Contents lists available at ScienceDirect

Reliability Engineering and System Safety

journal homepage: www.elsevier.com/locate/ress

How reliable is satellite navigation for aviation? Checking availability properties with probabilistic verification

Yu Lu [a], Zhaoguang Peng [b,c,*], Alice A. Miller [a], Tingdi Zhao [c], Christopher W. Johnson [a]

[a] School of Computing Science, University of Glasgow, Glasgow, United Kingdom
[b] China Ceprei Laboratory, Guangzhou, China
[c] School of Reliability and Systems Engineering, Beijing University of Aeronautics and Astronautics, Beijing, China

ARTICLE INFO

Article history:
Received 11 September 2014
Received in revised form
15 July 2015
Accepted 19 July 2015
Available online 29 July 2015

Keywords:
Satellite systems
Formal methods
Model checking
Reliability
Availability
Probabilistic verification

ABSTRACT

This paper highlights a promising application of the analysis technique of probabilistic verification. We prove that it is able and suitable to analyse GNSS based positioning in aviation sectors for aircraft guidance. In particular, the focus is a widely used formal method called probabilistic model checking, and its generalisation to the analysis of quantitative aspects of a specific civil flight. We construct a formal model of the GNSS based positioning system for this application in the probabilistic π-calculus, a process algebra which supports modelling of concurrency, uncertainty, and mobility. After that, we encode our model in language of the PRISM symbolic probabilistic model checker. We then formalise and analyse the logical properties that relate to the dependability of the underlying system to check the system reliability and availability. We demonstrate how model specification and verification techniques can be successfully applied to the reliability and availability analysis of our case study.

1. Introduction

Satellite positioning systems are used within the transport industries such as marine, rail, and aviation sectors extensively. For example, in aviation, a three-dimensional global navigation satellite system (GNSS) enables an aircraft to determine its position (latitude, longitude, and altitude) anywhere on or above the earth. Data transmitted from a navigation and communication satellite provides the user with the time, the precise orbital position of the satellite and the position of other satellites in the system. In the past, satellites were only deployed for military purposes. However nowadays they are used for a wide range of civil aviation applications, including navigation, communication, tracking, and flight management.

Our work has been inspired by a number of previous European Commission (EC) projects such as GADEROS, GRAIL, LOCASYS, and SATLOC. These projects have proved the feasibility of introducing GNSS in non-critical systems by means of theoretical studies and demonstrations. The current EC project EATS [1] proposes a novel positioning system based on different techniques that have proved useful from other industry viewpoints such as using information sources from GNSS, UMTS, and GSM. Furthermore, reliability, availability, maintainability, and safety (RAMS) analysis [2] is used to study the dependability properties of the technical solution in the critical applications, which aims to verify the proposed solution.

Availability requirements are identified as the most challenging obstacles towards GNSS aided positioning systems in [2]. Many approaches [3–6] can be used to analyse availability properties. Among them, simulation, analytical analysis, and quantitative analysis are popular and practical. Each approach has its advantages and disadvantages that we do not discuss in this paper. We consider probabilistic verification, a quantitative analysis technique based on Markov models. It is a formal verification technique for analysing and verifying quantitative properties of a system's design, such as time, stochastic behaviour or resources. It is therefore highly suitable for modelling characteristics of our underlying system.

The mobility of an aircraft and satellites is universally recognised as an essential parameter for analysing the availability of satellite navigation systems. Our first task is to specify the communication between the airplane and satellites and their combined mobility. The second task is rendering these two models independently, in order to study the availability of the system in terms of different mobility models without changing the communication models.

In an example illustrated in Fig. 1 (a), some cars are on the road, and each is connected by a unique wavelength to a single transmitter. The transmitters have fixed connections to a central control. On some events such as signal fading, a car may be switched to another

* Corresponding author.
E-mail addresses: y.lu.3@research.gla.ac.uk (Y. Lu), zgpeng@buaa.edu.cn (Z. Peng), alice.miller@glasgow.ac.uk (A.A. Miller), ztd@buaa.edu.cn (T. Zhao), christopher.johnson@glasgow.ac.uk (C.W. Johnson).

http://dx.doi.org/10.1016/j.ress.2015.07.020
0951-8320/© 2015 Elsevier Ltd. All rights reserved.

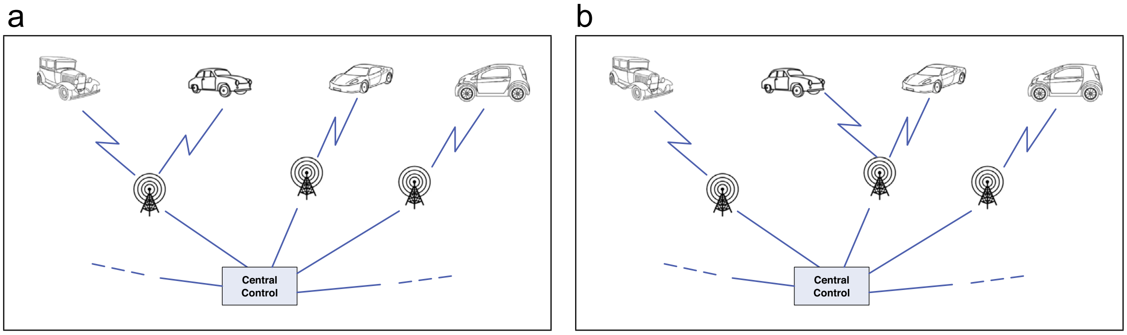


Fig. 1. Component mobility.

transmitter. We distinguish two types of movement: the physical movement of vehicles and virtual movement of communication links between vehicles and transmitters. The two types of movements are independent, but the physical movement of a vehicle may give rise to the virtual movement of its link to a transmitter (Fig. 1(b)).

In our study, we mainly deal with this kind of relationship between the movement of satellites and aircraft, and we study how the physical movement of both satellites and aircraft give rise to the mobility of links between them. As well as mobility, the π-calculus can be used to model parallel composition, alternative composition and sequential composition. Properties of the modelled system can be verified by studying the underlying labelled transition system. For this purpose, we specify the underlying models using the probabilistic π-calculus, an extension of the π-calculus [7,8] for modelling mobile systems.

Therefore, we first specify the communication between an aircraft and the associated satellites, taking into account their combined mobility. We then analyse the models of the aircraft and satellite set independently before the combined system. Note that behaviour of the system contains a high level of uncertainty (e.g., in signal transmission unreliability due to solar radiation, etc.). Since PRISM only model checks expressions in the reactive modules language, and this does not allow for component mobility, it is not currently possible to model check the underlying process algebraic models directly. In order to allow for automatic verification using PRISM, the underlying Markov Decision Processes (MDPs) semantic models of our specification are first constructed using rules presented in [9].

The basic idea is to first build a Markov models that captures the behaviour of the system, and then to use the model to analyse precisely specified properties using temporal logics. This analysis is automatically performed using the model checker PRISM [10], using a combination of a traversal of the state transition system of the model and numerical computation. A PRISM specification can be generated directly via a Markov chain variant described using the PRISM reactive modules language [11]. Alternatively, a high level model (using timed automata, or a process algebra, say) can be translated into the PRISM language. According to PRISM's manual, the latter approach can be more efficient than the former. This is due to the fact that PRISM is a symbolic model checker and the underlying data structures used to represent the system specification may function better when there is a high-level structure and regularity to exploit.

Our paper is organised as follows. In Section 2 we describe the underlying satellite navigation systems. In Section 3 the application of probabilistic verification is introduced. In Section 4 we present our formal specifications of a satellite navigation system for a specific navigation mission and their associated Markov decision processes respectively. Then, we verify availability properties using PRISM in Section 5. In Section 6 we discuss related work on analysing availability of satellite systems. Finally, in Section 7 we conclude.

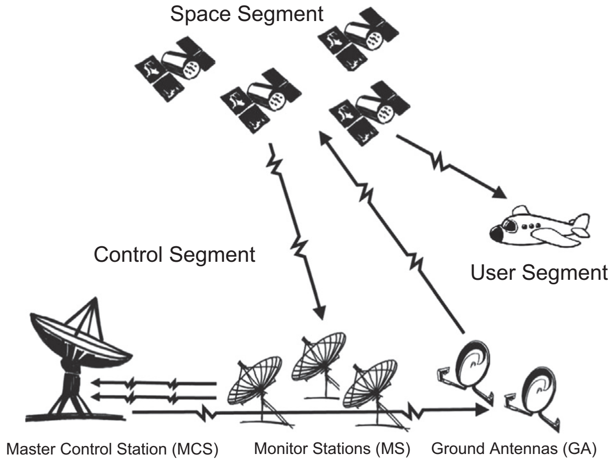


Fig. 2. Three segments of a GNSS system.

2. GNSS-based navigation systems

A GNSS-based navigation system consists of three major parts: the space segment, control segment and user segment. Recent theoretical research and standards have added a fourth environment segment to the satellite navigation system. The Galileo navigation system includes the environment segment in the composition of its navigation system. Although not explicitly mentioned, the environment segment is implied in the GPS system. To be conservative, the three traditional segments were used in this paper as the components of the study, and the environmental segment was treated as an influencing factor on the system. Failure of any subsystem will lead to errors in the final positioning. Fig. 2 is a schematic diagram of GNSS segments.

First, the monitor stations measure the pseudo-range of visible satellites every 6 s, correct them with ionospheric and meteorological data, smooth the measurement to generate data with a time interval of 15 s, perform smoothing again to generate data with a 15 min' time interval, and finally send the data to the master control station. The master control station is responsible for collecting and tracking data from each monitor station and calculating the satellite orbit and clock parameters using a Kalman estimator [12]. The results are transmitted to ground antennas and then to the satellite. Under the control of the master control station, the clock error, satellite ephemeris, navigation data, etc., are calculated and then transmitted to the corresponding satellite, and at the same time, the information is verified. The satellites transmit data associated with their current states to the users. The users need to use the position information provided by the satellites for positioning during navigation. According to [13], in general, at least four satellites are required to determine the user's position.

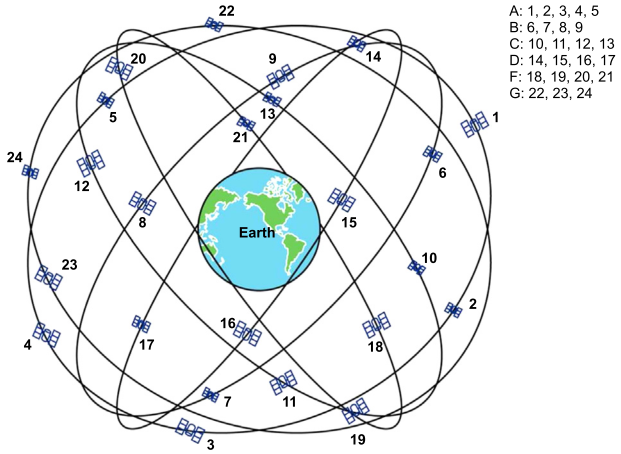


Fig. 3. A GNSS consisting of a constellation of 24 satellites.

The accuracy of the information that each subsystem provides is critical and depends directly on the navigation accuracy. From the monitor station to the master control station, from the master control station to the ground antenna, from the ground antenna to the satellite, and from the satellite to the user, the entire process is implemented by information transmission. Errors may exist in the process of information transmission, and if these errors are passed on all the way to the user, the position provided by the navigation system is unusable.

2.1. Space segment

The space segment of a standard GNSS is composed of a constellation of global navigation satellites, as shown in Fig. 3 (e.g., GPS comprises 24 satellites). The arrangement of the satellite constellation can guarantee that four or more satellites can be observed at the same time from any location at any time and ensure that the propagation of the satellite signal will not be disturbed by the environment. Therefore, a constellation navigation system should be a global and around-the-clock navigation system that continuously provides uninterrupted real-time navigation.

The functions of the space segment are described by the functional specification as follows: (1) continuously transmit uninterrupted navigation signals to users worldwide with carrier radio waves at a specific band, including the pseudo-range for satellite navigation, the exact time for users, the distance measurement and a navigation message that contains the spatial location and current health status of the satellite; (2) receive messages, ephemeris and other related information from the ground antenna via a specific band when the satellite passes above a ground antenna; (3) transmit and receive satellite commands from the master control station through the ground antenna, including activating redundant satellites, correcting on-orbit satellite errors and adjusting the spatial attitude of satellites at the appropriate time; (4) adjust the direction of the pair of solar panels on both sides of the navigation satellite according to the position of the sun to ensure a stable power supply.

The satellite transmits signals at a specific frequency, providing high-standard timing service to users worldwide. This function is implemented primarily by the atomic clock onboard the satellite.

2.2. Control segment

The control segment consists of three parts: monitor stations, master control stations (MCS) and ground antennas. This segment is implemented in the form of a number of detecting and measuring systems distributed across various locations in the world. The control segment continuously monitors and tracks the satellites. The roles of control segment components include: (1) monitor the satellite's operation and orbit states; (2) track and compute the orbit parameters of satellites and then send them to the satellites to be retransmitted to the users via a navigation message; (3) synchronise the clocks of satellites; (4) perform scheduling for satellites when necessary.

In the control segment of the satellite constellation, the monitor stations and ground antennas are unmanned, and the master control station is staffed. The unattended intelligent schema and information transmission among the advanced communication networks is implemented through coordination between computers and atomic clocks.

2.2.1. Master control station

The master control station acts as the brain of the control segment. It is responsible for processing the information received by the receiving station and feeding the correct information to the ground antenna. The main functions of the master control station are summarised as follows.

First, it provides satellites of the navigation system with the accurate time. The atomic clocks onboard the navigation satellites and the atomic clock of the monitor stations are synchronised by the master station, or, if a time difference between them is obtained, the master control station will include it in the navigation data and send it to the ground antenna. Second, it corrects the environmental parameters of the atmosphere, satellite ephemeris, satellite clock correction, etc., by calculation based on the data of satellites in the navigation constellation system that are monitored by various monitor stations and then transmits this information to the ground antennas to update the satellites. Third, it controls and sends commands to the satellites, and coordinates backup satellites to replace failed satellites once satellites under normal operation fail to receive and transmit. Finally, it controls satellites that deviate from their orbit and return them to their planned orbit.

2.2.2. Monitor stations

Monitor stations are centres that measure and examine the data under the control of the master control station. These stations consist of computers, high-precision atomic clocks, navigation receivers, and some environmental data detection sensors. The navigation receivers continuously monitor the status of the GNSS satellites, measure the on-orbit state of the satellites and collect environmental data to ensure the standards required for navigation accuracy.

The environmental sensors acquire data on the local environment, and the atomic clock ensures an accurate time. The computer processes these measurements and stores the data, then transmits the data to the master control station for calculating the satellite orbit. The control segment of the GPS is made up of five Monitor Stations located at Hawaii, Kwajalein, Ascension Island, Diego Garcia, and Colorado Springs.

2.2.3. Ground antennas

The functions of ground antennas under the control of the master control station are to receive clock errors, ephemeris, navigation data and other commands, which are all calculated and determined by the master control station; transmit this information to the navigation satellite system; and check the correctness of the transmitted information. There are also three Ground Antennas located at Ascension Island, Diego Garcia, and Kwajalein. Ground antennas consist primarily of a computer, a transmitter at a specific band, and a transmitting antenna.

2.3. User segment

The user segment of the navigation system consists of multiple parts, generally including system software, a navigation system

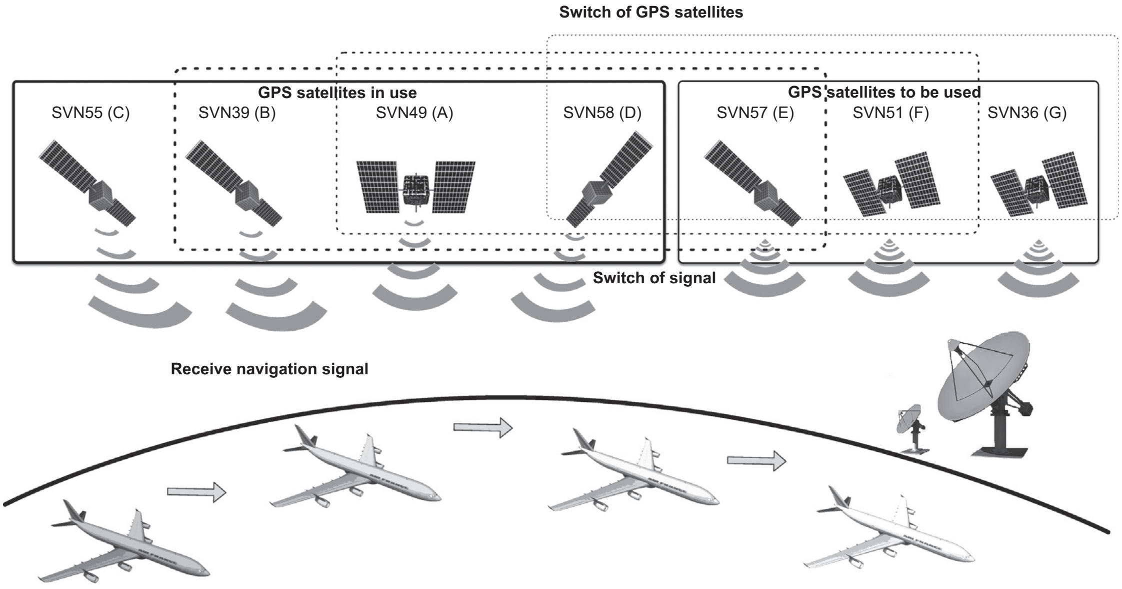


Fig. 4. GNSS (GPS) based navigation for an air line.

