J Autom Reasoning
DOI 10.1007/s10817-017-9416-6

Towards Probabilistic Formal Analysis of SATS-Simultaneously Moving Aircraft (SATS-SMA)

Muhammad Usama Sardar[1] · **Nida Afaq**[1] · **Osman Hasan**[1] · **Khaza Anuarul Hoque**[2]

Received: 6 December 2016 / Accepted: 16 May 2017

Abstract The objective of NASA's Small Aircraft Transportation System (SATS) Concept of Operations (ConOps) is to facilitate high volume operation of advanced small aircraft operating in non-towered, non-radar airports. This system can provide improved and accessible air travel at a lower cost. Given the safety-critical nature of SATS, its analysis accuracy is extremely important. However, the commonly used analysis techniques, like pilot/computer simulation and traditional model checking, do not ascertain an error-free and complete verification of SATS due to the wide range of possibilities involved in SATS or the inability to capture the randomized and unpredictable aspects of the SATS ConOps environment in their models. Another limitation of these studies is that a limited speed range was used in the analysis. To overcome these limitations, we propose to formulate the SATS ConOps as a fully synchronous and probabilistic model, i.e., SATS-SMA, that supports simultaneously moving aircraft. The distinguishing features of our work include the preservation of safety of aircraft while providing a precise timing model, which is closer to reality compared to the previous hybrid analyses. Important insights related to the aircraft take-off and landing operations during the instrument meteorological conditions are also presented.

Keywords Formal verification · Probabilistic analysis · Model checking · SATS concept of operations · Aircraft safety

✉ Muhammad Usama Sardar
usama.sardar@seecs.nust.edu.pk

Nida Afaq
nida.afaq@seecs.nust.edu.pk

Osman Hasan
osman.hasan@seecs.nust.edu.pk

Khaza Anuarul Hoque
khaza.hoque@cs.ox.ac.uk

[1] School of Electrical Engineering and Computer Science (SEECS), National University of Sciences and Technology (NUST), Islamabad, Pakistan

[2] Department of Computer Science, University of Oxford, Oxford, UK

Published online: 01 June 2017

1 Introduction

There has always been an increasing demand [30] for an innovative technology in the U.S. National Airspace System that leads to a reduction in the cost of air travel while meeting the stringent safety requirements [21]. One of the major threats to air travel safety is the congestion at major airports and there have been numerous calls to reduce airport traffic since the 9/11 incident [43]. As a solution to these requirements, Small Aircraft Transportation System (SATS) [56] was proposed by National Aeronautics and Space Administration (NASA), Federal Aviation Administration, and the National Consortium for Aviation Mobility. SATS has the capability to reduce travel times [21] by availing the 3400 under-utilized small community airports in the U.S. that are not equipped with control towers or radar coverage [56]. SATS aircraft include affordable and high performance small aircraft, such as the single-engine, multi-engine (piston or turbo-powered), and jet aircraft with advanced avionic equipment [21]. The NextGen communication and navigation capabilities will be incorporated by utilizing the Ground Positioning System/Wide Area Augmentation System to reduce infrastructural costs [56]. SATS utilizes advanced guidance display systems, such as Synthetic Vision System, Highway-In-The-Sky, Flight Director, Enhanced Vision System, and Head-Up Displays, to aid in low visibility conditions [56]. These systems can reduce pilot workload and improve accuracy of the approach as well as situational awareness of the pilot [35].

During Instrument Meteorological Conditions (IMC), non-towered, non-radar airports traditionally rely on procedural separation, i.e., allowing only one aircraft to get access to the airport airspace at a given time [8]. This practice decreases the potential airport throughput [44] to the extent where the rate of operations can become as low as three landings per hour [53]. The SATS vision includes Higher Volume Operations (HVO), which enables multiple operations inside the SATS airspace, even during IMC [30] with a goal of increasing the capacity to 15–30 operations per hour [53].

An increase in capacity of the SATS airspace can lead to the increase in risk of collision with other aircraft in the SATS airspace. Therefore, it becomes necessary to ensure safety of aircraft through appropriate separation and sequencing. Various simulation environments [17,17,18,25,48,56,58,59] and formal methods [14,32–34,44–46,55] have been utilized to validate SATS Concept of Operations (ConOps). However, in all the existing methods of validation, the main focus is on the procedures and compliance with transition rules. With these limited considerations, any model with appropriate conditions can verify that the procedures are enough for the assurance of safe separation between the aircraft. However, there may be a number of random factors affecting the results of the application of procedures in the real world. For instance, the uncertainty of aircraft and pilot performance can lead to unpredictable situations of SATS procedures, such as assurance of separation between two aircraft while transitioning between zones [22]. The pilot's performance depends on his/her decision making skills, experience and situational awareness. Thus, it cannot be quantified or assumed to be same in all situations. Pilot's unexpected responses in some situations can lead to severe accident conditions, such as the collision of two aircraft over Überlingen, Germany in 2002 [31]. Another random factor is the missed approach transition. The probability of a missed approach transition may increase due to several factors, such as bad weather conditions leading to low visibility at decision height, large density of incoming air traffic, large number of aircraft taxiing from runway and pilot error [24]. Due to these uncertainties involved, traditional model checking cannot ascertain a realistic verification of SATS.

In order to increase the robustness of the system [57] by including the probabilistic considerations of SATS in its analysis and safety verification, we propose to use probabilistic

model checking techniques [6,16] for the verification of the SATS ConOps using the abstract timing from our previous work [51]. Particularly, we developed a fully synchronous Discrete-Time Markov Chain (DTMC) model of the SATS ConOps and then verified the safety and performance properties of SATS, including the landing and take-off procedures, using the probabilistic model checker PRISM [39]. However, all aircraft were assumed to have a Missed Approach Holding Fix (MAHF) of right side and thus the alternating assignment of MAHF was not modeled. Hence, the analysis could not validate some critical safety scenarios, such as two aircraft coming from the opposite sides of the base segment of the Self Controlled Area (SCA). Moreover, the aircraft kinematics were abstracted to a level that it could cover each zone of the SCA in one time unit. Such an abstract timing could not provide realistic information about time for landing or potential throughput of the airport. In this paper, we overcome these limitations by incorporating the following details in the DTMC model of SATS ConOps presented in [51]:

- Both the left and right sides of the SATS airspace have been modeled in PRISM and thus the assignment of alternate MAHF is enabled.
- The abstract timing is refined by considering zone distances and kinematics of SATS aircraft. For this purpose, we evaluated the time spent in each zone based on the speed profile of the aircraft by solving the corresponding partial differential equations in MATLAB.
- The modeling approach for a generalized SATS aircraft is presented, i.e., any SATS aircraft can be modeled by simply updating the variables representing time to cover a specific zone based on the aircraft speed.

We show that our improved DTMC is free of deadlocks. Moreover, we present the refined safety properties with the improved timing model. We also analyze the expected time for landing and expected number of departure operations in a fixed time. The analysis of two SATS aircraft, Learjet45 and Cessna172 [13], is performed for illustration.
Open-Source Contributions: We have made our PRISM model and properties and MATLAB timing analysis codes available as open-source [50] for download to facilitate researchers and verification engineers for further developments and analysis of the SATS ConOps. It can also provide key insights into the various SATS aircraft through trivial modifications in speed ranges.
The rest of the paper is organized as follows: Sect. 2 presents the state-of-the-art SATS analysis techniques and their limitations. Section 3 provides an introduction to the SATS operational concept and the PRISM model checker to facilitate the understanding of the rest of the paper. Section 4 explains the proposed methodology along with the main challenges that we faced while modeling the proposed fully synchronous system in PRISM. Section 5 presents the probabilistic verification results of the two SATS aircraft and the novel observations made. Finally, Sect. 6 concludes this paper.

