feasible and efficient. To the best of our knowledge, we are the first to use probabilistic model checking to perform RAM analysis of satellite systems.

Actually, numerous failures are distributed differently other than exponential distributions. In particular, a number of failures of satellites have a Weibull distribution [10], which follows the conventional three-component bathtub curve which models a burn-in and wear-out phase for failure prediction. For future work, we will look at how to represent arbitrary distributions in probabilistic models, and to what extent such kind of distributions are able to be supported by probabilistic model checking approaches. Further, we plan to extend our work of analysing a single satellite to satellite



(a) Availability property 2



(b) Availability property 3

Figure 4. Analysis results of availability properties.

constellation consisting of multiple satellites.

REFERENCES

- [1] M. Kwiatkowska, G. Norman, and D. Parker, “PRISM: Probabilistic symbolic model checker,” in *Computer Performance Evaluation: Modelling Techniques and Tools*, ser. LNCS, T. Field, P. G. Harrison, J. Bradley, and U. Harder, Eds. Springer, 2002, vol. 2324, pp. 200–204.
- [2] C. Baier and J.-P. Katoen, *Principles of Model Checking*. MIT Press, 2008.
- [3] A. Aziz, K. Sanwal, V. Singhal, and R. Brayton, “Verifying continuous time markov chains,” in *Computer Aided Verification*, ser. LNCS, R. Alur and T. Henzinger, Eds. Springer, 1996, vol. 1102, pp. 269–276.
- [4] C. Baier, J.-P. Katoen, and H. Hermanns, “Approximative symbolic model checking of continuous-time markov chains,” in *CONCUR’99 Concurrency Theory*, ser. LNCS, J. C. Baeten and S. Mauw, Eds. Springer, 1999, vol. 1664, pp. 146–161.
- [5] E. A. Emerson, “Temporal and modal logic,” in *Handbook of Theoretical Computer Science*, J. van Leeuwen, Ed. Elsevier, 1990, pp. 996–1072.
- [6] H. Hansson and B. Jonsson, “A logic for reasoning about time and reliability,” *Formal Aspects of Computing*, vol. 6, no. 5, pp. 512–535, 1994.
- [7] R. Alur, C. Courcoubetis, and D. Dill, “Model-checking for real-time systems,” in *Proceedings of the 1990 Fifth Annual IEEE Symposium on Logic in Computer Science*, 1990, pp. 414–425.
- [8] http://dcs.gla.ac.uk/~yu/index_files/satellite.zip.
- [9] <http://www.gps.gov/systems/gps/space/III>.
- [10] A. Høyland and M. Rausand, *System Reliability Theory: Models and Statistical Methods*. Springer, 2009.
- [11] S. Owre, J. M. Rushby, and N. Shankar, “PVS: A prototype verification system,” in *Automated Deduction—CADE-11*, ser. LNCS, D. Kapur, Ed. Springer, 1992, vol. 607, pp. 748–752.
- [12] L. Devauchelle, P. G. Larsen, and H. Voss, “PICGAL: Practical use of formal specification to develop a complex critical system.pdf,” in *FME ’97: Industrial Applications and Strengthened Foundations of Formal Methods*, ser. LNCS, J. Fitzgerald, C. B. Jones, and P. Lucas, Eds. Springer, 1997, vol. 1313, pp. 221–236.
- [13] J. Rushby, “Formal methods and their role in the certification of critical systems,” in *Safety and Reliability of Software Based Systems*, R. Shaw, Ed. Springer, 1997, pp. 1–42.
- [14] C. W. Johnson, “The natural history of bugs: Using formal methods to analyse software related failures in space missions,” in *FM 2005: Formal Methods*, ser. LNCS, J. Fitzgerald, I. J. Hayes, and A. Tarlecki, Eds. Springer, 2005, vol. 3582, pp. 9–25.
- [15] G. Brat, E. Denney, D. Giannakopoulou, J. Frank, and A. Jonsson, “Verification of autonomous systems for space applications,” in *Proceedings of the 2006 IEEE Aerospace Conference*, 2006, pp. 1–10.
- [16] W. Chan, R. J. Anderson, P. Beame, S. Burns, F. Modugno, and D. Notkin, “Model checking large software specifications,” *IEEE Transactions on Software Engineering*, vol. 24, no. 7, pp. 498–520, 1998.
- [17] G. Lowe, “Breaking and fixing the needham-schroeder public-key protocol using fdr,” *Software Concepts and Tools*, vol. 17, no. 3, pp. 93–102, 1996.
- [18] E. M. Clarke, O. Grumberg, H. Hiraishi, S. Jha, D. E. Long, K. L. McMillan, and L. A. Ness, “Verification of the futurebus+ cache coherence protocol,” *Formal Methods in System Design*, vol. 6, no. 2, pp. 217–232, March 1995.
- [19] J. R. Burch, E. M. Clarke, D. E. Long, K. L. McMillan, and D. L. Dill, “Symbolic model checking for sequential circuit verification,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 13, no. 4, pp. 401–424, April 1994.
- [20] K. Havelund, M. Lowry, and J. Penix, “Formal analysis of a space-craft controller using spin,” *IEEE Transactions on Software Engineering*, vol. 27, no. 8, pp. 749–765, 2001.
- [21] G. J. Holzmann, *The SPIN model checker*. Addison-Wesley, 2004.
- [22] D. L. Dill, A. J. Drexler, A. J. Hu, and C. H. Yang, “Protocol verification as a hardware design aid,” in *Proceedings of the IEEE 1992 International Conference on Computer Design: VLSI in Computers and Processors*, 1992, pp. 522–525.
- [23] Z. Shen, “Model checking for the MPL entry and descent sequence,” Iowa State University, Tech. Rep., 2001.
- [24] X. Gan, J. Dubrovin, and K. Heljanko, “A symbolic model checking approach to verifying satellite onboard software,” *Science of Computer Programming*, 2013.
- [25] 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*, ser. LNCS, E. Brinksma and K. G. Larsen, Eds. Springer, 2002, vol. 2404, pp. 359–364.