Table 1
Parameters of navigation satellites.

No	SVN	Launch date	Model	Life (years)	Reliability	Navigation interval	Duration (minutes)
A	49	24 March 2009	Block IIRM	10	0.8	12:00–14:29	149
B	39	26 January 1993	Block IIA	7.5	0.7	12:00–13:55	115
C	55	17 October 2007	Block IIRM	10	0.8	12:00–13:15	75
D	58	17 November 2006	Block IIRM	10	0.8	12:00–14:35	155
E	57	20 December 2007	Block IIRM	10	0.8	13:15–14:35	80
F	51	11 May 2000	Block IIR	7.5	0.75	13:55–14:35	40
G	36	10 March 1994	Block IIA	7.5	0.7	14:29–14:35	6

receiver, a computer and meteorological equipment. The receiver hardware mainly consists of several parts: the controller, host, power supply and antenna.

The main functions of the receiver are as follows: (1) Receive signals from satellites in the satellite navigation system capture the signals selected by the satellite's cut-off angle, check the operating orbits of the satellites and calculate the user's position information and the measurements of the satellites; (2) Perform data conversion, expansion and calculation on the signals received from the navigation system to calculate the transmission time of the signal from the navigation satellite to the receiver antenna; (3) Calculate the user's position, time and velocity (PVT) based on the data transmitted from the navigation satellite; (4) Present the processed data to the user through the data display.

2.4. Reference models

In our study, commercial aircraft is considered to be the user, and the analysis considers the impact of navigation satellites' availability on aircraft navigation throughout the flight mission. Fig. 4 shows a schema of satellite navigation. At least four satellites are required for satellite navigation. In the schematic diagram, the user receives navigation signals from satellites with the serial numbers *A*, *B*, *C*, and *D*. However, both users and navigation satellites are constantly moving as the users are processing the information. Thus, the information that the users receive from the navigation satellites is also constantly changing.

A particular flight was studied in this paper. The flight was from Beijing to Guangzhou, and the entire flight time was 2 h 35 min. The specific time was January 2, 2012 (Beijing time); the flight departed at 12:00 and arrived in Guangzhou at 14:39. The entire flight was guided sequentially by 17 navigation satellites. Although the airplane could generally receive satellite signals from more than 4 satellites at a time, usually only the signals from the four satellites with the best signals were used by the receiver for calculating the position. Therefore, 7 out of 17 satellites were determined to be the navigation satellites to be analysed in this study based on their navigation times and the task of the flight. The SVNs (Space Vehicle Numbers) of these 7 satellites were *SVN49*, *SVN39*, *SVN55*, *SVN58*, *SVN57*, *SVN51* and *SVN36*. The parameters of navigation satellites are shown in Table 1.

The system comprises 5 components: the satellite, monitor station, master control station, ground antenna and user. Each subsystem transmits information to objects to which it is connected. The user receives a satellite signal. The satellite receives information from the ground antenna which it then transmits to the monitor station and the user. The monitor station receives information from the satellite and transmits it to the master control station. The master station analyses the data from the monitor station and transmits it to the ground antenna. The ground antenna receives the control commands from the master control station and sends them to the satellite.

The US National Geospatial-Intelligence Agency (NGA) provides GPS satellites' status data available daily.[1] The space segment consists of 7 satellites due to the fact that the navigation mission requires a minimum of 7 satellites, which are identified as *A*, *B*, *C*,

[1] http://www.navcen.uscg.gov/?Do=constellationStatus.

D, *E*, *F* and *G*. They receive information from ground stations and transmit navigation information to the user.

3. The formal approach

It is fundamental to have an effective solution to the challenge of verifying large and complex satellite systems. Generally, simulation is the common testing and validation approach used for the verification of such systems. Given a system, a finite subset of the possible scenarios are selected in a specific simulation environment, and then statistical analysis techniques are applied to obtain probabilistic results on that system. Simulation based verification has been unable to keep pace with the growth in design complexity. As simulation requires the number of scenarios and simulation environments to be restricted, and so one cannot ensure that all conditions have been covered. Formal verification, on the other hand, can be applied to model and verify all scenarios. One automated method of verification is model checking. In particular, probabilistic model checking has proved to be a suitable formal verification technique for exposing errors in satellite systems, mainly due to classical concurrency errors.

Probabilistic verification mainly consists of four stages as illustrated in Fig. 5. First, we model in the probabilistic π-calculus the behaviour of the whole mission. This model is composed of two separate models characterising the communication between different segments and their mobility. The latter must be able to be modified without changing the former. Second, the global model is translated into PRISM, and is then internally generated into an MDP (stage 1). The availability requirements that the system must satisfy are formalised in Probabilistic Computation Tree Logic (PCTL) [14] properties (stage 2). These formal quantitative properties are then checked with PRISM (stage 3). They can be checked according to our specific flight navigation mission. Finally, we analyse the results given by PRISM (stage 4).

3.1. The PRISM model checker

In this paper, we use the PRISM probabilistic model checker [10]. 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 probabilistic timed automata (PTAs), with optional extensions of costs and rewards. Moreover, 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.

Fig. 6 shows a screenshot of the PRISM graphical user interface, illustrating the results of a model checking verification being plotted on a graph. In addition, the automated tool integrate a text editor for building probabilistic models based on the PRISM language. PRISM is a free and open source application, and it supports difference operating systems such as Mac OS X, Windows, and Linux.

3.1.1. Markov decision processes (MDPs)

Our approach is event based because of the fault and failure events that can be sensed and monitored in the satellite systems, and our underlying semantics is MDPs. In this section, we briefly review the basic concepts of MDPs.

Definition 1. Let *Act* be a set of actions, and *AP* a fixed, finite set of atomic propositions. Formally, a Markov decision process (MDP) $\mathcal{M}$ is a tuple $(S, s_{init}, Steps, \mathcal{L})$ where

- $S = \{s_1, s_2, \ldots, s_n\}$ is a finite set of states;
- $s_{init} \in S$ is the initial state;
- $Steps : S \to 2^{Act \times Dist(S)}$ is the transition probability function where *Act* is a set of actions and $Dist(s)$ is the set of discrete probability distributions over the set S;
- $\mathcal{L} : S \to 2^{AP}$ is a labelling function with atomic propositions.

We model a very simple communication protocol using an MDP in Fig. 7. There is a single process. After one step, the process starts trying to send a message. Then, a nondeterministic choice is made between: (a) waiting because the channel is not ready; (b) sending the message. If the latter choice is made, with probability 0.99 the message is sent successfully and stops, and with probability 0.01, message sending fails, and the process restarts.

3.1.2. Reactive modules of PRISM

Markov models to be verified using PRISM are specified using the PRISM modelling language which is based on the Reactive Modules formalism [11]. A fundamental component of this language is a *module*. A system is represented as the parallel composition of a number of modules. A module is specified as

module *name* … **endmodule**

A module definition consists of two parts: one containing variable declarations, and the other *commands*. At any time, the

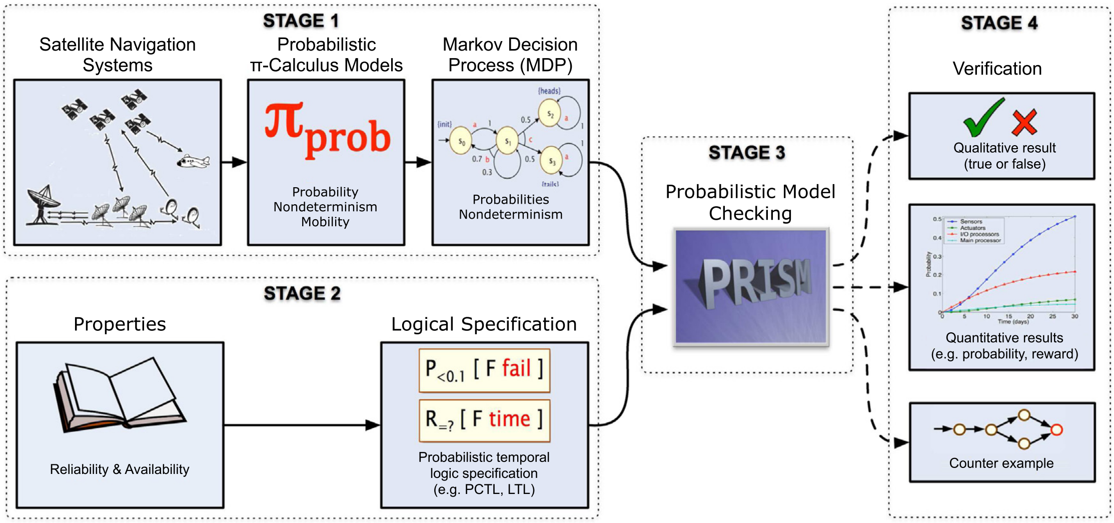
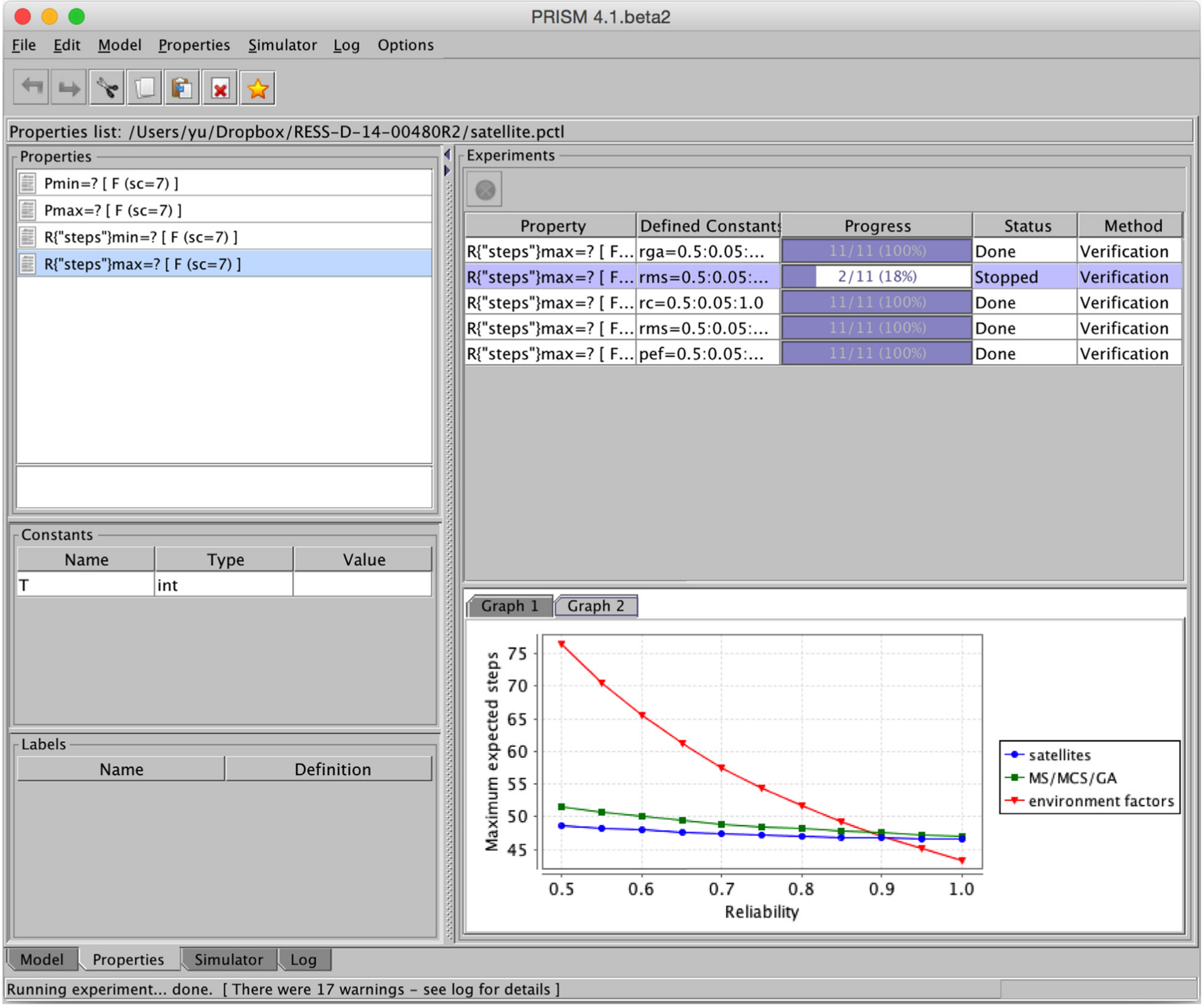


Fig. 5. Stages in probabilistic verification.



Properties list: /Users/yu/Dropbox/RESS-D-14-00480R2/satellite.pctl

Properties

- Pmin=? [F (sc=7)]
- Pmax=? [F (sc=7)]
- R{"steps"}min=? [F (sc=7)]
- R{"steps"}max=? [F (sc=7)]

Experiments

Property	Defined Constants	Progress	Status	Method
R{"steps"}max=? [F...	rga=0.5:0.05:...	11/11 (100%)	Done	Verification
R{"steps"}max=? [F...	rms=0.5:0.05:...	2/11 (18%)	Stopped	Verification
R{"steps"}max=? [F...	rc=0.5:0.05:1.0	11/11 (100%)	Done	Verification
R{"steps"}max=? [F...	rms=0.5:0.05:...	11/11 (100%)	Done	Verification
R{"steps"}max=? [F...	pef=0.5:0.05:...	11/11 (100%)	Done	Verification

Constants

Name	Type	Value
T	int	

Labels

Name	Definition

Running experiment... done. [There were 17 warnings – see log for details]

Fig. 6. A screenshot of the PRISM model checker.

state of a model is determined by the current value of all of the variables of all of the components (modules). A variable declaration has the form

x : [0 .. 2] **init** 0;

In this example, variable x is declared, with range $[0..2]$ and initial value 0. The behaviour of each module is specified using commands, which include a guard and one or more updates of the form:

[*action*] *guard* $\to$ *probability* : *update*

or,

[*action*] *guard* $\to$ p_1 : $update_1 + p_2$: $update_2 + \cdots$

The (action) label is optional, and is used to force two or more modules to synchronise. The + indicates the usual nondeterministic choice. Within a module, multiple transitions can be modelled either as different individual updates in a command, or as multiple commands with overlapping guards. The following examples:

[ ] $x=0\to0.2:(x'=0)$;
[ ] $x=0\to\ 0.8:(x'=1)$;

and

[ ] $x=0\to\ 0.2:(x'=0)\ +\ 0.8:(x'=1)$;

are equivalent. The guard $x=0$ indicates that command is only executed when variable x has value 0. The updates $(x'=0)$ and $(x'=1)$ and their associated probabilities indicate that the value of x will remain at 0 with probability 0.2 and change to 1 with probability 0.8.

3.2. Overview of the probabilistic π-calculus

The probabilistic π-calculus is a probabilistic extension of the process algebra π-calculus. π-calculus are used to model communicating, distributed, and mobile systems, and it provides strong techniques to reason about systems with concurrency at the modelling level. Communication, either inside the system or between the system and its environment, are modelled by synchronous actions on shared channels. It allows channel names to be sent along the channels themselves, thereby enabling one to model dynamically changing networks. Mobility is of central importance in the π-calculus. To appreciate the definition of the probabilistic π-calculus and the mobility our case study in satellite navigation for aviation, we illustrate what kind of mobility that the π-calculus is suitable for.

3.2.1. The probabilistic π-calculus

For some classes of concurrent and mobile systems, probabilistic behaviours can be key ingredients. Satellite navigation systems, for instance, can exhibit probabilistic behaviours due to either unreliable communication or component failures. We can model such behaviours with the probabilistic π-calculus (π_{proc}). The basic component of probabilistic π-calculus is its syntax as determined by the well-formed combination of operators and more elementary terms. We use "terms" to describe systems, and these are then mapped to labelled transition systems (LTSs). More explicitly, the states of a LTS are just "terms" of the probabilistic π-calculus while the labels of transitions between states represent the actions or the interactions that are possible from a given state and the state that is reached after the action is performed by means of actions.

The probabilistic π-calculus adds a discrete probabilistic choice operator to the classical π-calculus (only non-deterministic choice operator exists). This probabilistic operator associates internal actions with probabilities.