2 Related Work

Traditionally, the analysis of SATS HVO has been done mainly by simulation using computer programs, in which pilots operate aircraft modules. These simulations develop the human-in-the-loop scenarios for the analysis of Air Traffic Control (ATC) in the actual operational environment to check the feasibility of SATS operations [25] and to assess the pilot's workload and situational awareness while performing an HVO task [59]. The subjective assessment of pilot's situational awareness, traffic awareness, and navigation guidance awareness was performed for SATS HVO as well as off-nominal scenarios [17]. Moreover, simulations

were performed on a General Aviation simulator and flight tests were conducted to verify the effectiveness and efficient utilization of SATS HVO by pilots [58]. Proof-of-concept simulation studies [56] performed at the Federal Aviation Administration William J. Hughes Technical Center verified that the ATC can accept the SATS procedures, support HVO, and is able to control SATS traffic into and out of the SCA. Similarly, in [18,59], the authors verified the conflict detection capabilities of HVO procedures in NASA Langley Research Center's Air Traffic Operation Lab and also performed flight tests on the NASA Cirrus-SR/22 aircraft. These simulations used different simulation environments and flight tests on SATS aircraft and verified that the SATS HVO is feasible with respect to capacity management and pilot acceptability of the procedures. Recently, a genetic algorithm has been developed using the Microsoft VC++ 6.0 environment in [4] to optimize the SATS landing sequence for multiple aircraft and to make it conflict-free while reducing delays. In addition to the nominal conditions, off-nominal situations were also simulated using the same platform in [17], to check the resulting effect on the pilot's situational awareness.

The SATS goal of increased number of operations in the airport was also verified through simulation. A Monte-Carlo analysis shows that by utilizing the SATS HVO, the throughput of a SATS airport can be increased up to 3–4 times the normal throughput of a non-towered, non-radar airport [19]. Specifically, this study shows the saturation point of SATS HVO to be 26 operations per hour, after which delays in operation increase because of queuing. The capacity of a SATS airport depends on its Airport Management Module (AMM), which acts as a sequencer of aircraft, and on the types of aircraft approaching, as the speed differences can cause delays and thus reduce capacity [8]. The Multi-Purpose Aircraft Simulation toolkit was used to simulate scenarios with different aircraft types to compare the capacity of an aircraft using procedural separation with that of AMM [48]. This simulation showed that AMM can provide significant reductions in delays during high arrival rates of the aircraft into the SCA and thus improve the capacity of the airport.

A thorough piloted simulation of all the possible conditions of the SATS ConOps and varying kinematics of each SATS aircraft in the airport requires a very large number of test runs, which is time-consuming and requires a significant amount of computational power. Thus, exhaustive simulation with precise aircraft kinematics is impractical for validation of SATS. Resultantly, Monte-Carlo simulations, as in [19], are usually performed to analyze the SATS ConOps. However, they are based on repeated random sampling and thus lack exhaustiveness [23] in terms of coverage of all the possible states in which SATS aircraft can go into. A random selection of test vectors cannot completely validate SATS ConOps as only some specific points in the input space are covered and there is always a chance of missing the input conditions that lead to an error [42]. Moreover, the results largely depend on the choice of the test scenario. For example, each aircraft considered in simulation will exhibit a different speed and performance affecting its spacing from other aircraft. Moreover, it may not be possible to consider or even foresee all corner cases. Consequently, simulation-based verification of the SATS ConOps is incomplete with respect to error detection, i.e., all errors in the system cannot be guaranteed to be detected, which is a severe limitation considering the safety-critical nature of passenger aircraft.

In order to have a complete analysis, automatic parameterized verification of hybrid automata was employed to verify the properties of SATS ConOps in [32,33]. Model-checking principles were used in this analysis, which considered position of the aircraft as a continuous variable modeled either as a timer [32] or as a rectangular differential inclusion [33]. This methodology allows verification regardless of the number of aircraft but a major limitation of this work is that the methodology requires the user to specify inductive invariants that are sufficient to establish safety. While the process of finding inductive invariants sufficient to establish

safety of the SATS ConOps has been successfully automated through an extension of invisible invariants [2], this is an incomplete (heuristic) method, that in general may fail to find such inductive invariants [34]. The analysis and formal verification of the timing constraints of SATS was done in [15] using Linear Real-Time Logic. Exhaustive state exploration using the Prototype Verification System (PVS) [47] has been performed extensively for the safety verification of the SATS ConOps [22,44,45]. In particular, it has been formally verified that SATS rules and procedures can provide minimum required spacing between two and more aircraft. Umeno et al. [55] constructed an I/O automata framework to prove the properties of the system in PVS. A hybrid modeling technique [14] has also been developed in PVS that proves separation properties by incorporating geometry of the airport and speed range of the aircraft. In [46], the PVS tool Besc was used for similar hybrid modeling. It is worth mentioning that hybrid modeling is an improvement to the verification of the transition rules only. A limitation of these studies is that they used a limited speed range that excludes the case where the two aircraft approaching the runway have a large difference in speeds. Another major limitation of these works is that they do not consider the randomized and unpredictable aspects of aircraft transitions, landing, and takeoff.

Formal verification of safety features of related terminal airspaces has also been carried out, which could be adapted for the analysis of SATS. For instance, the Runway Safety Monitor (RSM) [26], which is a protocol by NASA and Lockheed Martin to detect runway incidents, was formally modeled using Petri nets [54]. In this study, exhaustive verification of the RSM algorithm was performed using the SMART (Stochastic and Model-checking Analyzer for Reliability and Timing) tool to detect all scenarios of incursion. Attempts to verify equivalent models of RSM using the symbolic model checker NuSMV and the explicit model checker SPIN were shown to have failed to build the state-space due to excessive memory consumption. Probabilistic models in aircraft safety studies were incorporated to represent uncertainty for verifying the airborne collision avoidance system, ACAS X [57]. The analysis of ACAS X system using a parallel Bayesian model checking engine was shown to be limited in resolution. Hence, the system was discretized and the controller was modeled as a MDP. PCTL model checking was performed for the verification of its properties.

3 Preliminaries

In this section, we present an overview of the SATS operational concept and the PRISM model checker. Table 1 defines all the acronyms used in the paper.

3.1 SATS ConOps

The ConOps for SATS is primarily a set of rules and procedures to be followed by an aircraft inside a volume surrounding the airport, known as the SCA. A ground-based automated system, known as the AMM, performs the job of sequencing the aircraft entering the SCA, while the pilots are responsible for their separation [11,13,45]. The SATS HVO concept allows a maximum of 4 aircraft at a time inside the SCA [44]. The SCA is typically taken as a region with 12–15 Nautical Miles (NM) radius and 3000 feet above the ground [13,14]. It is arranged in a T structure, consisting of a base, an intermediate and a final zone. It is divided into a number of segments and fixes: Initial Arrival Fixes (IAFs), Intermediate Fix (IF), Final Approach Fix (FAF), and Departure Fixes (DFs), as shown in Fig. 1a. The IAFs serve as the holding fix when an aircraft enters the SCA, and as the MAHF when an aircraft misses landing and flies back to the IAF via the missed approach path.