Definition 2 (*Syntax*). We assume P and P_i range over terms and α ranges over actions. We assume a countable set of names $\mathcal{N}$ that range over x, y, x_i, where $i \in \{1, 2, \ldots, n\}$. A process P is defined in π_{proc} using the following syntax:

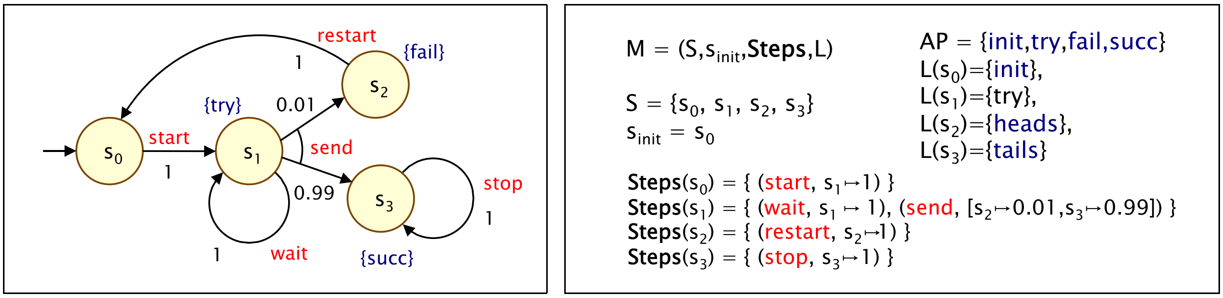
- $\alpha ::= \tau \mid x(y) \mid \overline{x}\langle y \rangle$
- $P ::= \mathbf{0} \mid \alpha.P \mid \sum_{i \in I} P_i \mid \sum_{i \in I} p_i \tau.P_i \mid P \mid P \mid \nu x P \mid [x = y]P \mid A(x_1, x_2, \ldots, x_i, \ldots, x_n)$,

where I is an index set, $p_i \in (0, 1]$ with $\sum_{i \in I} p_i = 1$, and A is a process identifier. We now informally describe the calculus.

The inactive process $\mathbf{0}$ can perform no actions. The process $\alpha.P$ performs action α and then evolves into process P, where α is one of three types: τ is the silent (invisible) action that corresponds to an internal interaction between sub-processes, $x(y)$ is an input action in which a process receives a name y on channel x, and $\overline{x}\langle y \rangle$ is an output action, in which a process sends a name y on channel x. There are two types of summation: nondeterministic choice $\sum_{i \in I} P_i$ and probabilistic choice $\sum_{i \in I} p_i \tau.P_i$. The first is common in the standard π-calculus, and the second is a new operator in π_{proc}. As for π_{proc}, branches of the probabilistic choice operator are normally prefixed with τ actions. Thus, the process $\sum_{i \in I} p_i \tau.P_i$ randomly selects an index $i \in I$ with probability p_i, performs a τ action, and then evolves to P_i.

The parallel composition of processes P_i and P_j is $P_i \mid P_j$, and it can either proceed in an asynchronous manner or synchronise between P_i and P_j via matching input and output actions. The restriction $\nu x P$ locally sets the scope of x in process P, so x is treated as a new and unique name within P. The match $[x = y]$ checks whether names x and y are identical, so the process $[x = y]P$ can evolve into process P only if the match $[x = y]$ is satisfied. Finally, $A(x_1, x_2, \ldots, x_i, \ldots, x_n)$ corresponds to a process definition clause and is used in the context $P = A(x_1, x_2, \ldots, x_i, \ldots, x_n)$.

The operational semantics of π_{proc} are typically expressed in terms of Markov Decision Processes (MDPs) or Probabilistic



M = (S, s_{init}, Steps, L)

S = {s_0, s_1, s_2, s_3}
s_{init} = s_0

AP = {init, try, fail, succ}
L(s_0)={init},
L(s_1)={try},
L(s_2)={heads},
L(s_3)={tails}

Steps(s_0) = { (start, $s_1 \mapsto 1$) }
Steps(s_1) = { (wait, $s_1 \mapsto 1$), (send, [$s_2 \mapsto 0.01$, $s_3 \mapsto 0.99$]) }
Steps(s_2) = { (restart, $s_2 \mapsto 1$) }
Steps(s_3) = { (stop, $s_3 \mapsto 1$) }

Fig. 7. An example of an MDP.

PRE $$\frac{}{\alpha.P \xrightarrow{\alpha} \{\!|1 : P|\!\}}$$

PROB $$\frac{}{(\sum_i p_i \tau.P_i) \xrightarrow{\tau} \{\!|p_i : P_i|\!\}_i}$$

SUM $$\frac{P_j \xrightarrow{M,\alpha} \{\!|p_{j_k} : P_{j_k}|\!\}_{j_k}}{(\sum_{i \in I} P_i) \xrightarrow{M,\alpha} \{\!|p_{j_k} : P_{j_k}|\!\}_{j_k}}\; j \in I$$

PAR $$\frac{P \xrightarrow{M,\alpha} \{\!|p_i : P_i|\!\}_i}{P \mid Q \xrightarrow{M,\alpha} \{\!|p_i : (P_i \mid Q)|\!\}_i}\; bn(\alpha) \cap fn(Q) = \emptyset$$

COM $$\frac{P \xrightarrow{M,y(z)} \{\!|1 : P'|\!\} \quad Q \xrightarrow{N,\bar{x}v} \{\!|1 : Q'|\!\}}{P \mid Q \xrightarrow{[x=y] \wedge M \wedge N,\tau} \{\!|1 : P'\{v/z\} \mid Q'|\!\}}$$

RES $$\frac{P \xrightarrow{M,\alpha} \{\!|p_i : P_i|\!\}_i}{\nu x\, P \xrightarrow{\nu x M,\alpha} \{\!|p_i : \nu x\, P_i|\!\}_i}\; x \notin n(\alpha)$$

CLOSE $$\frac{P \xrightarrow{M,y(z)} \{\!|1 : P'|\!\} \quad Q \xrightarrow{N,\bar{x}(v)} \{\!|1 : Q'|\!\}}{P \mid Q \xrightarrow{[x=y] \wedge M \wedge N,\tau} \{\!|1 : \nu v(P'\{v/z\} \mid Q')|\!\}}$$

OPEN $$\frac{P \xrightarrow{M,\bar{y}x} \{\!|1 : P'|\!\}}{\nu x\, P \xrightarrow{\nu x M,\bar{y}(x)} \{\!|1 : P'|\!\}}\; x \neq y$$

MATCH $$\frac{P \xrightarrow{M,\alpha} \{\!|p_i : P_i|\!\}_i}{[x=y]P \xrightarrow{[x=y] \wedge M,\alpha} \{\!|p_i : P_i|\!\}_i}\; \{x, y\} \cap bn(\alpha) = \emptyset$$

IDE $$\frac{P\{y_1, \ldots, y_n/x_1, \ldots, x_n\} \xrightarrow{M,\alpha} \{\!|p_i : P_i|\!\}_i}{A(y_1, \ldots, y_n) \xrightarrow{M,\alpha} \{\!|p_i : P_i|\!\}_i}\; A(x_1, \ldots, x_n) \triangleq P$$

$$\begin{array}{rcll}
\nu x\,(\mathit{true}) & = & \mathit{true} & \\
\nu x\,[x{=}x] & = & \mathit{true} & \\
\nu x\,[x{=}y] & = & \mathit{false} & (x{\neq}y) \\
\nu x\,[y{=}z] & = & [y{=}z] & (x{\neq}y \wedge x{\neq}z) \\
\nu x\,(M \wedge N) & = & (\nu x\, M) \wedge (\nu x\, N) &
\end{array}$$

Fig. 8. The symbolic semantics for π_{prob}.

Automata (PAs). The symbolic semantics of π_{proc} is expressed in terms of *probabilistic symbolic transition graphs* (PSTGs). These are a simple probabilistic extension of the *symbolic transition graphs* introduced in [15].

Definition 3. Let P be a π_{prob} process. The probabilistic symbolic transition graph (PSTG) representing the semantics of the process P is a tuple $(S, s_{init}, \mathcal{T}_{prob})$ where

- S is a finite set of symbolic states, each of which is a term of the probabilistic π-calculus;
- $s_{init} \in S$, the initial state, is the term P;
- $\mathcal{T}_{prob} \subseteq S \times Cond \times Act \times Dist(S)$ is the probabilistic symbolic transition relation and is the least relation given by the rules in Fig. 8.

In the above, *Cond* denotes the set of all conditions (finite conjunctions of matches) over the set of names $\mathcal{N}$. *Act* is a set of actions of basic types: τ, $x(y)$, $\overline{x}\langle y\rangle$, where $x, y \in \mathcal{N}$. $Dist(S)$ is the set of probability distributions over S. The notation $Q_i \xrightarrow{M,\alpha} \{| p_i : R_i^i |\}$ is used for the probabilistic symbolic transition $(Q, M, \alpha, \mu) \in \mathcal{T}_{prob}$, where $\mu(R) = \sum_{Q_i = R} p_i$ for any π_{prob} term R. The multi-sets [9] are used to ensure that processes with duplicate components such as $Q = 0.5\tau.\mathbf{0} \oplus 0.5\tau.\mathbf{0}$ have transition of the form $Q_i \xrightarrow{\tau} \{| 0.5 : \mathbf{0}, 0.5 : \mathbf{0} |\}$.

3.2.2. An example of π_{proc} processes

To illustrate the idea of probabilistic models, we present a simple π_{proc} model of a set of traffic lights and drivers from [16]. The process P_{light} models the traffic lights signalling to drivers, which are probabilistically red, yellow, or green.

$$P_{light} \triangleq 0.45\tau.\overline{a}\langle Red\rangle.P_{light} \oplus 0.1\tau.\overline{a}\langle Yellow\rangle.P_{light} \oplus 0.45\tau.\overline{a}\langle Green\rangle.P_{light}$$

Here, the traffic light is red with probability 0.45, yellow with probability 0.1, and green with probability 0.45. We distinguish drivers according to how they behave depending on the colours of the lights they see. A cautious driver is modelled by the process P_{c_driver} as follows:

$$P_{c\_driver} \triangleq a(x).([x = red]P_{c\_red} + [x = yellow]P_{c\_yellow} + [x = green]P_{c\_green})$$

$$P_{c\_red} \triangleq 0.2\tau.\overline{b}\langle braking\rangle.\mathbf{0} \oplus 0.8\tau.\overline{b}\langle stopped\rangle.\mathbf{0}$$

$$P_{c\_yellow} \triangleq 0.9\tau.\overline{b}\langle braking\rangle.\mathbf{0} \oplus 0.1\tau.\overline{b}\langle driving\rangle.\mathbf{0}$$

$$P_{c\_green} \triangleq \overline{b}\langle driving\rangle.\mathbf{0}$$

A cautious driver sees what colour the light is (through the form of match $[x = y]$) and behaves accordingly (through the probabilistic choice: $\sum_{i \in I} p_i \tau.P_i$). If it is red, he brakes or stops. If it is yellow, mostly likely he brakes. If it is green, he drives on. Similarly, an aggressive driver can be modelled by the process P_{a_driver} as follows:

$$P_{a\_driver} \triangleq a(x).([x = red]P_{a\_red} + [x = yellow]P_{a\_yellow} + [x = green]P_{a\_green})$$

$$P_{a\_red} \triangleq 0.3\tau.\overline{b}\langle braking\rangle.\mathbf{0} \oplus 0.6\tau.\overline{b}\langle stopped\rangle.P_{a\_driver} \oplus 0.1\tau.\overline{b}\langle driving\rangle.\mathbf{0}$$

$$P_{a\_yellow} \triangleq 0.1\tau.\overline{b}\langle braking\rangle.P_{a\_driver} \oplus 0.9\tau.\overline{b}\langle driving\rangle.\mathbf{0}$$

$$P_{a\_green} \triangleq \overline{b}\langle driving\rangle.\mathbf{0}$$

Therefore, the aggressive driver is more likely to drive on at red and yellow. We may analyse what is the probability of a crash if two different drivers go through a single traffic light from different streets. The behave process P_{a_driver} is as the following:

$$P_{behave} \triangleq b(y).([y = braking]\mathbf{0} + [y = stopped]\mathbf{0} + [y = driving]\mathbf{0}$$

3.3. Translation of a π_{proc} model into the PRISM language

We show that for closed and finite processes (i.e., which do not replicate themselves), the semantics of a probabilistic π-calculus process can be represented by an MDP.

3.3.1. Translation rules

We assume that the set of all names in the system is $\mathcal{N}$, which is partitioned into disjoint subsets: $\mathcal{N}^{fn}$, the set of all free names appearing in processes $P_1, P_2, \ldots, P_i, \ldots, P_n$, and $\mathcal{N}_1^{bn}, \mathcal{N}_2^{bn}, \ldots, \mathcal{N}_i^{bn}, \ldots, \mathcal{N}_n^{bn}$, the sets of input-bound names for processes $P_1, P_2, \ldots, P_i, \ldots, P_n$. A match is an equality test on names from $\mathcal{N}$ and a condition M is a finite conjunction of matches, i.e., M is of the form $[x_1 = y_1] \wedge \ldots [x_n = y_n]$. The translation rules of a π_{proc} model into the PRISM language, defined in [9], can be summarised as follows:

- *Rule 1.* Each of the n sub-processes P_i becomes a PRISM *module* with the same name.
- *Rule 2.* Each element Q_j^i of the finite set of terms $S_i = \{Q_1^i, \ldots, Q_k^i\}$, which is the set of the states of process P_i after each of its transitions (In [9], the set of all these states is called the PSTG of P_i), becomes an integer variable s_i whose values vary from 1 to k.
- *Rule 3.* Module P_i has $|\mathcal{N}_i^{bn}| + 1$ local variables. Each bound name x_j^i of process P_i has a corresponding variable x_j^i with range $0, \ldots, |\mathcal{N}^{fn}|$ and it is initialised to 0.
- *Rule 4.* The model includes $|\mathcal{N}^{fn}|$ integer constants, one for each free name, which are assigned distinct, consecutive non-zero values. If the value of variable x_j^i is equal to one of these constants, then the corresponding bound name has been assigned the appropriate free name (by an input action). On the contrary, $x_j^i = 0$ means that no input to the bound name has occurred yet.
- *Rule 5.* (*Probabilistic internal transition*). For a transition $Q_i \xrightarrow{M,\tau} \{| p_1 : R_1^i, \ldots, p_m : R_m^i |\}$, we add the command:

  $$[](s_i = Q_i)\&M \rightarrow p_1 : (s_1' = R_1^i) + \ \ + p_m : (s_i' = R_m^i).$$

- *Rule 6.* (*Output on free name*). Process P_i outputs y on free name x to P_j. For a transition $Q_i \xrightarrow{M,\overline{x}\langle y\rangle} R_i$, where $x \in \mathcal{N}^{fn}$, we add, for each $j \in \{1, \ldots, n\} \backslash \{i\}$, the command:

  $$[x\_P_i\_P_j\_y](s_i = Q_i)\&M \rightarrow (s_i' = R_i).$$

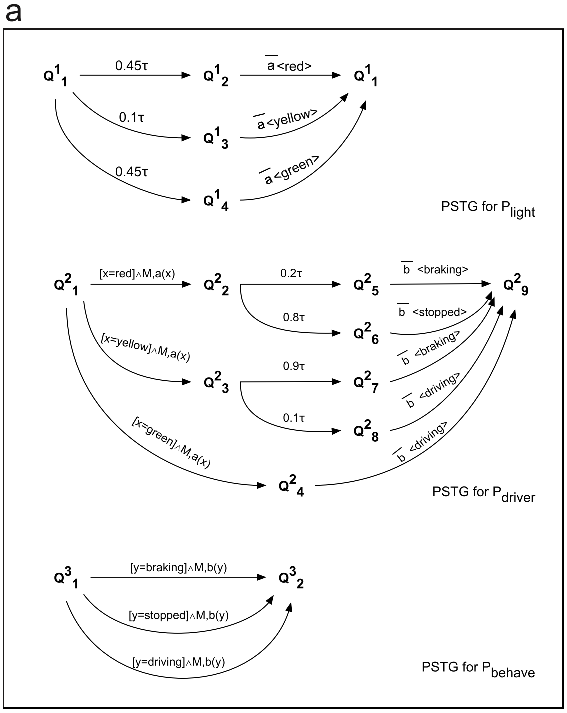
  The channel x, sender P_i, receiver P_j, and sent name y are all encoded in the action label. See [9] for details.
- *Rule 7.* (*Output on bound name*). Process P_i outputs y on bound name x to P_j. For a transition $Q_i \xrightarrow{M,\overline{x}\langle y\rangle} R_i$, where $x \in \mathcal{N}_i^{bn}$, we add, for each $a \in \mathcal{N}^{fn}$ and $j \in \{1, \ldots, n\} \backslash \{i\}$, the command:

  $$[a\_P_i\_P_j\_y](s_i = Q_i)\&M\&(x = a) \rightarrow (s_i' = R_i).$$

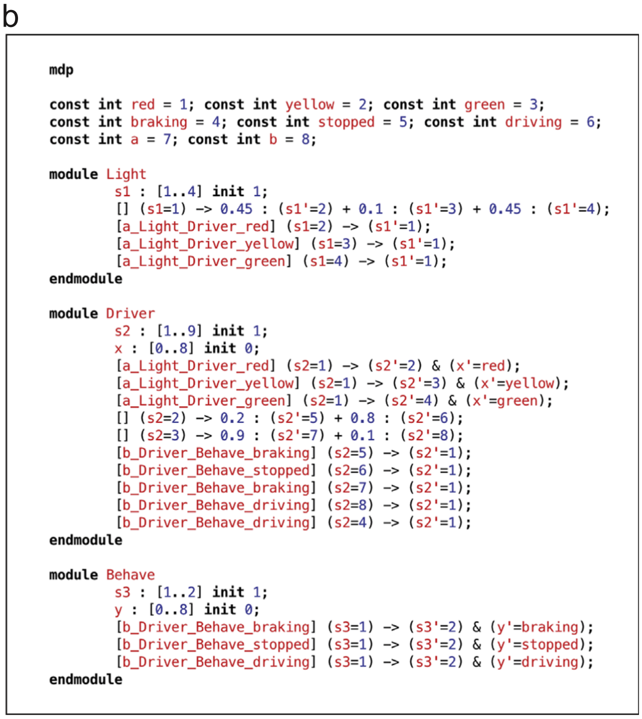
  This is similar to Rule 6 except that it includes a command for each possible value a of x.
- *Rule 8.* (*Input on free name*). Process P_j inputs z on free name x from P_i. For a transition $Q_i \xrightarrow{M,x(z)} R_i$, where $x \in \mathcal{N}^{fn}$, we add, for each $y \in \mathcal{N} \backslash \mathcal{N}_i^{bn}$ and $j \in \{1, \ldots, n\} \backslash \{i\}$, the command:

  $$[x\_P_j\_P_i\_y](s_i = Q_i)\&M \rightarrow (s_i' = R_i)\&(z' = y).$$

  For input actions, an extra assignment $(z' = y)$ is added to



PTSGs



```
mdp

const int red = 1; const int yellow = 2; const int green = 3;
const int braking = 4; const int stopped = 5; const int driving = 6;
const int a = 7; const int b = 8;

module Light
        s1 : [1..4] init 1;
        [] (s1=1) -> 0.45 : (s1'=2) + 0.1 : (s1'=3) + 0.45 : (s1'=4);
        [a_Light_Driver_red] (s1=2) -> (s1'=1);
        [a_Light_Driver_yellow] (s1=3) -> (s1'=1);
        [a_Light_Driver_green] (s1=4) -> (s1'=1);
endmodule

module Driver
        s2 : [1..9] init 1;
        x : [0..8] init 0;
        [a_Light_Driver_red] (s2=1) -> (s2'=2) & (x'=red);
        [a_Light_Driver_yellow] (s2=1) -> (s2'=3) & (x'=yellow);
        [a_Light_Driver_green] (s2=1) -> (s2'=4) & (x'=green);
        [] (s2=2) -> 0.2 : (s2'=5) + 0.8 : (s2'=6);
        [] (s2=3) -> 0.9 : (s2'=7) + 0.1 : (s2'=8);
        [b_Driver_Behave_braking] (s2=5) -> (s2'=1);
        [b_Driver_Behave_stopped] (s2=6) -> (s2'=1);
        [b_Driver_Behave_braking] (s2=7) -> (s2'=1);
        [b_Driver_Behave_driving] (s2=8) -> (s2'=1);
        [b_Driver_Behave_driving] (s2=4) -> (s2'=1);
endmodule

module Behave
        s3 : [1..2] init 1;
        y : [0..8] init 0;
        [b_Driver_Behave_braking] (s3=1) -> (s3'=2) & (y'=braking);
        [b_Driver_Behave_stopped] (s3=1) -> (s3'=2) & (y'=stopped);
        [b_Driver_Behave_driving] (s3=1) -> (s3'=2) & (y'=driving);
endmodule
```

PRISM code

Fig. 9. Model transformation example of traffic light and driver.

consider each possible received name y. It models the update of the bound name z to y.

- *Rule 9.* (*Input on bound name*). Process P_j inputs z on bound name x from P_i. For a transition $Q_i \xrightarrow{M,x(z)} R_i$, where $x \in \mathcal{N}_i^{bn}$, we add, for each $a \in \mathcal{N}^{fn}$, $y \in \mathcal{N}\setminus\mathcal{N}_i^{bn}$ and $j \in \{1,\dots,n\}\setminus\{i\}$, the command:

  $[a\_P_j\_P_i\_y](s_i=Q_i)\&M\&(x=a)\rightarrow(s'_i=R_i)\&(z'=y).$

  This rule combines elements of Rules 8 and 9, since a command is added to consider each possible pairing of channel a that x may represent and name y that may be received.

In addition, Rules 8 and 9 include some commands that need to be removed. More specifically, labels $x\_P_i\_P_j\_y$ appear on a command of each module P_j, but do not appear in any of the commands in module P_i. Therefore, commands with such action labels are removed from P_j.

3.3.2. *Translation of the example*

In Fig. 9, we show an example translation for the traffic light example in Section 3.3.2. The intermediate PTSG is illustrated in Fig. 9(a), and the PRISM model is shown in Fig. 9(b).

4. Specification

4.1. *The π_{prob} models*

The formal models of the system consist of 12 π_{prob} processes (P_A, P_B, P_C, P_D, P_E, P_F, P_G, *MS*, *MCS*, *GA*, *Uer*, *Switch*) for different subsystems: each of 7 processes for each of 7 satellites, 1 process for the monitor station, 1 process for the master control station, 1 process for the ground antenna, 1 process for the user, and 1 process for the mobility model. There are also 7 types of channels: a, b, c, d, e, which are used for transmitting messages between subsystems.

4.1.1. *π_{prob} models of the space segment*

The model of the space segment consists of 7 π_{prob} processes of 7 satellites, referred to P_A, P_B, P_C, P_D, P_E, P_F and P_G. These satellites receive information from the ground antenna simultaneously and then transmit the navigation information to the user via the monitor station. In this paper, the user and the monitor station are assumed to receive navigation signals from the satellites simultaneously.

There are 3 types of channels for each of the 7 satellites, in which d_i (where $i \in \{1,2,\dots,7\}$) is the channel between the ground antenna and each individual satellite j (where $j \in \{A,B,C,D,E,F,G\}$ for all following denotations), e_i is the channel between the satellite and the aircraft, and a_i the channel between the satellite and the monitor station. The π_{prob} model of satellite C, D, and E of the space segment are given as below, and the π_{prob} models of other satellites can be derived similarly.

$$P_C \triangleq r_c\tau.\overline{a_3}\langle m_c\rangle.d_3(x_c).([x_c=v_c]P'_C+[x_c=no]P_C) \oplus (1-r_c)\tau.P_C$$
$$P'_C \triangleq r_c\tau.\overline{e_3}\langle m_c\rangle.out_c(y_c).\mathbf{0} \oplus (1-r_c)\tau.P'_C$$
$$P_D \triangleq r_d\tau.\overline{a_4}\langle m_d\rangle.d_4(x_d).([x_d=v_d]P'_D+[x_d=no]P_D) \oplus (1-r_d)\tau.P_D$$
$$P'_D \triangleq r_d\tau.\overline{e_4}\langle m_d\rangle.\mathbf{0} \oplus (1-r_d)\tau.P'_D$$
$$P_E \triangleq in_e(y_e).P'_E$$
$$P'_E \triangleq r_e\tau.\overline{a_5}\langle m_e\rangle.d_5(x_e).([x_e=v_e]E''+[x_e=no]P'_E) \oplus (1-r_e)\tau.P'_E$$
$$P''_E \triangleq r_e\tau.\overline{y_e}\langle m_e\rangle.\mathbf{0} \oplus (1-r_e)\tau.P''_E$$

In the above, r_j denotes the reliability of transmission from the corresponding satellite to the monitor station, which is represented by reliability (probability) as shown in Table 1. For communication, m_j is the information sent by a satellite to the monitor station via channel a_i. Afterwards this message is relayed to the master control station and verified there, so, v_j denotes that the message has been verified, otherwise it will be *no* for the message that has been corrupted due to the influence of environmental factors. The satellite then sends the verified message to the aircraft via channel e_i for the purpose of continuous navigation.

4.1.2. π_{prob} models of the control segment

Here, navigation information mainly refers to the data of the on-orbit state of satellites that are transmitted by the navigation satellites. Satellites in this system do not exchange information with one another. In the real world, all GPS satellites are monitored by a set of 6 monitor stations. In this paper, we make the simplifying assumption that there is a single monitor station, which is essentially a combination of the 6 stations. As a result, each satellite transmits information to the monitor station independently and simultaneously. The π_{prob} model of the monitor stations is MS.

$$\begin{aligned} MS \triangleq\ & a_1(x).MS_1+a_2(x).MS_2+a_3(x).MS_3+a_4(x).MS_4+a_5(x).MS_5 \\ & +a_6(x).MS_6+a_7(x).MS_7 \end{aligned}$$

$$MS_i \triangleq r_{ms}\tau.\overline{b}\langle x\rangle.MS \oplus (1-r_{ms})\tau.MS_i(1 \le i \le 7)$$

In the above, MS_i denotes the processes for communication between satellites A, B, C, D, E, F, and G and the monitor station respectively. The direct summation $+$ is used due to the fact that in our assumption the single monitor station (MS) is unable to receive simultaneous transmissions, so there will be a nondeterministic choice between simultaneous transmissions from different satellites to the monitor station. Then, r_{ms} denotes reliability of transmission from the monitor station to the master control station, which is a probability ($r_{ms}=0.99999$ as default) as shown in Table 4. For communication, x is the message received from the satellite and relayed to the master control station via channel b.

The master control station receives information from the monitor station via channel b, then transmits it to the ground antenna via channel c. Further, MCS_i represents the sub-process for verifying the relayed message from a satellite via the monitor station. Its π_{prob} model is MCS, defined as the following process:

$$\begin{aligned} MCS \triangleq\ & b(x).([x=m_a]MCS_1+[x=m_b]MCS_2+[x=m_c]MCS_3 \\ & +[x=m_d]MCS_4+[x=m_e]MCS_5+[x=m_f]MCS_6+[x=m_g]MCS_7) \end{aligned}$$

$$\begin{aligned} MCS_i \triangleq\ & r_{mcs}\cdot p_{ef}\tau.\overline{c}\langle v_j\rangle.MCS \oplus r_{mcs}\cdot(1-p_{ef})\tau.\overline{c}\langle n_j\rangle.MCS \\ & \oplus(1-r_{mcs})\tau.MCS_i((i,j)\in\{(1,a),(2,b),(3,c),(4,d),(5,e),(6,f),(7,g)\}) \end{aligned}$$

In the above, r_{mcs} denotes the reliability of transmission from the monitor station to the master control station, which is a probability ($r_{mcs}=0.99999$ as default) as shown in Table 4. p_{ef} is the probability of whether the message is corrupted due to the influence of environmental factors. For communication, the master control station sends the verified result back to the corresponding satellite through the ground antenna via channel c. The nondeterministic choice $+$ is used for name matching of messages sent from the monitor station to the master control station.

Similar to the monitor station, the ground antenna communicates with the 7 satellites simultaneously. There are 4 ground antennas worldwide that perform the daily routine of transmitting commands to each satellite. We also make a similar abstraction that we use a single ground antenna instead of the 4 original ground antennas. The π_{prob} model of the ground antenna is GA, defined as the following process:

$$\begin{aligned} GA \triangleq\ & c(y).([y=v_a]GA_1+[y=n_a]GA_1+[y=v_b]GA_2+[y=n_b]GA_2 \\ & +[y=v_c]GA_3+[y=n_c]GA_3+[y=v_d]GA_4+[y=n_d]GA_4 \\ & +[y=v_e]GA_5+[y=n_e]GA_5+[y=v_f]GA_6+[y=n_f]GA_6 \\ & +[y=v_g]GA_7)+[y=n_g]GA_7 \end{aligned}$$

$$GA_i \triangleq r_{ga}\tau.\overline{d_i}\langle y\rangle.GA \oplus (1-r_{ga})\tau.GA_i\ (1 \le i \le 7)$$

In the above, GA_i denotes the processes for communication between the ground antenna and a satellite. Then, r_{ga} denotes the reliability of transmission from the ground antenna to the corresponding satellite, which is a probability ($r_{ga}=0.99999$ as default) as shown in Table 4. For communication, the ground antenna receives the verified result from the master control station via channel c, and then sends the message to the different satellites based on the verified result via channel d_i respectively. Similarly, the nondeterministic choice $+$ is used for name matching of messages sent from the master control station to the ground antenna.

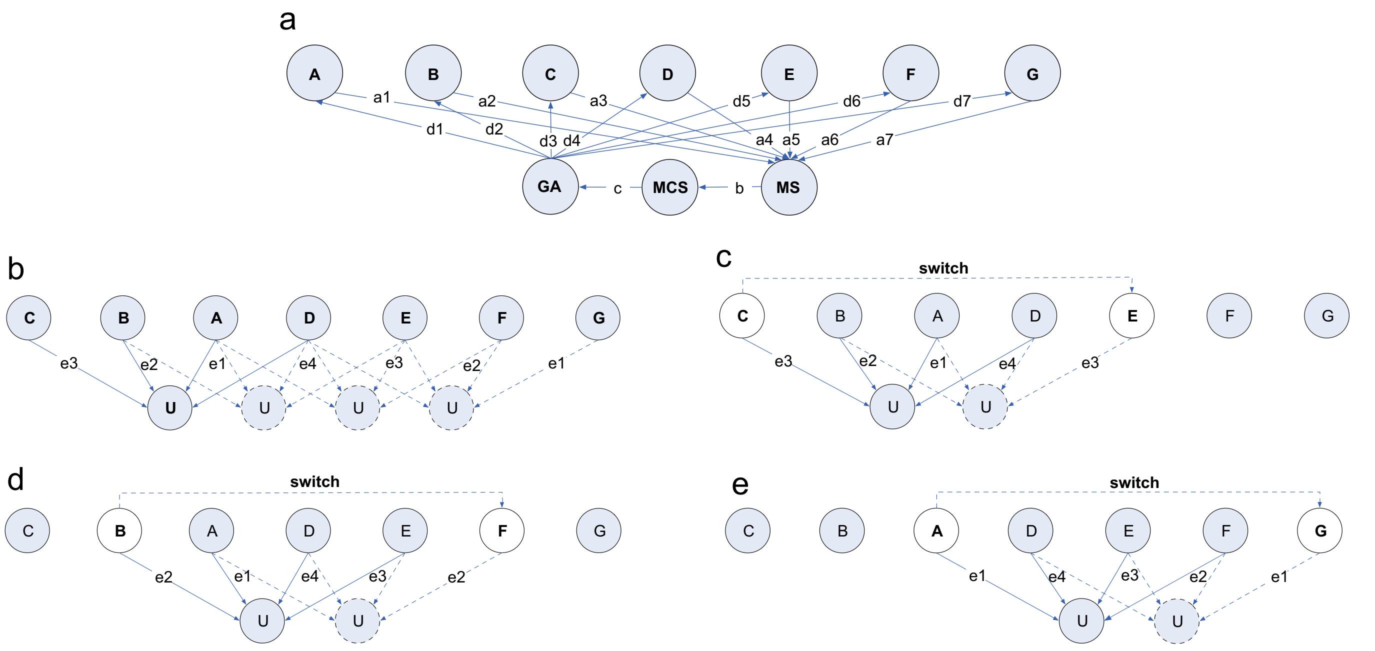


Fig. 10. Reference Model of GNSS Segments. (a) Reference model of control and space segments. (b) Reference model of user and space segments. (c) Switch satellite C with E. (d) Switch satellite B with F. (e) Switch satellite A with G.