Table 1 Nomenclature used in the paper

Acronym	Definition
SATS	Small Aircraft Transportation System
ConOps	Concept of operations
HVO	High volume operation
IMC	Instrument meteorological conditions
SATS-SMA	SATS-simultaneously moving aircraft
NASA	National Aeronautics and Space Administration
MAHF	Missed approach holding fix
SCA	Self controlled area
ATC	Air traffic control
AMM	Airport management module
PVS	Prototype verification system
NM	Nautical miles
IAF	Initial arrival fix
IAF-R	Initial arrival fix-right
IAF-L	Initial arrival fix-left
IF	Intermediate fix
FAF	Final approach fix
DF	Departure fix
DF-R	Departure fix-right
DF-L	Departure fix-left
RT	Runway threshold
DTMC(s)	Discrete-time Markov chain(s)
CTMC(s)	Continuous-time Markov chain(s)
MDP(s)	Markov decision process(es)
PA	Probabilistic automata
PTA	Probabilistic timed automata

There are two types of entries into the SCA: vertical and lateral [14,46], as depicted in Fig. 1b. In vertical entry, the aircraft has to hold at 3000 feet holding fix (IAF-R or IAF-L) unless it can descend to the corresponding 2000 feet holding fix. It then moves to the base segment (IAF to IF) if the transition conditions hold. On the other hand, in a lateral entry, the aircraft

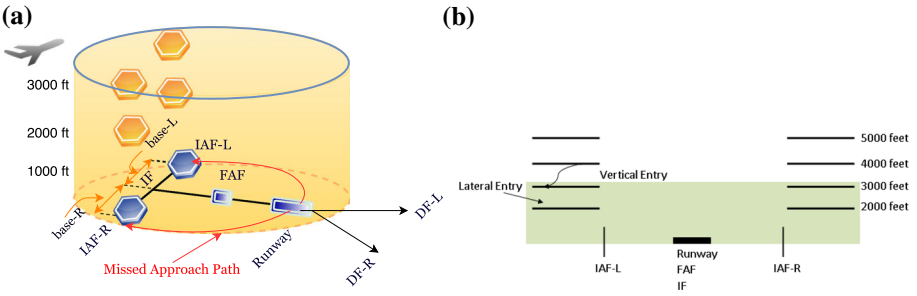

Fig. 1 *Top* and *side views* of the SCA depicting its fixes and segments, along with types of entries into the SCA. **a** *Top view* of the SCA [22]. **b** Side view of the SCA [22]. (Color figure online)

flies from the entry point to the base segment directly or through the 2000 feet holding fix. Once the aircraft is in the base segment or 2000 feet holding fix, there is no dependency on its type of entry. After base segment, the aircraft goes through the IF, FAF, and finally reaches the runway under certain conditions. If an aircraft misses its landing due to any reason, it has to follow the missed approach path to move to the IAF corresponding to its assigned MAHF, as shown in Fig. 1a.
DFs are outside the SCA and under the ATC control. An aircraft ready to depart requests ATC for clearance. After clearance, the departure operation starts at the runway and it moves to the DF corresponding to its MAHF assignment. A safe distance of 5 or 10 NM has to be maintained from the aircraft flying to the opposite or same DFs, respectively [22].

3.2 PRISM Model Checker

PRISM is a probabilistic model checker, i.e., a software tool for the formal modelling and analysis of systems that exhibit random or probabilistic behaviour. It incorporates state-of-the-art symbolic data structures and algorithms. It has been extensively applied to model and analyze stochastic systems from a wide variety of application domains, including biological systems [41], embedded control systems [37], communication [29], and security protocols [9]. PRISM supports several types of probabilistic models, such as discrete-time Markov chains, continuous-time Markov chains (CTMCs) [36], Markov decision processes (MDPs) [49], probabilistic automata (PAs) [52], probabilistic timed automata (PTAs) [12] as well as extensions of these models with rewards (or costs), referred to as (discrete- or continuous-time) Markov reward models and priced PTA. The models of the systems are developed using the *PRISM language*, which is a simple, state-based language based on Alur's Reactive Modules formalism [1]. The PRISM language primarily consists of modules and variables. A model is composed of a parallel composition of a set of modules that can interact with each other. A module consists of local variables and guarded commands. The values of these variables at any given time represent the state of the modules and the guarded commands mimic the behavior of the modules. The global state of the whole model is determined by the local state of all modules. The syntax of a command is as follows:

```
[action] guard -> prob_1 : update_1 + ... + prob_n : update_n;   (1)
```

where `action` is the optional synchronization label, `guard` is a predicate over all the variables in the model (including those belonging to other modules), `update` represents the new values of the variables in the module and `prob` represents a probability (or rate) assigned to the corresponding transition, which the module can make if the guard is true.
In order to verify and analyze the behavior of a given system, the desired functionality has to be expressed as a property in a suitable probabilistic logic using a property specification language. The PRISM's property specification language is based on temporal logic and subsumes several well-known probabilistic temporal logics, including PCTL [28], CSL [3, 7], LTL [20], and PCTL* [5], as well as support for rewards (or costs) and quantitative specifications. PCTL is used for specifying properties of DTMCs, MDPs, or PTAs; CSL is an extension of PCTL for CTMCs; LTL and PCTL* can be used to specify properties of DTMCs and MDPs (or untimed properties of CTMCs) [40].
The probabilistic operator `P` is used to reason about the probability of an event's occurrence. It can be used to verify bounded or quantitative properties. The bounded properties take the form:

```
P bound [ pathprop ]                                    (2)
```

bound can be >=p, >p, <=p or <p, where p is a PRISM language expression evaluating to a double value in the range [0,1] and pathprop is a path property using temporal operators X (next), U (until), F (eventually/future), G (always/globally), W (weak until), R (release) and their complex combinations. The property mentioned in Code Listing 2 is true in a state s of a model if the probability that path property pathprop is satisfied by the paths from state s meets the bound bound. The quantitative properties compute the actual probabilities, rather than just verifying the bound, and take the form:

```
P =? [ pathprop ]                                       (3)
```

The steady-state operator S and reward operator R [38] have similar forms [40].

4 Formal Modeling of SATS in PRISM

In this section, we first present our refinements to the SATS ConOps. Then, the main challenges encountered in modeling the system in PRISM and our proposed solutions are described.

4.1 Refinements to the Original SATS

Our proposed model of the SATS ConOps in the PRISM language overcomes some of the limitations of the fully non-deterministic, asynchronous transition system presented by Dowek et al. [22]. Before presenting the details of our DTMC model, we point out the discrepancies in the existing algorithm and our proposed solution.