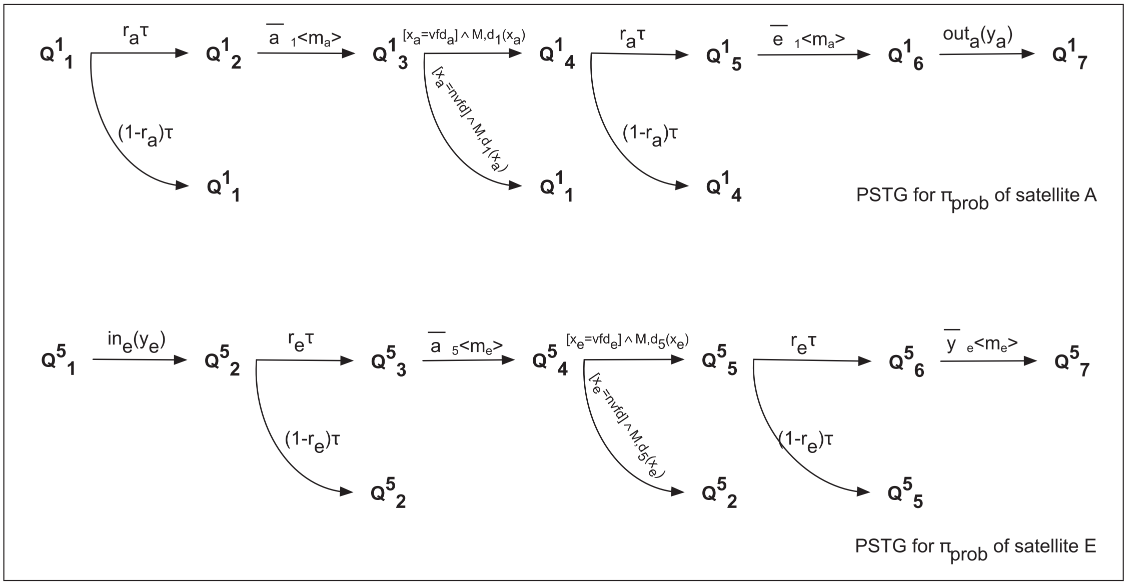


Fig. 11. PTSGs of π_{prob} process of satellites A and E.

4.1.3. π_{prob} models of the user segment

The user segment usually refers to the "GNSS receivers" that capture, process and track L-band signals from visible satellites to calculate the airplane's PVT (Section 2.3). The navigation mission of the flight was used to study the availability of navigation satellites to accomplish the mission during a specific segment of the flight. The 7 satellites were used for navigation during the flight. Due to the coverage limitation of satellites, the aircraft needs to switch to different satellites for navigation guidance during the flight. Fig. 10 gives the schema of the satellite navigation switching that occurred during the entire flight. As a result, there are 4 satellite groups available for navigation during the entire flight: $\{A, B, C, D\}$, $\{A, B, D, E\}$, $\{A, D, E, F\}$ and $\{D, E, F, G\}$.

There are two kinds of independent movement: the physical movement of satellites A, $B, \ldots, G$ and the aircraft Usr, and the virtual movement of communication links between them. Their combined physical movement gives rise to the virtual movement of the link between them.[2]

For mobility models, switching occurred between satellite pairs: C and E, B and F, and A and G. The switch from C to E occurs at 13:15, as shown in Fig. 10(c). The switch from B to F occurs at 13:55, as shown in Fig. 10(d). The switch from A to G occurs at 14:29, as shown in Fig. 10(e). In Fig. 10(c), the airplane sequentially uses satellite groups $\{A, B, C, D\}$ and $\{A, B, D, E\}$ for navigation. First, the aircraft uses satellites C, B, A and D; the linking channels between these 4 satellites and the airplane are $e1$, $e2$, $e3$ and $e4$. When the aircraft uses satellites B, A, D and E for navigation, E replaces C at the last stage and the channel of C is replaced by that of E. Fig. 10(d) shows the scenario when the aircraft changes from using satellite group $\{A, B, D, E\}$ to group $\{A, D, E, F\}$, and in Fig. 10 (e), the aircraft changes from using satellite group $\{A, D, E, F\}$ to group $\{D, E, F, G\}$. Similarly, when satellites $\{A, D, E, F\}$ or $\{D, E, F, G\}$ are used.

$$\begin{aligned} Usr &\triangleq e_1(z).Usr + e_2(z).Usr + e_3(z).Usr + e_4(z).Usr + e_5(z).Usr \\ &\quad + e_6(z).Usr + e_7(z).Usr \end{aligned}$$

$$Switch \triangleq \overline{out_c}\langle e_3\rangle.\overline{in_e}\langle e_3\rangle.\overline{out_b}\langle e_2\rangle.\overline{in_f}\langle e_2\rangle.\overline{out_a}\langle e_1\rangle.\overline{in_a}\langle e_1\rangle.\mathbf{0}$$

4.2. Translation from π_{prob} models to PRISM language

The π_{prob} processes must be translated to PRISM in order to perform probabilistic verification using the model checker. Translation from π_{prob} models of the satellite navigation system to their representation in PRISM follows the translation rules given in Section 3.3.1. We use the process P_A of satellite A to illustrate the procedure of the translation. The π_{prob} model of the communication between satellite A and the monitor station is

$$P_A \triangleq r_a\tau.\overline{a_1}\langle m_a\rangle.d_1(x_a).([x_a = v_a]P'_A + [x_a = no]P_A) \oplus (1 - r_a)\tau.P_A$$

$$P'_A \triangleq r_a\tau.\overline{e_1}\langle m_a\rangle.out_a(y_a).\mathbf{0} \oplus (1 - r_a)\tau.P'_A$$

Then, the process is converted into a graphical representation, namely a PSTG. For comparison, the converted PSTGs of processes P_A and P_E are both shown in Fig. 11.

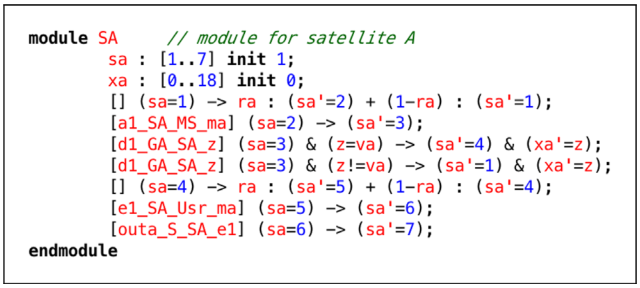
Finally, the PSTG of the system is translated into the PRISM modules according to the transition rules, and the corresponding module of π_{prob} model of satellite A can be derived, as shown in Fig. 12.

The translation of π_{prob} models of the remaining 6 satellites, the monitor station, the master control station, the ground antenna, the aircraft, and the mobility model can be derived similarly using the translation rules. The entire PRISM code is shown in Appendix.

We built a small, but detailed, model of the system. The satellite navigation systems exhibit both probabilistic behaviour (re-transmission due to unreliability of space segment and control segment) and nondeterministic behaviour (scheduling of transmission of satellites by control segment within the mission) and can be naturally modelled as an MDP. We translated a constructed process algebraic model based on the underlying reference model in PRISM. The state space for different number of satellites are shown in Table 2.

Because of the detailed nature of the model and the corresponding state space size, we first consider a small number of satellites (N=4, 5, 6, 7, 8). It is possible, though, that availability properties analysed in these small models will also be exhibited by a more large size (e.g., 17 satellites). With regards to the initial configuration of the model, we assume that the control segment and the user segment communicate with one satellite at a time.

[2] The links and their movement are obtained using the modelling, simulation, analysis, and operations software Satellite Tool Kit (STK).



```
module SA     // module for satellite A
        sa : [1..7] init 1;
        xa : [0..18] init 0;
        [] (sa=1) -> ra : (sa'=2) + (1-ra) : (sa'=1);
        [a1_SA_MS_ma] (sa=2) -> (sa'=3);
        [d1_GA_SA_z] (sa=3) & (z=va) -> (sa'=4) & (xa'=z);
        [d1_GA_SA_z] (sa=3) & (z!=va) -> (sa'=1) & (xa'=z);
        [] (sa=4) -> ra : (sa'=5) + (1-ra) : (sa'=4);
        [e1_SA_Usr_ma] (sa=5) -> (sa'=6);
        [outa_S_SA_e1] (sa=6) -> (sa'=7);
endmodule
```

Fig. 12. PRISM module of satellite A.

Table 2
State space for different number of satellites.

N	States	Transitions	Time (s)
4	153,824	750,368	1.999
5	331,120	1,625,059	9.074
6	501,290	2,466,627	18.153
7	659,252	3,249,969	33.051
8	724,230	3,554,991	61.741

This configuration is suitably realistic and ensures that the mobility is possible. For $N=7$, the model has 659,252 states and 3,249,969 transitions; for $N=8$, it has 724,230 states and 3,554,991 transitions. These MDPs are constructed by PRISM on a 2.4 GHz Mac with 8 GB RAM in 33.051 s and 61.741 s, respectively.

5. Verification

5.1. Availability parameters

Although the accuracy of satellite positioning in the aviation environment is in general sufficient, it is its availability that limits the system dependability and overall performance. Availability properties relate to the reliability and maintainability of GNSS. Traditionally, availability is the probability that the system is operating at a satisfactory level and can be committed at the start of a navigation mission when the mission is called for at an unknown and random point in time.

Availability depends on the reliability and maintainability. For repairable satellites, we usually use the term Mean Time between Failure (MTBF). MTBF is the average time from one failure to the next, including the repair time. Mean Time To Repair (MTTR) is the time taken to repair a failed satellite. System designers should aim to allow for a high MTTR value and still achieve the reliability requirements. Availability is a mathematical function of MTBF and MTTR. We assume that there is negligible delay before repair of a failed satellite begins. The availability factor can be computed using the following formula:

$$Availability = \frac{MTBF}{MTBF + MTTR} \tag{1}$$

Availability can range from 0% (never available) to 100% (always available). Clearly, satellite navigation systems that can offer high availability are more desirable than ones that offer lower availability. As a result, an availability requirement for a satellite navigation system is that it should achieve a satisfactory degree to which the system is in an operable state at any time.

During signal transmission from the monitor station to the master control station and from the master control station to the ground antenna, abnormal signal transmission may occur, resulting in errors in information and corresponding anomalies in the subsequent update information for the satellites. This can affect the navigation safety of users if the situation is severe. If anomalies occur in signal transmission, the master control station can correct the signal after a certain period of time. The reliability of space and control segments based on MTBF and MTTR is given in Table 3.

Table 3
Reliability of space and control segments.

Systems	MTBF (hours)	MTTR
Satellite	model	6 months
Monitor station	1,56,000	25.2 minutes
Master control station	1248	52.3 minutes
Ground antenna	2310	4.2 hours

Furthermore, we propose a modified concept for the GNSS availability properties associated with the underlying specification. The current approach involves the prediction of the "mean" availability over the system lifetime, assuming that the system is in a steady state. This approach is not suited to the specification of GNSS positioning systems, where the objective is to guarantee what can be obtained from the system during short periods of time that are meaningful to users, and that this short term availability will be maintained during the lifetime of the system. This requires a modification of the availability concept, as it is currently understood.

Based on a preliminary investigation, it is assumed in our analysis that the information exchange among the satellites, monitor station and ground antenna does not itself generate information anomalies, but its reliability is a direct consequence of the reliabilities of the satellites and ground antenna. It is additionally assumed that information anomalies can occur in the signal transmission between satellites, master control station, monitor station, and ground antenna, and environmental factors as well. These assumptions and related data are based on relevant reports[3] on GPS, as summarised in Table 4.

Where available, the data used for quantitative analysis in this study were collected from the official published data [17,18]. In other cases we used data for similar systems. The satellite models involved in the navigation satellite availability analysis of this section are Block-IIA, Block-IIR and Block-IIRM.

5.2. Probabilistic computation tree logic (PCTL)

We use PCTL to specify various availability properties. One of the most important operators in PCTL is the *P* operator, which is used to reason about the probability of an event's occurrence. It is often useful to compute the actual probability that some behaviour of a model is observed. Therefore, PRISM allows a variation of the *P* operator to be used in a query, i.e., $\mathbf{P}_{=?}[pathprop]$, which returns a numerical rather than a Boolean value.

In MDP models, there are two types of branching, nondeterministic, determined by a scheduler, and probabilistic, governed by the probability distribution. In order to interpret this, the properties in PCTL consider under any scheduling of processes, yielding the minimum or maximum over all the possible ways of resolving nondeterminism instead of the exact probability. Simple examples of such properties are "the maximum probability of an error occurring within *T* time steps": $\mathbf{P}_{max=?}[F \leq T \text{ "}error\text{"}]$; and "what is the worst-case expected time taken for a backup satellite to be launched?": $\mathbf{R}^{time}_{max=?}[\mathbf{F} \text{ "}launch\text{"}]$, where both "*error*" and "*launch*" are labels on system states specified in PRISM.

[3] Global Positioning System (GPS) Performance Quarterly Report.

Table 4
Transmission reliability of satellite navigation systems.

Systems	Transmission reliability
Satellite → MS	Reliability of satellites
MS → MCS	0.99999
MCS → GA	0.99999
GA → Satellite	0.99999
Environmental factors	0.9

PRISM includes the support for the specification and verification of properties based on costs and rewards. This means that PRISM can be used to reason, for example, about properties such as "expected time", "expected number of lost messages" or "expected energy consumption". The basic idea is that probabilistic models developed in PRISM can be augmented with costs (something bad) or rewards (something good): real values associated with certain states or transitions of the model (the costs and rewards are numerically identical). For MDPs, where time proceeds in discrete steps, the time interval is simply an integer upper bound. In our study, we use rewards "steps" to calculate expected time "*T*". Rewards are associated with models using the rewards... endrewards construct.

```
rewards ``time''
true : 1;
endrewards
```

5.3. Best and worst case availability

In this section, we perform quantitative analysis of satellite availability and channel availability of the satellite navigation system using PRISM respectively. Some typical examples of availability properties formalised with PCTL are given in Table 5.

We first study how the longest expected time of satellite navigation mission for aircraft varies over the execution of the mission. To consider the probability of some behaviour of our MDP, the nondeterministic choices need to be resolved first. PRISM provides us an exhaustive search and exact quantitative results of all possible behaviours of the system, including both best case and worst case scenarios. This is done using PCTL properties of the form: $\mathbf{R}_{\{''time''\}min=?}[F(s4=4)]$ and $\mathbf{R}_{\{''time''\}max=?}[F(s4=4)]$, which represent the minimum (best case) and maximum (worst case) expected value of time that is from the beginning until the end of the mission (at the time instant $F(s4=4)$). Since we have added a reward structure called "time" to the PRISM model, it associates with each state of the MDP a value representing the longest expected time between any two components at that point. The obtained result of the minimum and maximum expected time that depends on reliability of different components is depicted in Fig. 13.

We see that as reliability increases, the expected time decreases. In Fig. 13(a) for best case scenario, if the reliability of satellites is larger than 0.65, it will have less influence for satellites than different components of control segment (MS, MCS, and GA) on finishing the navigation mission. However, it is clear that the environmental factor has greatest influence on the total execution time. The reliability of environment is to what extent the environmental factors can jeopardise the transmission reliability between satellites and the control segment, so higher reliability means more reliable transmission. The default value of environmental factors is considered to be 0.9, but we can see that it has less influence on mission time when it is larger than 0.95. So, we should design the course of movement of satellites in the environment as gentle and stable (e.g., less solar radiation) as possible. From Fig. 13(b), we see that the curves of satellites and control segment are similar for worst case scenario. But, when the reliability of environment is larger than 0.9 (which is compared with 0.95 in minimum case), the environment influence on mission time can be neglected.

The properties $\mathbf{P}_{\min=?}[F \le T(sc=6)]$ and $\mathbf{P}_{\max=?}[F \le T(sc=6)]$ enable us to compute the minimum and maximum probability that satellite C finishes signal transmission with the aircraft within *T* time steps. The form of "$\le t$" or "$< t$" (where *t* is a PRISM expression evaluating to a constant, non-negative value) is the upper time bound.

As shown in Fig. 14, we have $t=T$ in our case, where *T* is a constant value between 0 and 100. We see that the probability increases as *T* increases after the probability equals to 0 and before it reaches to 1. For both cases, the satellite C eventually sent signal to the aircraft. However, for the minimum case, it takes much less time for the probability to reach 0.9 from 0 (about 10 time steps), compared to the maximum case (about 55 time units). We should be aware of that in realistic cases, the probability distribution is between the two of them.

The minimum availability of satellite C varies on time, and it can be derived by the following formula, where $\mathbf{R}_{\{''time''\}min=?}[F\ (sc=5)] - \mathbf{R}_{\{''time''\}min=?}[F(sc=4)])$ is the unavailable time of satellite C in the minimum case.

$$Avail^{C}_{\min}(T) = \begin{cases} 1 & \text{for} \quad 0 \le T < \mathbf{R}_{\{''time''\}\min=?}[F(sc=4)] \\ \dfrac{\mathbf{R}_{\{``time"\}\min=?}[\mathbf{F(sc=4)}]}{T} & \text{for} \quad \mathbf{R}_{\{``time"\}\min=?}[F(sc=4)] \le T \le \mathbf{R}_{\{``time"\}\min=?}[F(sc=5)] \\ \dfrac{T-(\mathbf{R}_{\{``time"\}\min=?}[F(sc=5)]-\mathbf{R}_{\{``time"\}\min=?}[F(sc=4)])}{T} & \text{for} \quad \mathbf{R}_{\{``time"\}\min=?}[F(sc=5)] < T \le \mathbf{R}_{\{``time"\}\min=?}[F(sc=7)] \end{cases}$$

The maximum availability of satellite C and minimum/maximum availability of all other satellites can be derived similarly. The results are illustrated in Figs. 15 and 16 respectively.