1. The Idle Effect: In a fully non-deterministic model, if multiple transitions are possible at the same time, either one of them may be executed. In other words, only one non-deterministic action is fired at a time. This means that in such a model of SATS, at each time step, *only one* aircraft will move to the next zone while all the other aircraft hold in their current zones, even if the conditions for their transitions are satisfied. Thus, one aircraft could change zones several times while others remain idle [22]. Hence, such a model is unrealistic, as it fails to depict the real scenario, where an aircraft cannot be idle in the air [44].
2. Simultaneous Transitions: The lowest available altitude determination for an aircraft executing a missed approach (Rule 12) [22] is a simultaneous transition, potentially involving 2 aircraft, when the holding pattern at 3000 feet is occupied but 2000 feet is available. In this case, the transition determines 3000 feet as the lowest available altitude and forces the aircraft holding at 3000 feet to descend to the holding pattern at 2000 feet concurrently. This is a weakness of the formalization [22,44] because simultaneous transitions are not possible in a fully non-deterministic model. If simultaneous transitions are suppressed, a deadlock scenario will be created [22].

Our proposed refinement for both of the above weaknesses is to build a fully synchronous model that allows concurrently moving aircraft. Hence, at each time step, all the aircraft are allowed to proceed simultaneously if the conditions for their transition are fulfilled. More-

over, such a model also facilitates simultaneous transitions in the lowest available altitude determination.

4.2 Modeling SATS as a DTMC in PRISM

In order to realistically model the semantics of the communication between aircraft and AMM, both aircraft and AMM should have different modules in PRISM. Unfortunately, there is no direct way of changing a variable in a different module for *only one* probabilistic update of a command in the *same* time step. Using synchronization labels to synchronize all modules over common actions, only a *specific scenario* can be modeled. Moreover, an important limit on the use of global variables in PRISM is the fact that global variables are not editable on a synchronized command [40]. PRISM detects this and reports an error if an attempt is made to do so. Therefore, the main challenge is to achieve synchronization in simultaneous aircraft movement whenever the guard conditions are satisfied, while incorporating probabilistic updates from the SATS ConOps in the model. In order to cope with this challenge, we modeled the system as fully synchronously parallel automata, as in [27], where the same synchronization label `t` is assigned to each PRISM command in each module. In order to allow simultaneous transitions, we ensure that *at least* one transition of each module is active for each reachable state in our model to avoid deadlocks. Hence, in such a fully synchronous model, all the aircraft move concurrently to the next respective zones whenever the conditions are satisfied.

We formalize the SATS ConOps as a DTMC in the PRISM model checker [39] using a refined timing model. Figure 2 depicts the possible transitions of the aircraft from one zone to the next zone of the SCA. Our model ensures that after a landing aircraft has landed safely, it unloads passengers of the current flight in the taxi state. Then, it loads passengers of the next flight and is ready for departure. After departure, it reaches its destination and the next time it becomes a landing aircraft for the SCA. Hence, the process of landing and departure continues. The model consists of three major modules, i.e., `aircraft`, `AMM`, and `Probability mapping`, and their interaction is shown in Fig. 3. The `aircraft` module implements the transition rules inside the SCA. The state of the aircraft inside the SCA is represented by the zone in which it is present, encoded as shown in Table 2, and the

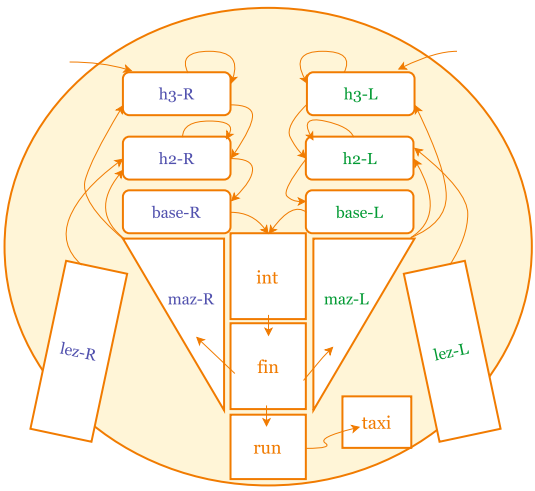

Fig. 2 Possible transitions of the SATS ConOps in zones of the SCA [22]. (Color figure online)

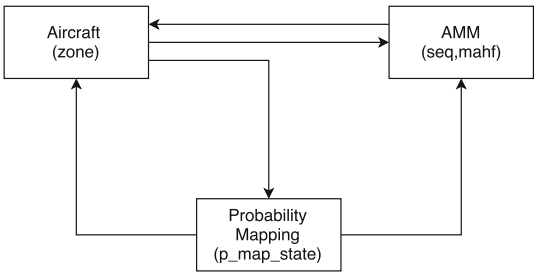

Fig. 3 PRISM modules along with their local variables

Table 2 Encoding for zones of the SCA in aircraft module of PRISM [51]

Symbol	Description	Encoding
h3-R	Holding at 3000 feet at right side	1
h3-L	Holding at 3000 feet at left side	2
h2-R	Holding at 2000 feet at right side	3
h2-L	Holding at 2000 feet at left side	4
lez-R	Lateral entry zone at right side	5
lez-L	Lateral entry zone at left side	6
base-R	Right segment of base (IAF-R to IF)	7
base-L	Left segment of base (IAF-L to IF)	8
int	Intermediate segment (IF to FAF)	9
fin	Final segment (FAF to runway)	10
run	Runway	11
maz-R	Missed approach zone at right of base	12
maz-L	Missed approach zone at left of base	13
taxi	Taxi	14
dep-R	Departure path towards right departure fix (DF-R)	15
dep-L	Departure path towards left departure fix (DF-L)	16

total time spent `count_time` inside the zone. The `AMM` module models how the sequence number `seq` and MAHF `mahf` are assigned to the aircraft. The complete information about the aircraft will thus include the sequence number `seq` and MAHF `mahf` assigned by the AMM and the current location `zone` and total time `count_time` in a zone of the aircraft. The `Probability mapping` module allows updating a variable in a different module for only one probabilistic update of a command in the *same* time step. It is required as a major module because of the lack of expressiveness in PRISM.

4.2.1 Model of SATS Transition Rules and Procedures

In this paper, we consider two aircraft in the SCA. The modules `aircraft1` and `aircraft2` in our formal model [50], corresponding to each aircraft, implement the rules of the ConOps, i.e., under what conditions the aircraft moves from one zone to the next. The current zone of the aircraft is represented by the state variables `zone1` and `zone2`. They are modelled as integer variables with values in the range 0–16 according to the encoding listed in Table 2, whereas the value 0 represents the 'fly zone' for an aircraft outside the SCA.

For instance, vertical entry left procedure (Rule 1) [22] for aircraft1 is modelled by the following PRISM command:

```
[t] zone1=0 & mahf1=false & (fix_total_L+fix_total_MAHF_L)<2 &
approach_L=0 & z13_total=0 & z6_total=0 & z2_total=0 -> (zone1'=2);
```
(4)

where mahf1 represents the MAHF assigned to aircraft1, fix_total_L represents the total number of aircraft at IAF-L, fix_total_MAHF_L represents the total number of aircraft assigned to IAF-L as a MAHF and approach_L represents the number of aircraft assigned to IAF-L as a MAHF on the approach. The variables z13_total, z6_total and z2_total represent the total number of aircraft in the missed approach zone (left), lateral entry zone (left) and 3000 feet holding fix (left), respectively. In our model [50], we used formulas for compact representation of these conditions and to avoid repetition. The update in the command shows that if all the guard conditions are satisfied, the aircraft proceeds to the 3000 feet holding fix (left).
The modules are symmetric except that priority is assigned to aircraft1 in case of simultaneous entry as well as departure. In addition to the simultaneous entry due to proposed synchronous system, we also allow the aircraft to enter individually in order to cater for the aircraft performance and pilot preferences. This is done by using the synchronization labels t1 and t2 for aircraft1 and aircraft2, respectively.