From above figures, we see that satellites of the same model have the same availability distribution. For instance, both the minimum and maximum probability of satellites *A*, *C*, and *D* are all same, except that their individual online duration are difference. They are online at the same time, but satellite *D* has a long tail than satellites *A* and *C*, and satellite *A* has longer tail than *C*. This is due to the fact that satellites *D* is never offline (never being switched) during the mission later than both *A* and *C*, and *A* gets offline (switching to *G*) later than *C*. Model Block IIRM (Satellites *A*, *C*, *D*, and *E*) had the largest satellite availability for navigation, followed by Block IIR (satellite *F*) and then Block IIA (satellites *B* and *G*). The availability curve indicates that the satellite online and offline time instant and the duration of use of a satellite do not have very significant impact on the satellite's availability. Rather, the factor that had the greatest effect on navigation was the design life and reliability of the navigation satellites.

Similar to the definition of satellite availability, we define channel availability. The minimum availability of channel e3 also varies on time, and it can be derived by the following formula, where $\mathbf{R}_{\{\text{“}time\text{”}\}\min=?}[F(s5=3)]-\mathbf{R}_{\{\text{“}time\text{“}\}\min=?}[F(s5=2)])$ is the unavailable time of channel $e3$ in the minimum case.

$$Avail_{min}^{e_3}(T)=\begin{cases}1 & \text{for}\quad 0\le T<\mathbf{R}_{\{\text{“}time\text{”}\}\min=?}[F(s5=2)]\\ \frac{\mathbf{R}_{\{\text{“}time\text{”}\}\min=?}[\mathbf{F(s5=2)}]}{T} & \text{for}\quad \mathbf{R}_{\{\text{“}time\text{”}\}\min=?}[F(s5=2)]\le T\le\mathbf{R}_{\{\text{“}time\text{”}\}min=?}[F(s5=3)]\\ \frac{T-(\mathbf{R}_{\{\text{"}time\text{"}\}\min=?}[F(s5=3)]-\mathbf{R}_{\{\text{“}time\text{”}\}\min=?}[F(s5=2)])}{T} & \text{for}\quad \mathbf{R}_{\{\text{“}time\text{”}\}\min=?}[F(s5=3)]<T\le\mathbf{R}_{\{\text{“}time\text{”}\}min=?}[F(s5=7)]\end{cases}$$

The maximum availability of channel $e3$ and minimum/maximum availability of all other channels can be derived similarly. The results are illustrated in Fig. 17. The backup satellite of a channel were not considered in this study. Neglecting backup satellite may cause the channel availability to be slightly greater than when it is considered. An actual mission will involve multiple satellites, and each channel has multiple backup satellites. Thus, once a failure occurs, the channel will be switched to a backup satellite.

Other than using high cost backup satellite for a navigation, a way to improve the channel availability is the addition of new signals. These signals complement the existing signal for navigation service. This additional signal will make GNSS a more robust navigation system for various aviation applications. Thus, the availability of additional signals means that errors that occur in the signals due to disturbances in the ionosphere can be significantly reduced through the simultaneous use of more signals. This will improve the overall system reliability, to increase accuracy and availability, and will allow a robust approach with little or no ground infrastructure.

Therefore, the availability of navigation satellites in the actual process is greater than this value. In general, the impact of environmental factors is small, and thus the availability of satellites for navigation could be larger than 98.5%. Moreover, the presence of multiple satellites will potentially increase the overall availability along an air line, but the increase of available satellites does not necessarily guarantee an improved user-satellites geometry due to the similar orbital arrangement of most GNSS satellites.

The SPS SIS availability is the probability that the slots in the GPS constellation will be occupied by satellites transmitting a trackable and healthy SPS SIS. For this SPS Performance Standard, there are two components of availability as follows: (1) per-slot availability: the fraction of time that a slot in the GPS constellation will be occupied by a satellite that is transmitting a trackable and healthy SPS SIS; (2) constellation availability: the fraction of time that a specified number of slots in the GPS constellation.

Table 5
Summary of PRISM properties used in the paper.

Name	PRISM notation	Meaning
“*avail. satellite*”	$\mathbf{P}_{min\ge1}[F(sc=7)]$	Whether satellite C is available during the navigation?
”min. *avail. satellite*”	$\mathbf{R}_{min=?}[F(sc=6)]$	The minimum available time of satellite C
“max. *avail. satellite*”	$\mathbf{R}_{max=?}[F(s4=4)]$	The maximum expected time of navigation mission
“*min. unavail. channel*”	$\mathbf{R}_{min=?}[F(s5=3)]-\mathbf{R}_{min=?}[F(s5=2)]$	The minimum unavailable time of channel e3
“*max. unavail. channel*”	$\mathbf{R}_{max=?}[F(s5=6)]-\mathbf{R}_{max=?}[F(s5=5)]$	The maximum unavailable time of channel e1
“*min. avail. time bound satellite*”	$\mathbf{P}_{min=?}[F\le T(sc=6)]$	The minimum probability that C done transmission with U within T
“*max. avail. time bound satellite*”	$\mathbf{P}_{max=?}[F\le T(se=7)]$	The maximum probability that E done transmission with U within T

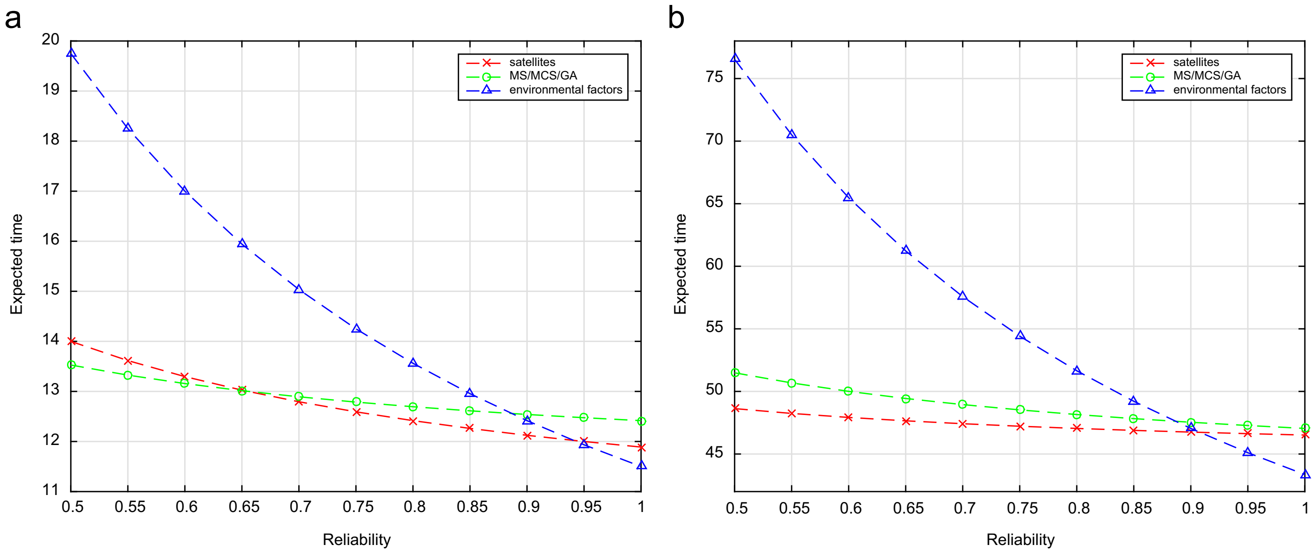


Fig. 13. Expected time results for different reliability of components. (a) Minimum expected time. (b) Maximum expected time.

SPS SIS availability is assessed through analysis of the broadcast navigation messages. To evaluate the usefulness of our results for SPS SIS availability, we referred to some official reports from the civil aviation sector. The U.S. Federal Aviation Administration (FAA) releases quarterly reports on the performance analysis of the system based on the operation of the GPS in each quarter to ensure the navigation safety of global aviation. According to the monitoring reports released by the FAA [19] for the period of 1 January 2013 to 31 March 2013, the average service availability of each individual GPS satellite is approximately 99%, and the worst-case service availability is approximately 90%. These numbers approximately lie in that obtained in our study, which are between 88% and 98% for minimum availability and between 97.5% and 98.5% for maximum availability. This supports, from one line of evidence, the feasibility and applicability of our approach.

6. Related work

Prediction of satellite navigation availability is very useful for numerous applications such as airplane navigation missions and in-car navigation systems. Simulation is nowadays widely used to analyse performance and predicate availability for a variety of satellite systems [20,21]. In [21], software simulation based on a Markov model of a GPS constellation of 24 satellites is used to obtain availability estimates of GNSS in Taiwan. The primary input data for the availability model is the MTBF and failure rate of the GPS satellites.

In [22], an automated method for predicting the number of satellites available to a GPS receiver, at any point on the Earth's surface at any time, is described. Availability analysis between a GPS receiver and each potentially visible GPS satellite is performed using a number of different surface models and satellite orbit calculations. In [23], the availability of an NCSS is studied to examine the feasibility of using an NCSS constellation in Australia. A performance model was proposed in [24] to evaluate the availability of satellite systems over geographic grid averaging areas over a given period of time. The corresponding cost model and performance model are designed in such a way as to minimise cost and maximise performance of the systems.

In [25], a method for determining the availability of three different GPS services (positioning, supplemental navigation, and

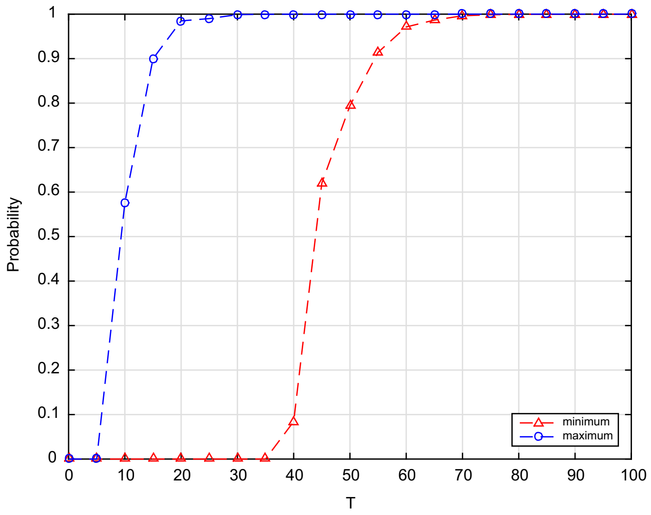


Fig. 14. Probability of satellite C within time T.

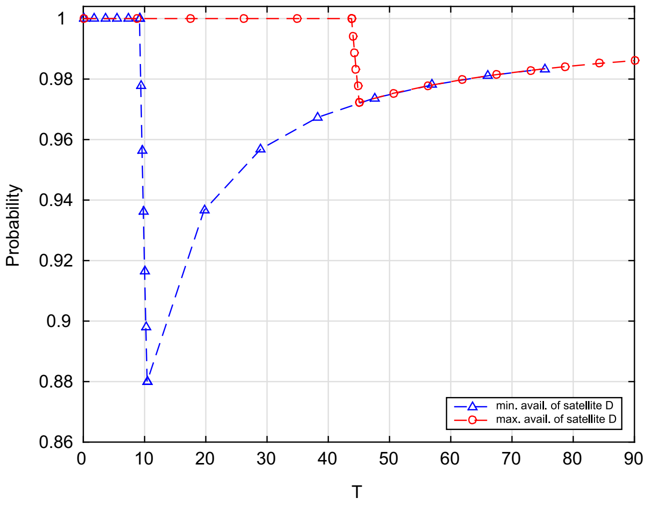


Fig. 16. Availability of satellite D.

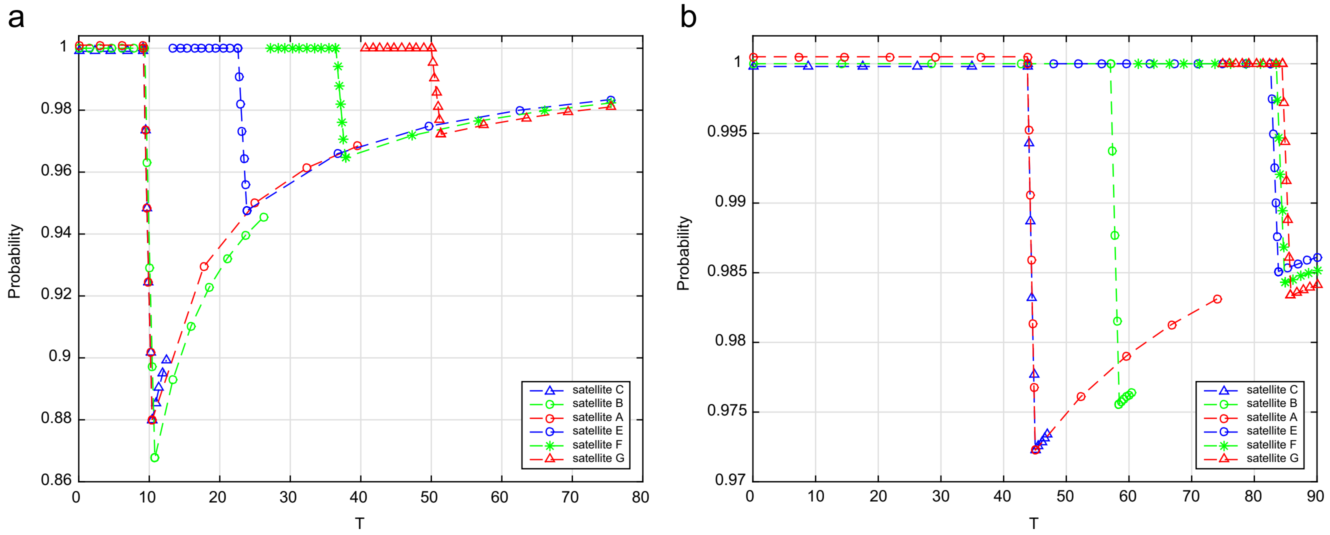


Fig. 15. Minimum and maximum availability of satellites. (a) Minimum availability. (b) Maximum availability.

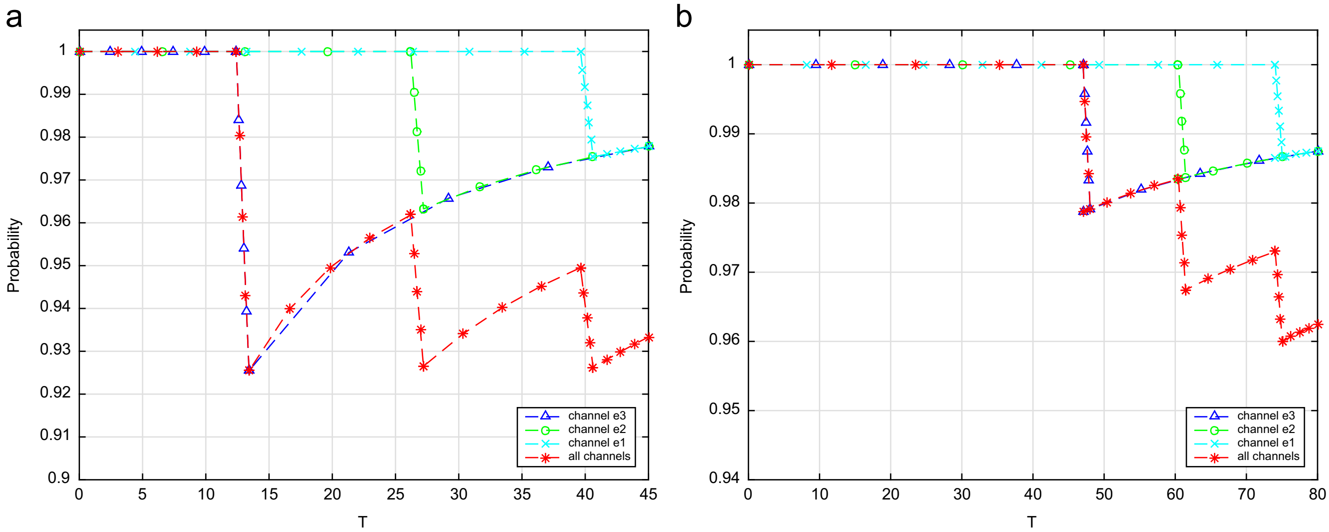


Fig. 17. Minimum and maximum availability of channels. (a) Minimum availability. (b) Maximum availability.

sole means navigation) is described for both two-dimensional and three-dimensional applications. A 21-satellite and a 24-satellite constellation are considered. In the companion paper [26], state probability analyses of 21- and 24-satellite constellations based on a Markov chain model are discussed. Availability characteristics for GPS and GPS augmented by geostationary satellites (GSs) are compared in [27]. Availability is determined for users in the contiguous zone in the United States, based on the planned operational GPS constellation and various GS deployments.

Formal methods have significantly impacted the aerospace systems engineering, and have successfully applied to the verification and validation of many aspects of spacecraft and satellite systems. A small aircraft transportation system by considering a number of approaching aircrafts has been formally modelled and its safety properties have been analysed in [28] using the interactive theorem prover PVS. Based on the Ada source code form, a mission critical satellite software control system is modelled using the input language of the symbolic model checker NuSMV 2, and its required behaviours has been specified as temporal logic properties [29]. In a series of work reported in [30], authors have developed the COMPASS toolset that utilises both qualitative and probabilistic model checking techniques. They have modelled a satellite platform in system level with AADL modelling language and analysed its reliability aspects. Our preliminary research into the verification of satellite systems, in which we restrict our analysis only to a single satellite and a satellite constellation but not a navigation mission for aviation, is presented in [31]. This paper is an extended version of our work in [32].

7. Conclusions and future work

In this paper, we have shown that probabilistic verification can be used to analyse interesting and important reliability and availability properties of GNSS based positioning systems that would be difficult to discover using alternative analysis techniques such as simulation. We have demonstrated the successful application of probabilistic model checking to the analysis of reliability and availability properties that relate to the dependability and overall performance of the underlying system of satellite navigation for aviation. To do this, we have guide the reader through the theory of Markov decision processes (MDPs), process algebras, and probabilistic model checking.

Although using probabilistic model checking limits the number of satellites in the navigation system that can be analysed, these representative systems can highlight interesting behaviour that may also occur in more realistic configurations for aviation navigation. We have modelled essential aspects (e.g., unreliable signal transmission, component movement, concurrency, nondeterminism) of satellite system for navigating a specific flight. The results we have obtained demonstrate that modelling unknown choices with randomness causes a variation on the mission execution time and availability: the actual scenario may be different from the best or worst scenarios. The introduction of nondeterminism shows that these measures can take a range of values. As a result, we compute minimum and maximum values, representing both the best case and worst case of mission execution time and satellite and channel availability under any scheduling of simultaneous transmission between different satellites and control segment.

Although nowadays satellite positioning is commonly used in the aviation sector, it is still to gain a foothold in other industries such as the rail industry. One major barrier that presents its application to railway safety is the lack of evidence that the concept and theory for the verification of railway applications with the introduction of GNSS is applicable based on the joint use of aviation and railway standards and requirements. Up to now availability analysis is non-trivial because difficult situations exist on the railways due to the limitations of the GNSS coverage in urban canyons, tunnels, and forest areas. For future work, we plan to add a fourth environment segment that simulates such difficult situations to the GNSS.

Acknowledgements

This research was partially supported by the European Commission (EC) under the EATS project: European Train Control System (ETCS) Advanced Testing and Smart Train Positioning System (FP7-TRANSPORT-314219). The author Yu Lu was funded by the Scottish Informatics and Computer Science Alliance (SICSA).

Appendix

```
mdp

const double ra=0.8; const double rb=0.7; const double rc=0.8; const double rd=0.8;
const double re=0.8; const double rf=0.75; const double rg=0.7;
const double rms=0.99999; const double rmcs=0.99999;
const double pef=0.9; const double rga=0.99999;
const int ma=1; const int mb=2; const int mc=3; const int md=4;
const int me=5; const int mf=6; const int mg=7;
const int va=8; const int vb=9; const int vc=10; const int vd=11;
const int ve=12; const int vf=13; const int vg=14; const int no=15;
const int e1=16; const int e2=17; const int e3=18;

module SC  // module for satellite C
        sc : [1..7] init 1;
        xc : [0..18] init 0;
        [] (sc=1) -> rc : (sc'=2) + (1-rc) : (sc'=1);
        [a3_SC_MS_mc] (sc=2) -> (sc'=3);
        [d3_GA_SC_z] (sc=3) & (z=vc) -> (sc'=4) & (xc'=z);
        [d3_GA_SC_z] (sc=3) & (z!=vc) -> (sc'=1) & (xc'=z);
        [] (sc=4) -> rc : (sc'=5) + (1-rc) : (sc'=4);
        [e3_SC_Usr_mc] (sc=5) -> (sc'=6);
        [outc_S_SC_e3] (sc=6) -> (sc'=7);
endmodule
   // add further processes through renaming
   // module for satellite B
   module SB = SC[ sc=sb, xc=xb, rc=rb, vc=vb, a3_SC_MS_mc=a2_SB_MS_mb,
                   d3_GA_SC_z=d2_GA_SB_z, e3_SC_Usr_mc=e2_SB_Usr_mb,
                   outc_S_SC_e3=outb_S_SB_e2 ] endmodule
   // module for satellite A
   module SA = SC[ sc=sa, xc=xa, rc=ra, vc=va, a3_SC_MS_mc=a1_SA_MS_ma,
                   d3_GA_SC_z=d1_GA_SA_z, e3_SC_Usr_mc=e1_SA_Usr_ma,
                   outc_S_SC_e3=outa_S_SA_e1 ] endmodule
```

```
   module SD  // module for satellite D
           sd : [1..6] init 1;
           xd : [0..18] init 0;
           [] (sd=1) -> rd : (sd'=2) + (1-rd) : (sd'=1);
           [a4_SD_MS_md] (sd=2) -> (sd'=3);
           [d4_GA_SD_z] (sd=3) & (z=vd) -> (sd'=4) & (xd'=z);
           [d4_GA_SD_z] (sd=3) & (z!=vd) -> (sd'=1) & (xd'=z);
           [] (sd=4) -> rd : (sd'=5) + (1-rd) : (sd'=4);
           [e4_SD_Usr_md] (sd=5) -> (sd'=6);
   endmodule
```

```
// module for satellite E
module SE
        se : [1..7] init 1;
        xe : [0..18] init 0;
        ye : [0..18] init 0;
        [ine_S_SE_e3] (se=1) -> (se'=2) & (ye'=e3);
        [] (se=2) -> re : (se'=3) + (1-re) : (se'=2);
        [a5_SE_MS_me] (se=3) -> (se'=4);
        [d5_GA_SE_z] (se=4) & (z=ve) -> (se'=5) & (xe'=z);
        [d5_GA_SE_z] (se=4) & (z!=ve) -> (se'=2) & (xe'=z);
        [] (se=5) -> re : (se'=6) + (1-re) : (se'=5);
        [ye_SE_Usr_me] (se=6) -> (se'=7);
endmodule
```

```
   // module for satellite F
   module SF = SE[ se=sf, xe=xf, ye=yf, re=rf, ve=vf, e3=e2,
                  ine_S_SE_e3=inf_S_SF_e2, a5_SE_MS_me=a6_SF_MS_mf,
                  d5_GA_SE_z=d6_GA_SF_z, ye_SE_Usr_me=yf_SF_Usr_mf ] endmodule
   // module for satellite G
   module SG = SE[ se=sg, xe=xg, ye=yg, re=rg, ve=vg, e3=e1,
                  ine_S_SE_e3=ing_S_SG_e1, a5_SE_MS_me=a7_SG_MS_mg,
                  d5_GA_SE_z=d7_GA_SG_z, ye_SE_Usr_me=yg_SG_Usr_mg ] endmodule
```

```
module MS  // module for the monitor station MS
        s1 : [1..15] init 1;
        x   : [0..18] init 0;
        [a1_SA_MS_ma] (s1=1) -> (s1’=2) & (x’=ma);
        [a2_SB_MS_mb] (s1=1) -> (s1’=3) & (x’=mb);
        [a3_SC_MS_mc] (s1=1) -> (s1’=4) & (x’=mc);
        [a4_SD_MS_md] (s1=1) -> (s1’=5) & (x’=md);
        [a5_SE_MS_me] (s1=1) -> (s1’=6) & (x’=me);
        [a6_SF_MS_mf] (s1=1) -> (s1’=7) & (x’=mf);
        [a7_SG_MS_mg] (s1=1) -> (s1’=8) & (x’=mg);
        [] (s1=2) -> rms : (s1’=9) + (1-rms) : (s1’=2);
        [] (s1=3) -> rms : (s1’=10) + (1-rms) : (s1’=3);
        [] (s1=4) -> rms : (s1’=11) + (1-rms) : (s1’=4);
        [] (s1=5) -> rms : (s1’=12) + (1-rms) : (s1’=5);
        [] (s1=6) -> rms : (s1’=13) + (1-rms) : (s1’=6);
        [] (s1=7) -> rms : (s1’=14) + (1-rms) : (s1’=7);
        [] (s1=8) -> rms : (s1’=15) + (1-rms) : (s1’=8);
        [b_MS_MCS_x] (s1=9) -> (s1’=1);
        [b_MS_MCS_x] (s1=10) -> (s1’=1);
        [b_MS_MCS_x] (s1=11) -> (s1’=1);
        [b_MS_MCS_x] (s1=12) -> (s1’=1);
        [b_MS_MCS_x] (s1=13) -> (s1’=1);
        [b_MS_MCS_x] (s1=14) -> (s1’=1);
        [b_MS_MCS_x] (s1=15) -> (s1’=1);
endmodule
module MCS  // module for the master control station MCS
        s2 : [1..22] init 1;
        y : [0..18] init 0;
        vj : [0..18] init 0;
        nj : [0..18] init 0;
        [b_MS_MCS_x] (s2=1) & (x=ma) -> (s2’=2) & (y’=x);
        [b_MS_MCS_x] (s2=1) & (x=mb) -> (s2’=3) & (y’=x);
        [b_MS_MCS_x] (s2=1) & (x=mc) -> (s2’=4) & (y’=x);
        [b_MS_MCS_x] (s2=1) & (x=md) -> (s2’=5) & (y’=x);
        [b_MS_MCS_x] (s2=1) & (x=me) -> (s2’=6) & (y’=x);
        [b_MS_MCS_x] (s2=1) & (x=mf) -> (s2’=7) & (y’=x);
        [b_MS_MCS_x] (s2=1) & (x=mg) -> (s2’=8) & (y’=x);
        [] (s2=2) -> rmcs*pef : (s2’=9) + rmcs*(1-pef) : (s2’=10) + (1-rmcs) : (s2’=2);
        [] (s2=3) -> rmcs*pef : (s2’=11) + rmcs*(1-pef):(s2’=12) + (1-rmcs):(s2’=3);
        [] (s2=4) -> rmcs*pef : (s2’=13) + rmcs*(1-pef):(s2’=14) + (1-rmcs):(s2’=4);
        [] (s2=5) -> rmcs*pef : (s2’=15) + rmcs*(1-pef):(s2’=16) + (1-rmcs):(s2’=5);
        [] (s2=6) -> rmcs*pef : (s2’=17) + rmcs*(1-pef):(s2’=18) + (1-rmcs):(s2’=6);
        [] (s2=7) -> rmcs*pef : (s2’=19) + rmcs*(1-pef):(s2’=20) + (1-rmcs):(s2’=7);
        [] (s2=8) -> rmcs*pef : (s2’=21) + rmcs*(1-pef):(s2’=22) + (1-rmcs):(s2’=8);
        [c_MCS_GA_va] (s2=9) -> (s2’=1); [c_MCS_GA_na] (s2=10) -> (s2’=1);
        [c_MCS_GA_vb] (s2=11) -> (s2’=1); [c_MCS_GA_nb] (s2=12) -> (s2’=1);
        [c_MCS_GA_vc] (s2=13) -> (s2’=1); [c_MCS_GA_nc] (s2=14) -> (s2’=1);
        [c_MCS_GA_vd] (s2=15) -> (s2’=1); [c_MCS_GA_nd] (s2=16) -> (s2’=1);
        [c_MCS_GA_ve] (s2=17) -> (s2’=1); [c_MCS_GA_ne] (s2=18) -> (s2’=1);
        [c_MCS_GA_vf] (s2=19) -> (s2’=1); [c_MCS_GA_nf] (s2=20) -> (s2’=1);
        [c_MCS_GA_vg] (s2=21) -> (s2’=1); [c_MCS_GA_ng] (s2=22) -> (s2’=1);
endmodule
```

```
module GA  // module for the ground antenna GA
        s3 : [1..15] init 1;
        z : [0..18] init 0;
        [c_MCS_GA_va] (s3=1) -> (s3’=2) & (z’=va);
        [c_MCS_GA_na] (s3=1) -> (s3’=2) & (z’=no);
        [c_MCS_GA_vb] (s3=1) -> (s3’=3) & (z’=vb);
        [c_MCS_GA_nb] (s3=1) -> (s3’=3) & (z’=no);
        [c_MCS_GA_vc] (s3=1) -> (s3’=4) & (z’=vc);
        [c_MCS_GA_nc] (s3=1) -> (s3’=4) & (z’=no);
        [c_MCS_GA_vd] (s3=1) -> (s3’=5) & (z’=vd);
        [c_MCS_GA_nd] (s3=1) -> (s3’=5) & (z’=no);
        [c_MCS_GA_ve] (s3=1) -> (s3’=6) & (z’=ve);
        [c_MCS_GA_ne] (s3=1) -> (s3’=6) & (z’=no);
        [c_MCS_GA_vf] (s3=1) -> (s3’=7) & (z’=vf);
        [c_MCS_GA_nf] (s3=1) -> (s3’=7) & (z’=no);
        [c_MCS_GA_vg] (s3=1) -> (s3’=8) & (z’=vg);
        [c_MCS_GA_ng] (s3=1) -> (s3’=8) & (z’=no);
        [] (s3=2) -> rga : (s3’=9) + (1-rga) : (s3’=2);
        [] (s3=3) -> rga : (s3’=10) + (1-rga) : (s3’=3);
        [] (s3=4) -> rga : (s3’=11) + (1-rga) : (s3’=4);
        [] (s3=5) -> rga : (s3’=12) + (1-rga) : (s3’=5);
        [] (s3=6) -> rga : (s3’=13) + (1-rga) : (s3’=6);
        [] (s3=7) -> rga : (s3’=14) + (1-rga) : (s3’=7);
        [] (s3=8) -> rga : (s3’=15) + (1-rga) : (s3’=8);
        [d1_GA_SA_z] (s3=9) -> (s3’=1);
        [d2_GA_SB_z] (s3=10) -> (s3’=1);
        [d3_GA_SC_z] (s3=11) -> (s3’=1);
        [d4_GA_SD_z] (s3=12) -> (s3’=1);
        [d5_GA_SE_z] (s3=13) -> (s3’=1);
        [d6_GA_SF_z] (s3=14) -> (s3’=1);
        [d7_GA_SG_z] (s3=15) -> (s3’=1);
endmodule
module User  // module for the aircraft (User segment) U
        s4 : [1..8] init 1;
        [e1_SA_Usr_ma] (s4=1) -> (s4’=1);
        [e2_SB_Usr_mb] (s4=1) -> (s4’=1);
        [e3_SC_Usr_mc] (s4=1) -> (s4’=1);
        [e4_SD_Usr_md] (s4=1) -> (s4’=1);
        [ye_SE_Usr_me] (s4=1) -> (s4’=2);
        [yf_SF_Usr_mf] (s4=2) -> (s4’=3);
        [yg_SG_Usr_mg] (s4=3) -> (s4’=4);
endmodule
// rewards (to calculate expected number of steps)
rewards "steps"
true : 1;
endrewardsdule SD  // module for satellite D
sd : [1..6] init 1;
        xd : [0..18] init 0;
[] (sd=1) -> rd : (sd’=2) + (1-rd) : (sd’=1);
[a4_SD_MS_md] (sd=2) -> (sd’=3);
        [d4_GA_SD_z] (sd=3) & (z=vd) -> (sd’=4) & (xd’=z);
        [d4_GA_SD_z] (sd=3) & (z!=vd) -> (sd’=1) & (xd’=z);
        [] (sd=4) -> rd : (sd’=5) + (1-rd) : (sd’=4);
        [e4_SD_Usr_md] (sd=5) -> (sd’=6);
endmodule
```

```
module Switch  // module for the mobility model
        s5 : [1..7] init 1;
        [outc_S_SC_e3] (s5=1) -> (s5’=2);
        [ine_S_SE_e3] (s5=2) -> (s5’=3);
        [outb_S_SB_e2] (s5=3) -> (s5’=4);
        [inf_S_SF_e2] (s5=4) -> (s5’=5);
        [outa_S_SA_e1] (s5=5) -> (s5’=6);
        [ing_S_SG_e1] (s5=6) -> (s5’=7);
endmodule
// module for satellite E
module SE
        se : [1..7] init 1;
        xe : [0..18] init 0;
        ye : [0..18] init 0;
        [ine_S_SE_e3] (se=1) -> (se’=2) & (ye’=e3);
        [] (se=2) -> re : (se’=3) + (1-re) : (se’=2);
        [a5_SE_MS_me] (se=3) -> (se’=4);
        [d5_GA_SE_z] (se=4) & (z=ve) -> (se’=5) & (xe’=z);
        [d5_GA_SE_z] (se=4) & (z!=ve) -> (se’=2) & (xe’=z);
        [] (se=5) -> re : (se’=6) + (1-re) : (se’=5);
        [ye_SE_Usr_me] (se=6) -> (se’=7);
endmodule
```

```
// module for satellite F
module SF = SE[ se=sf, xe=xf, ye=yf, re=rf, ve=vf, e3=e2, ine_S_SE_e3=inf_S_SF_e2,
       a5_SE_MS_me=a6_SF_MS_mf, d5_GA_SE_z=d6_GA_SF_z, ye_SE_Usr_me=yf_SF_Usr_mf ] endmodule
// module for satellite G
module SG = SE[ se=sg, xe=xg, ye=yg, re=rg, ve=vg, e3=e1, ine_S_SE_e3=ing_S_SG_e1,
       a5_SE_MS_me=a7_SG_MS_mg, d5_GA_SE_z=d7_GA_SG_z, ye_SE_Usr_me=yg_SG_Usr_mg ] endmodule
```

```
module MS  // module for the monitor station MS
        s1 : [1..15] init 1;
        x   : [0..18] init 0;
        [a1_SA_MS_ma] (s1=1) -> (s1’=2) & (x’=ma);
        [a2_SB_MS_mb] (s1=1) -> (s1’=3) & (x’=mb);
        [a3_SC_MS_mc] (s1=1) -> (s1’=4) & (x’=mc);
        [a4_SD_MS_md] (s1=1) -> (s1’=5) & (x’=md);
        [a5_SE_MS_me] (s1=1) -> (s1’=6) & (x’=me);
        [a6_SF_MS_mf] (s1=1) -> (s1’=7) & (x’=mf);
        [a7_SG_MS_mg] (s1=1) -> (s1’=8) & (x’=mg);
        [] (s1=2) -> rms : (s1’=9) + (1-rms) : (s1’=2);
        [] (s1=3) -> rms : (s1’=10) + (1-rms) : (s1’=3);
        [] (s1=4) -> rms : (s1’=11) + (1-rms) : (s1’=4);
        [] (s1=5) -> rms : (s1’=12) + (1-rms) : (s1’=5);
        [] (s1=6) -> rms : (s1’=13) + (1-rms) : (s1’=6);
        [] (s1=7) -> rms : (s1’=14) + (1-rms) : (s1’=7);
        [] (s1=8) -> rms : (s1’=15) + (1-rms) : (s1’=8);
        [b_MS_MCS_x] (s1=9) -> (s1’=1);
        [b_MS_MCS_x] (s1=10) -> (s1’=1);
        [b_MS_MCS_x] (s1=11) -> (s1’=1);
        [b_MS_MCS_x] (s1=12) -> (s1’=1);
        [b_MS_MCS_x] (s1=13) -> (s1’=1);
        [b_MS_MCS_x] (s1=14) -> (s1’=1);
        [b_MS_MCS_x] (s1=15) -> (s1’=1);
endmodule
module MCS  // module for the master control station MCS
        s2 : [1..22] init 1;
        y : [0..18] init 0;
        vj : [0..18] init 0;
        nj : [0..18] init 0;
        [b_MS_MCS_x] (s2=1) & (x=ma) -> (s2’=2) & (y’=x);
        [b_MS_MCS_x] (s2=1) & (x=mb) -> (s2’=3) & (y’=x);
        [b_MS_MCS_x] (s2=1) & (x=mc) -> (s2’=4) & (y’=x);
        [b_MS_MCS_x] (s2=1) & (x=md) -> (s2’=5) & (y’=x);
        [b_MS_MCS_x] (s2=1) & (x=me) -> (s2’=6) & (y’=x);
        [b_MS_MCS_x] (s2=1) & (x=mf) -> (s2’=7) & (y’=x);
        [b_MS_MCS_x] (s2=1) & (x=mg) -> (s2’=8) & (y’=x);
        [] (s2=2) -> rmcs*pef : (s2’=9) + rmcs∗(1-pef) : (s2’=10) + (1-rmcs) : (s2’=2);
        [] (s2=3) -> rmcs*pef : (s2’=11) + rmcs∗(1-pef):(s2’=12) + (1-rmcs):(s2’=3);
        [] (s2=4) -> rmcs*pef : (s2’=13) + rmcs∗(1-pef):(s2’=14) + (1-rmcs):(s2’=4);
        [] (s2=5) -> rmcs*pef : (s2’=15) + rmcs∗(1-pef):(s2’=16) + (1-rmcs):(s2’=5);
        [] (s2=6) -> rmcs*pef : (s2’=17) + rmcs∗(1-pef):(s2’=18) + (1-rmcs):(s2’=6);
        [] (s2=7) -> rmcs*pef : (s2’=19) + rmcs∗(1-pef):(s2’=20) + (1-rmcs):(s2’=7);
        [] (s2=8) -> rmcs*pef : (s2’=21) + rmcs∗(1-pef):(s2’=22) + (1-rmcs):(s2’=8);
        [c_MCS_GA_va] (s2=9) -> (s2’=1); [c_MCS_GA_na] (s2=10) -> (s2’=1);
        [c_MCS_GA_vb] (s2=11) -> (s2’=1); [c_MCS_GA_nb] (s2=12) -> (s2’=1);
        [c_MCS_GA_vc] (s2=13) -> (s2’=1); [c_MCS_GA_nc] (s2=14) -> (s2’=1);
        [c_MCS_GA_vd] (s2=15) -> (s2’=1); [c_MCS_GA_nd] (s2=16) -> (s2’=1);
        [c_MCS_GA_ve] (s2=17) -> (s2’=1); [c_MCS_GA_ne] (s2=18) -> (s2’=1);
        [c_MCS_GA_vf] (s2=19) -> (s2’=1); [c_MCS_GA_nf] (s2=20) -> (s2’=1);
        [c_MCS_GA_vg] (s2=21) -> (s2’=1); [c_MCS_GA_ng] (s2=22) -> (s2’=1);
endmodule
```

```
module GA  // module for the ground antenna GA
        s3 : [1..15] init 1;
        z : [0..18] init 0;
        [c_MCS_GA_va] (s3=1) -> (s3’=2) & (z’=va);
        [c_MCS_GA_na] (s3=1) -> (s3’=2) & (z’=no);
        [c_MCS_GA_vb] (s3=1) -> (s3’=3) & (z’=vb);
        [c_MCS_GA_nb] (s3=1) -> (s3’=3) & (z’=no);
        [c_MCS_GA_vc] (s3=1) -> (s3’=4) & (z’=vc);
        [c_MCS_GA_nc] (s3=1) -> (s3’=4) & (z’=no);
        [c_MCS_GA_vd] (s3=1) -> (s3’=5) & (z’=vd);
        [c_MCS_GA_nd] (s3=1) -> (s3’=5) & (z’=no);
        [c_MCS_GA_ve] (s3=1) -> (s3’=6) & (z’=ve);
        [c_MCS_GA_ne] (s3=1) -> (s3’=6) & (z’=no);
        [c_MCS_GA_vf] (s3=1) -> (s3’=7) & (z’=vf);
        [c_MCS_GA_nf] (s3=1) -> (s3’=7) & (z’=no);
        [c_MCS_GA_vg] (s3=1) -> (s3’=8) & (z’=vg);
        [c_MCS_GA_ng] (s3=1) -> (s3’=8) & (z’=no);
        [] (s3=2) -> rga : (s3’=9) + (1-rga) : (s3’=2);
        [] (s3=3) -> rga : (s3’=10) + (1-rga) : (s3’=3);
        [] (s3=4) -> rga : (s3’=11) + (1-rga) : (s3’=4);
        [] (s3=5) -> rga : (s3’=12) + (1-rga) : (s3’=5);
        [] (s3=6) -> rga : (s3’=13) + (1-rga) : (s3’=6);
        [] (s3=7) -> rga : (s3’=14) + (1-rga) : (s3’=7);
        [] (s3=8) -> rga : (s3’=15) + (1-rga) : (s3’=8);
        [d1_GA_SA_z] (s3=9) -> (s3’=1);
        [d2_GA_SB_z] (s3=10) -> (s3’=1);
        [d3_GA_SC_z] (s3=11) -> (s3’=1);
        [d4_GA_SD_z] (s3=12) -> (s3’=1);
        [d5_GA_SE_z] (s3=13) -> (s3’=1);
        [d6_GA_SF_z] (s3=14) -> (s3’=1);
        [d7_GA_SG_z] (s3=15) -> (s3’=1);
endmodule
module Switch  // module for the mobility model
        s5 : [1..7] init 1;
        [outc_S_SC_e3] (s5=1) -> (s5’=2);
        [ine_S_SE_e3] (s5=2) -> (s5’=3);
        [outb_S_SB_e2] (s5=3) -> (s5’=4);
        [inf_S_SF_e2] (s5=4) -> (s5’=5);
        [outa_S_SA_e1] (s5=5) -> (s5’=6);
        [ing_S_SG_e1] (s5=6) -> (s5’=7);
endmodule
```

```
module User  // module for the aircraft (User segment) U
        s4 : [1..4] init 1;
        [e1_SA_Usr_ma] (s4=1) -> (s4’=1);
        [e2_SB_Usr_mb] (s4=1) -> (s4’=1);
        [e3_SC_Usr_mc] (s4=1) -> (s4’=1);
        [e4_SD_Usr_md] (s4=1) -> (s4’=1);
        [ye_SE_Usr_me] (s4=1) -> (s4’=2);
        [yf_SF_Usr_mf] (s4=2) -> (s4’=3);
        [yg_SG_Usr_mg] (s4=3) -> (s4’=4);
endmodule
```