4.2.2 Model of the AMM

The AMM typically resides at airport ground and communicates with the aircraft via a data link [13]. It grants permissions to the aircraft for entering the SCA [11,59] and assigns a landing sequence and a MAHF side (right or left) to the aircraft. The landing sequence numbers encode the leader information and also identify whether an aircraft is the first aircraft in a specific zone of the SCA. The aircraft entering later thus follows the leader during the transitions. If the entering aircraft is the first one in the sequence, then its MAHF will be in the same side from which it is entering. Whereas, the next aircraft, with sequence other than 1, will have the MAHF that is opposite to that of its leader.
We model the AMM as a separate module AMM in PRISM to represent the communication with the aircraft. It has two state variables, i.e., seq and mahf, for each aircraft. For a landing aircraft, seq represents the relative landing sequence number, such that the aircraft with landing sequence n is the leader of the aircraft with landing sequence $n+1$, i.e., an aircraft with sequence number 1 is the leader of the aircraft with sequence number 2. It is modelled as an integer variable with values in the range 0–10. When an aircraft enters the SCA, seq is assigned a new value calculated by the formula nextseq. This value is calculated based on the number of the aircraft already in the landing zones of the SCA. In case of a simultaneous entry by both aircraft, different sequence numbers are assigned to both the aircraft, with priority to aircraft1. A new sequence number is also assigned when an aircraft initiates a missed approach path and the sequence numbers of all other aircraft in the landing zones of the SCA are decremented by one. Moreover, when an aircraft enters the runway, the sequence numbers of all other aircraft in the SCA are again decremented by one. When an aircraft moves to the taxi state, its sequence number becomes 0. For a departing aircraft, seq represents the distance of the aircraft from the runway in nautical miles. It is incremented by one in each time step when it is in one of the departure zones, until it becomes 10, where it is assumed to have left the SCA.

The MAHF of an aircraft, represented by `mahf`, is a Boolean variable with `true` representing right MAHF, and `false` representing left MAHF. It is assigned whenever an aircraft enters the SCA. Moreover, it is re-assigned when an aircraft executes a missed path approach.

4.2.3 Timing Model

In our previous work [51], we assumed that the aircraft remains in a SCA zone for one time unit and transitions to another zone take place whenever the conditions of transition are satisfied. In this work, we propose the refinement of the abstract timing model, in which the objective is to calculate the actual time taken by the aircraft in each zone during the landing operation, and to determine the precise location of each aircraft. Thus, the time of an aircraft's stay in one zone may be more than one time unit depending on the aircraft's speed and the conditions of transition to the next zone.

During the landing operation, the speed of the aircraft decreases gradually in small steps, i.e., the aircraft first stabilizes in the new decreased speed and then the speed decreases further as the aircraft moves forward. The conventional laws of kinematics are not applicable here since the deceleration is not uniform. To incorporate the kinematics of aircraft, we consider the speed profiles of two SATS aircraft, given in [13] for a Learjet45 and a Cessna172 in which the speed of the first one is slightly higher than the latter one. The distance between runway threshold (RT) and IF is taken as 10 nautical miles [13] whereas the distance between IAF and IF is taken as 5 nautical miles [22].

The speed of an aircraft remains constant in some parts of the SCA while it is decreasing in other parts (R2, R4 in Learjet45 profile, and R3, R5 in Cessna172 profile, respectively), as shown in Fig. 4. The representation of speed is not very straightforward, due to the fact that the speed has been plotted against the traveled distance towards RT, instead of time [13]. This is because the actual time taken by an aircraft in a zone also depends on the satisfaction of the conditions for transition from one zone to the next. In the regions of constant speed, the time t can be calculated by $t = x/v$, where x represents the distance and v represents the speed. As mentioned earlier, the deceleration of the aircraft is not constant, which restricts us from applying the laws of motion directly to these regions. We use differential equations to represent the regions of non-uniform deceleration. For instance, let us take one representative region, say R2 (in Fig. 4) in the Learjet45 speed profile whose speed decreases from initial speed v_1 to final speed v_2, covering a distance of $x = x_2 - x_1$ in time t.

If the slope of the speed–distance graph in region R2 is $C = (v_2 - v_1)/(x_2 - x_1)$, then from the straight line equation we can write the speed v as:

$$\dot{x} = v = \frac{v_2 - v_1}{x_2 - x_1}(x - x_1) + v_1 \tag{5}$$

where x represents the distance covered and $\dot{x}$ is the time derivative of distance. The solution to this differential equation after substitution of the boundary value condition $x = x_1$ at time $t = 0$ (for specific region, instant at which speed has not declined), we get the following expression for time:

$$t = \frac{1}{C}\frac{ln(Cx + b)}{ln(Cx_1 + b)} \tag{6}$$

where b is a constant and can be defined as $b = v_1 - Cx_1$. The resulting minimum time required to cover each region for a symmetric SCA is presented in Tables 3 and 4 for a Learjet45 and a Cessna172, respectively. It can be observed that the fast aircraft Learjet45 requires less time as compared to Cessna172 to reach RT because of its higher speed range. The corresponding speed–time graphs are depicted in Fig. 5.

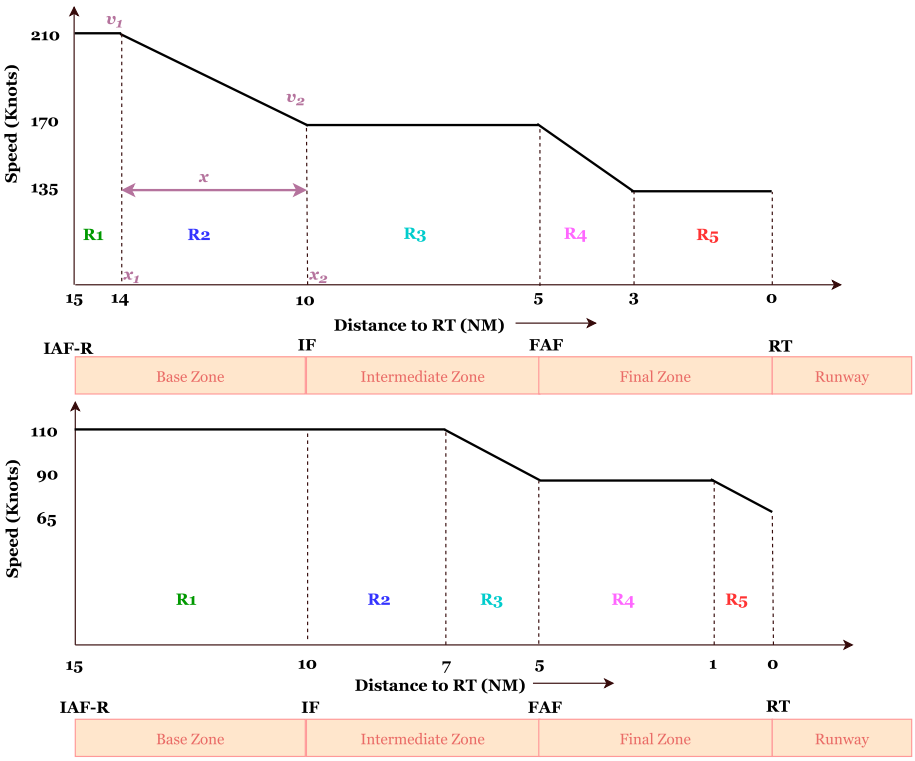

Fig. 4 Speed profiles of the two types of SATS aircraft in base, intermediate and final zones of the SCA, *Top* Learjet45, *Bottom* Cessna172 [13]