References

[1] Arrizabalaga S, Mendizabal J, Pinte S, Sánchez J, González J, Bauer J, Themistokleous M, Lowe D. Development of an advanced testing system and smart train positioning system for ETCS applications. in: Proceedings of the 5th Transport Research Arena Conference (TRA 2015); 2014.

[2] Johnson CW. Innovation vs safety: hazard analysis techniques to avoid premature commitment in the early stage development of national critical infrastructures. In: Proceedings of 32nd international systems safety conference; 2014.

[3] Lu D, Schnieder E. Performance evaluation of GNSS for train localization. IEEE Trans. Intell. Transp. Syst. 2015;16(2):1054–9. http://dx.doi.org/10.1109/TITS.2014.2349353.

[4] Beugin J, Marais J. Simulation-based evaluation of dependability and safety properties of satellite technologies for railway localization. Transp Res Part C 2012;22(June):42–57. http://dx.doi.org/10.1016/j.trc.2011.12.002.

[5] Khanh Nguyen JMTP, Julie Beugin. RAMS analysis of GNSS based localisation system for the train control application. In: Proceedings of the 2nd international conference on computing, management and telecommunications (ComManTel 2014), IEEE; 2014. p. 101–6. http://dx.doi.org/10.1109/ComManTel.2014.6825587.

[6] Lu D, Toro FG, Schnieder E. RAMS evaluation of GNSS for railway localisation. In: Proceedings of the IEEE international conference on intelligent rail transportation (ICIRT), IEEE; 2013. p. 209–14. http://dx.doi.org/10.1109/ICIRT.2013.6696295.

[7] Milner R. A calculus of mobile processes, I. Inf Comput 1992;100(1):1–40. http://dx.doi.org/10.1016/0890-5401(92)90008-4.

[8] Milner R. A calculus of mobile processes, II. Inf Comput 1992;100(1):41–77. http://dx.doi.org/10.1016/0890-5401(92)90009-5.

[9] Norman G, Palamidessi C, Parker D, Wu P. Model checking probabilistic and stochastic extensions of the π-calculus. IEEE Trans Softw Eng 2009;35(2):209–23. http://dx.doi.org/10.1109/TSE.2008.77.

[10] Kwiatkowska M, Norman G, Parker D. PRISM: probabilistic model checking for performance and reliability analysis. ACM SIGMETRICS Perform Eval Rev 2009;36(4):40–5. http://dx.doi.org/10.1145/1530873.1530882.

[11] Alur R, Henzinger TA. Reactive modules. Form Methods Syst Des 1999;15(1):7–48. http://dx.doi.org/10.1023/A:1008739929481.

[12] Grewal MS, Andrews AP. Applications of Kalman filtering in aerospace 1960 to the present. IEEE Control Syst Mag 2010;30(3):69–78. http://dx.doi.org/10.1109/MCS.2010.936465.

[13] Lee Y-W, Suh Y-C, Shibasaki R. A simulation system for GNSS multipath mitigation using spatial statistical methods. Comput Geosci 2008;34(11):1597–609. http://dx.doi.org/10.1016/j.cageo.2008.01.004.

[14] Hansson H, Jonsson B. A logic for reasoning about time and reliability. Form Asp Comput 1994;6(5):512–35. http://dx.doi.org/10.1007/BF01211866.

[15] Hennessy M, Lin H. Symbolic bisimulations. Theor Comput Sci 1995;138(2):353–89 doi:h10.1016/0304-3975(94)00172-F.

[16] Ramsey N, Pfeffer A. Stochastic lambda calculus and monads of probability distributions. ACM SIGPLAN Not 2002;37(1):154–65. http://dx.doi.org/10.1145/565816.503288.

[17] Shaw M. GPS modernization: on the road to the future GPS IIR/IIR-M and GPS III. In: Proceedings of the international symposium of global navigation satellite systems (IGNSS 2009); 2009.

[18] Jackson K. Global Positioning System (GPS) (2013).〈http://www.lockheedmartin.com/us/products/gps.html〉.

[19] FAA. Global Positoning System (GPS) Standard Positioning Service (SPS) Performance Analysis Report (2013).

[20] Cui H, Han C. Satellite constellation configuration design with rapid performance calculation and ordinal optimization. Chin J Aeronaut 2011;24(5):631–9. http://dx.doi.org/10.1016/S1000-9361(11)60074-5.

[21] Wang H-S, Hsiao P-C. GNSS availability analysis in Taiwan—a Markov model approach In: Proceedings of the national technical meeting of the institute of navigation; 2006. p. 759–69.

[22] Taylor G, Li J, Kidner D, Brunsdon C, Ware M. Modelling and prediction of GPS availability with digital photogrammetry and LiDAR. Int J Geograph Inf Sci 2007;21(1):1–20. http://dx.doi.org/10.1080/13658810600816540.

[23] Kubik K, Feng Y, Tang T. An availability study for a Nav-Com satellite system (NCSS) in Australia. In: Proceedings of the 9th national space engineering symposium; 1994. p. 59–66.

[24] Kelley C, Dessouky M. Minimizing the cost of availability of coverage from a constellation of satellites: evaluation of optimization methods. Syst Eng 2004;7(2):113–22. http://dx.doi.org/10.1002/sys.10059.

[25] Durand J-M, Michal T, Bouchard J. GPS availability, part I: availability of service achievable for different categories of civil users. Navigation 1990;37(2):123–39. http://dx.doi.org/10.1002/j.2161-4296.1990.tb01542.x.

[26] Durand J-M, Caseau A. GPS availability, Part II: evaluation of state probabilities for 21 satellite and 24 satellite constellations. Navigation 1990;37(3):285–96. http://dx.doi.org/10.1002/j.2161-4296.1990.tb01556.x.

[27] Phlong WS, Elrod BD. Availability characteristics of gps and augmentation alternatives. Navigation 1993;40(4):409–28. http://dx.doi.org/10.1002/j.2161-4296.1993.tb02317.x.

[28] Muñoz C, Carreño V, Dowek G. Formal analysis of the operational concept for the small aircraft transportation system. In: Butler M, Jones CB, Romanovsky A, Troubitsyna E (Eds.), Rigorous development of complex fault-tolerant systems, Lecture notes in computer science, vol. 4157, Springer, Berlin, Heidelberg; 2006. p. 306–25. http://dx.doi.org/10.1007/11916246_16.

[29] Gan X, Dubrovin J, Heljanko K. A symbolic model checking approach to verifying satellite onboard software. Sci Comput Progr 2014;82:44–55. http://dx.doi.org/10.1016/j.scico.2013.03.005.

[30] Bozzano M, Cimatti A, Katoen J-P, Katsaros P, Mokos K, Nguyen VY, Noll T, Postma B, Roveri M. Spacecraft early design validation using formal methods. Reliab Eng Syst Saf 2014;132:20–35. http://dx.doi.org/10.1016/j.ress.2014.07.003.

[31] Peng Z, Lu Y, Miller A, Johnson C, Zhao T. Formal Specification and Quantitative Analysis of a Constellation of Navigation Satellites, Quality and Reliability Engineering International. http://dx.doi.org/10.1002/qre.1754.

[32] Lu Y, Miller A, Johnson C, Peng Z, Zhao T. Availability analysis of satellite positioning systems for aviation using the PRISM model checker. In: Proceedings of the 17th IEEE international conference on computational science and engineering (CSE 2014), IEEE; 2014. p. 704–13. http://dx.doi.org/10.1109/CSE.2014.148.