Table 3 Minimum time required for a Learjet45 in base, intermediate and final zones of the SCA

Region	SCA zone	Distance (NM)	Speed (knots)	Time (s)
15–14 NM	Base	1	210	17.14
14–10 NM	Base	4	210–170	76.074
IF to FAF	Intermediate	5	170	105.88
FAF to 3 NM	Final	2	170–135	47.424
3 NM to RT	Final	3	135	80

Table 4 Minimum time required for a Cessna172 in base, intermediate and final zones of the SCA

Region	SCA zone	Distance (NM)	Speed (knots)	Time (s)
15 NM to IF	Base	5	110	163.63
IF to 7 NM	Intermediate	3	110	98.18
7 NM to FAF	Intermediate	2	110–90	72.24
FAF to 1 NM	Final	4	90	160
1 NM to RT	Final	1	90–65	46.86

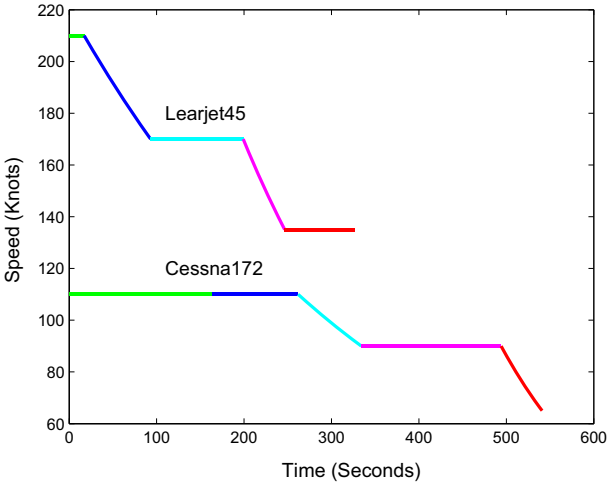

Fig. 5 Minimum time required in various regions in base, intermediate and final zones of the SCA for a Learjet45 and a Cessna172. *Green* Region1, *Blue* Region2, *Cyan* Region3, *Magenta* Region4, *Red* Region5. (Color figure online)

In this paper, we assume that one time step of PRISM is equivalent to one second. We keep track of the time spent in a zone by using counters `count1_time` and `count2_time` for aircraft1 and aircraft2, respectively, in our formal model. The counters are incremented at each time step till the desired value corresponding to the region in Tables 3 and 4 is achieved. When the desired value is reached for a specific zone, the conditions for the transition to the next zone are checked. If the conditions are satisfied, the aircraft moves on to the next zone and the counter is reset. If the conditions are not satisfied, the aircraft remains in the same zone and the counter is incremented at each time step. This formalism allows us to use the same counter for all zones of the SCA and thus save memory space to make the model scalable. For instance, the counter in the intermediate zone for aircraft1 is implemented as follows:

```
[t] zone1=9 & count1_time < int_time ->
(count1_time'=min(count1_time+1,count_max));        (7)
```

where `count1_time` represents the time spent by aircraft1 in its current zone and `int_time` is the value for the desired time in intermediate zone from Tables 3 and 4 for a Learjet45 and a Cessna172, respectively. The PRISM function `min(i,j)` returns the minimum of the two values i and j, and is utilized to restrict the values within the maximum value of the count `count_max`.

4.2.4 Randomness in Model

Since there is no direct way of changing a variable in a different module for only one probabilistic update of a command in the *same* time step, we introduce an additional probability mapping module for each probabilistic decision. For instance, consider an aircraft in the final zone. Now it can either choose the missed approach path with a probability `p_map` or it can continue landing and transit to the runway with probability `1-p_map`. In case of the

missed approach path, a new sequence number and MAHF is to be assigned to the aircraft. However, there is no change in its sequence number and MAHF if it proceeds to the runway. We propose to use the probability mapping module, i.e., `choose_p_map`, which contains a single state variable `p_map_state` of type integer and with two possible values: 0 and 1. When the probability `p_map` is selected, `p_map_state` is set to 1, otherwise it is 0. This is achieved by using the following command in PRISM:

```
[t] Guard -> p_map:(p_map_state' = 1) + (1-p_map):(p_map_state' = 0);
```
(8)

It is important to note that instead of setting `true` as a guard, we use the conditions of transition to the final zone, i.e., one step back condition as the guard [50]. This way, the command does not execute on each time step. `p_map_state` is updated when the aircraft enters the final zone and is ready to be used when checking conditions for the next transition to runway or missed approach zone in the next time step.
The value of `p_map_state` is now used in such a way that the guard condition of `p_map_state=1` checks whether `p_map` is selected. For instance, in the `AMM` module, the following command ensures that `seq1` and `mahf1` are updated as soon as it makes the transition to zone 12:

$$[t]\ Guard\ \&\ p\_map\_state = 1 \rightarrow (seq1' = nextseq)\ \&\ (mahf1' = nextmahf1);$$

4.2.5 Model Statistics

The model contains 5 local state variables for each aircraft. The `aircraft1` module contains two local state variables: `zone1` and `count1_time`. The `AMM_a1` module contains two local state variables: `seq1` and `mahf1`. The `choose_p_map1` module has only one local state variable, i.e., `p_map_state1`. There are no global variables in our model. The model with abstract timing contains 67 formulas while the one with refined timing contains 68 formulas. The number of states and transitions are presented in Table 5. The complete formalization took approximately 150 man hours.

5 Verification Results

With the improved timing model and inclusion of left MAHF, the model is verified deadlock free using the PRISM property `filter(exists, "deadlock")`. This section presents the safety and performance of the two types of aircraft by considering each type of aircraft individually, i.e., all the traffic is by one aircraft.

Table 5 Statistics related to size of the model

Model statistics	Abstract timing	Refined timing	
		Learjet45	Cessna172
Reachable states	508	275952	809114
Transitions	667	278666	813949

5.1 Safety Properties

The safety verification is based on the number of aircraft in a zone and their separation from other aircraft in other zones [44]. In our previous work [51], we assumed that an aircraft covers a SCA zone in one time unit and thus two landing aircraft were considered at risk of collision when they were in the same zone. Moreover, only right side of the SCA was considered. With the refined timing model and consideration of both sides of the SCA, we refine the safety properties accordingly. According to the new model, two landing aircraft will collide when they are at the same location in the corresponding zones. These zones include the approach, final approach, missed approach and runway zones. Based on the refined model, explained in Sect. 4, the location of the two landing aircraft, aircraft1 and aircraft2, is tracked by the counter variables `count1_time` and `count2_time`, respectively. Hence, we label the state of collision `land_danger` as follows:

```
label "land_danger" = ((zone1=7 & zone2=8)|(zone1=9 & zone2=9)
                    |(zone1=10 & zone2=10)|(zone1=11 & zone2=11)
           |(zone1=12 & zone2=13)) & (count1_time=count2_time);   (9)
```

The precise locations of the departing aircraft are represented by the variables `seq1` and `seq2`. Aircraft on opposite departure zones must be safely separated by at least 3 NM. Thus, they are at risk when they are on the opposite departure zones and the absolute difference of `seq1` and `seq2` is less than 3 NM. Using the definition of absolute value, we label this state `dep_danger` as follows:

```
label "dep_danger" = (zone1=15 & zone2=16 & (seq1-seq2)>=0 &
                                              (seq1-seq2)<3)
    |(zone1=15 & zone2=16 & (seq1-seq2)<0 & -(seq1-seq2)<3 );   (10)
```

Using the above labels, we analyze safety in all paths in our model by computing the value of the probability that any of the `land_danger` and `dep_danger` is satisfied in the future by the paths from the initial state. This is achieved by the following property:

```
P=? [F "land_danger" | "dep_danger"];                          (11)
```

PRISM shows a result of 0, which confirms that no path leads to a collision in the landing or departure zones from the initial state.
In order to confirm that the probability of occurrence of any of the `land_danger` and `dep_danger` remains 0 for all *reachable* states, we formalize the property using filters as follows:

```
filter(forall, P<=0 [ F "land_danger" | "dep_danger" ]);       (12)
```

The property verifies to be true in PRISM and thus guarantees the safety in our model.

5.2 Analysis of Landing and Departure Operations

5.2.1 Expected Time for Landing

We utilize the *reachability* reward [40] in PRISM to find the *expected* time taken for the landing of an aircraft in our model. We assign a reward of unity to each state of the model using the reward structure `Landing_time` and accumulate the rewards along a path until the aircraft is in the taxi state. For instance, the reward-based property for `aircraft1` is presented below:

```
R "Landing_time" = ? [ F "landings1" ];          (13)
```

where `landings1` is the label assigned to a state where `aircraft1` has landed.
Since very limited information is available on the probability of executing a missed approach path `p_map` for SATS, we utilize the PRISM's parametric model checking functionality to perform the sensitivity analysis on the values of `p_map` from 0.001 to 0.7 with a step size of 0.01. Figure 6a shows the expected time for landing of an aircraft with an abstract timing model of one time step in each zone. The overall expected time for any aircraft to land is also shown. The results after incorporation of precise timing are shown in Fig. 6b for both the Cessna172 and Learjet45 aircraft. The results depict an exponential increase in the expected time taken for landing with `p_map`. Interestingly, the time for landing of Learjet45 exceeds the overall time for landing of Cessna172 at a probability of 0.65 approximately. Such interesting

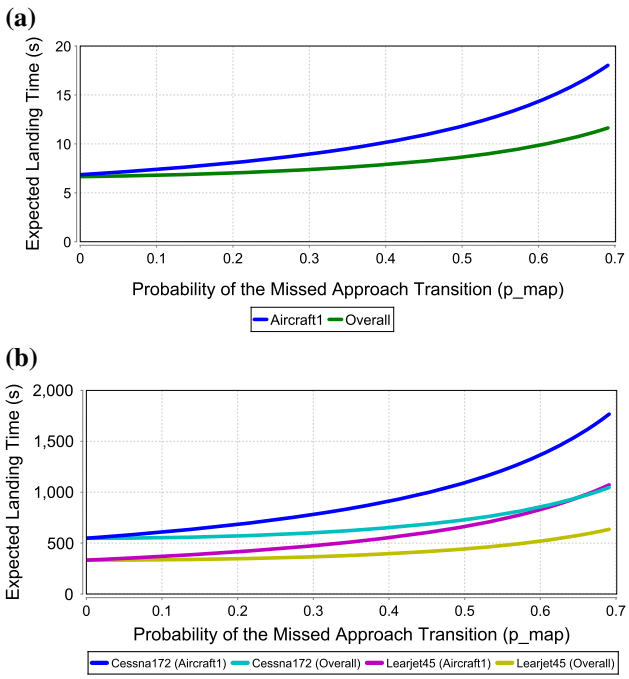

Fig. 6 Expected time for landing of aircraft. **a** Abstract timing. **b** Refined timing. (Color figure online)

results could not be obtained without the incorporation of aircraft kinematics. Moreover, the values for the Learjet45 aircraft are smaller as compared to those of the Cessna172 aircraft, because of the faster speed profile of the Learjet45. Finally, as the probability of a missed approach transition increases, the difference between the time for landing of both the aircraft increases. It is important to note that all these results have been obtained after the incorporation of alternating MAHF.

5.2.2 Expected Number of Departures in a Fixed Time

We utilize the *cumulative* reward properties [40] to find the *expected* number of departures of the aircraft in a fixed time in our model. In this case, a reward of unity is awarded to each transition of departure and the rewards are accumulated until T time steps have elapsed. The experiment is performed with T set to 100,000, which is large enough for the purpose of comparative analysis between Cessna172 and Learjet45 aircraft. Figures 7a and 7b show the results with abstract and refined timing, respectively. As the probability of missed approach transition increases, the difference between the number of departures of both the aircraft decreases.

5.2.3 Verification Statistics

The properties file for abstract timing contains four labels while the one with refined timing contains five labels. The verification is performed on a high-end server Intel Xeon processor

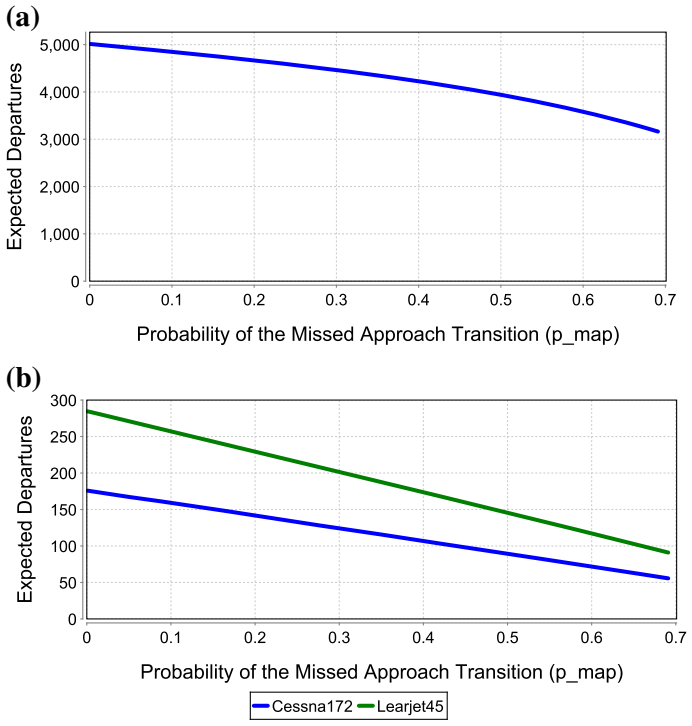

Fig. 7 Expected number of departures in a fixed time. **a** Abstract timing. **b** Refined timing. (Color figure online)

Table 6 Timing statistics for landing (all times in seconds)

Timing statistics	Abstract timing	Refined timing	
		Learjet45	Cessna172
Time for model construction	0.055–0.159	20.802–24.026	16.718–17.433
Time for model checking (landing)	0.002–0.035	26.379–117.803	102.231–515.239
Time for model checking (departure)	0.413–0.448	268.377–306.964	870.107–938.816

E5-2407 v2 (2.40 GHz, 4 CPUs) with 32 GB of RAM. At a fixed value of the probability of missed approach path `p_map`, the time for model checking for a Cessna172 is generally higher than that of a Learjet45. Moreover, it was noticed that time for model checking generally increases with the increase in `p_map`. Finally, the time for model checking with refined timing is higher than that of abstract timing. The summary of the timing is presented in Table 6.

6 Conclusion

A number of random factors affect the operation of aircraft inside the SCA, such as pilot's and aircraft performance, entry of aircraft into the SCA and transitions between zones. Therefore, we propose to use a probabilistic model checker, PRISM, to analyze the SATS ConOps in this paper. A fully synchronous DTMC model of SATS is proposed. This model allows simultaneously moving aircraft as opposed to the traditional non-deterministic, asynchronous model in which only one aircraft moves at a time instant while others remain idle. Moreover, the successful modeling and verification of the transition procedures for two aircraft with different speed profiles, has verified the safety of aircraft in terms of safe separation in all zones including take-off and landing. Such modeling has made a realistic analysis possible, due to the incorporation of timing analysis. To the best of our knowledge, this is the first reported analysis of SATS with an accurate timing model of the system, which is done with actual speed ranges of the aircraft. The landing and departure operations of SATS are analyzed with respect to the probability associated with the missed approach transition.

An important direction of future work is to carry out a detailed comparison of non-SATS (one-in/one-out), SATS, and SATS-SMA. The consideration of both Cessna172 and Learjet45 simultaneously along with 4 aircraft inside the SCA is also a very interesting future work. Furthermore, we also plan to conduct the probabilistic analysis of the SATS ConOps under off-nominal conditions [10,17,45], such as equipment malfunction and emergency situations, using the parametric model checking functionality of PRISM, like it was utilized for the analysis of probability of missed approach in this paper.

References

1. Alur, R., Henzinger, T.A.: Reactive modules. Form. Methods Syst. Des. **15**(1), 7–48 (1999)
2. Arons, T., Pnueli, A., Ruah, S., Xu, Y., Zuck, L.: Parameterized verification with automatically computed inductive assertions? In: Computer Aided Verification, vol. 2102, pp. 221–234. Springer (2001)
3. Aziz, A., Sanwal, K., Singhal, V., Brayton, R.: Verifying continuous time markov chains. In: Computer Aided Verification, vol. 1102, pp. 269–276. Springer (1996)

4. Bai, C., Zhang, X.: Aircraft landing scheduling in the small aircraft transportation system. In: Computational and Information Sciences, pp. 1019–1022. IEEE (2011)
5. Baier, C.: On algorithmic verification methods for probabilistic systems. Technical Report, Universität Mannheim (1998)
6. Baier, C., Katoen, J.P.: Principles of Model Checking. MIT Press, Cambridge (2008)
7. Baier, C., Katoen, J.P., Hermanns, H.: Approximative symbolic model checking of continuous-time markov chains. In: Concurrency Theory, vol. 1664, pp. 146–161. Springer (1999)
8. Balakrishnan, H., Chandran, B.: Scheduling aircraft landings under constrained position shifting. In: Guidance, Navigation, and Control Conference and Exhibit. American Institute of Aeronautics and Astronautics (2006)
9. Basagiannis, S., Petridou, S., Alexiou, N., Papadimitriou, G., Katsaros, P.: Quantitative analysis of a certified e-mail protocol in mobile environments: a probabilistic model checking approach. Comput. Secur. **30**(4), 257–272 (2011)
10. Baxley, B., Williams, D., Consiglio, M., Conway, S., Adams, C., Abbott, T.: The small aircraft transportation system, higher volume operations off-nominal operations. In: Aviation, Technology, Integration, and Operations Conference. American Institute of Aeronautics and Astronautics (2005)
11. Baxley, B., Williams, D., Consiglio, M., Adams, C., Abbott, T.: Small aircraft transportation system, higher volume operations concept and research summary. J Aircr **45**(6), 1825–1834 (2008)
12. Beauquier, D.: On probabilistic timed automata. Theor. Comput. Sci. **292**(1), 65–84 (2003)
13. Carreño, V.: Concept for multiple operations at non-tower non-radar airports during instrument meteorological conditions. In: Digital Avionics Systems Conference, pp. 5.B.1–1–5.B.1–9. IEEE (2003)
14. Carreño, V., Muñoz, C.: Safety verification of the small aircraft transportation system concept of operations. In: Aviation, Technology, Integration, and Operations Conference. American Institute of Aeronautics and Astronautics (2005)
15. Cheng, A., Niktab, H., Walston, M.: Timing analysis of small aircraft transportation system (SATS). In: Embedded and Real-Time Computing Systems and Applications, pp. 58–67. IEEE (2012)
16. Clarke, E.M., Grumberg, O., Peled, D.A.: Model Checking. MIT Press, Cambridge (1999)
17. Consiglio, M., Conway, S., Adams, C., Syed, H.: SATS HVO procedures for priority landings and mixed VFR/IFR operations. In: Digital Avionics Systems Conference, pp. 13.B.2–1–13.B.2–8. IEEE (2005)
18. Consiglio, M., Carreno, V.A., Williams, D.M., Muñoz, C.: Conflict prevention and separation assurance in small aircraft transportation systems. J. Aircr. **45**(2), 353–358 (2008)
19. Consiglio, M., Sturdy, J.: Monte carlo analysis of airport throughput and traffic delays using self separation procedures. In: International Council of the Aeronautical Sciences (2006)
20. Demri, S., Goranko, V., Lange, M.: Temporal Logics in Computer Science: Finite-State Systems. Cambridge University Press, Cambridge (2016)
21. Dou, L., David, L., Jesse, J., Peter, K.: A small aircraft transportation system (SATS) demand model. Technical Reports NASA/CR-2001-210874, NASA Technical Reports Server (2001)
22. Dowek, G., Muñoz, C., Carreño, V.: Abstract model of the SATS concept of operations: initial results and recommendations. Technical Reports NASA/TM-2004-213006, NASA Technical Reports Server (2004)
23. Fedeli, A., Fummi, F., Pravadelli, G.: Properties incompleteness evaluation by functional verification. IEEE Trans Comput **56**(4), 528–544 (2007)
24. Gariel, M., Spieser, K., Frazzoli, E.: On the statistics and predictability of go-arounds. In: Intelligent Data Understanding, pp. 75–91 (2011)
25. Greco, A., Magyarits, S., Doucett, S.: Air traffic control studies of small aircraft transportation system operations. In: Digital Avionics Systems Conference. pp. 13.A.4–1–13.A.4–12. IEEE (2005)
26. Green Jr, D.F., Jones, D.R.: Runway safety monitor algorithm for runway incursion detection and alerting. Technical Reports NASA/CR-2002-211416, NASA Technical Reports Server (2002)
27. Güdemann, M., Ortmeier, F.: A framework for qualitative and quantitative formal model-based safety analysis. In: High-Assurance Systems Engineering, pp. 132–141. IEEE (2010)
28. Hansson, H., Jonsson, B.: A logic for reasoning about time and reliability. Form. Asp. Comput. **6**(5), 512–535 (1994)
29. Harine, G., Marie, R., Puigjaner, R., Trivedi, K.: Loss formulas and their application to optimization for cellular networks. IEEE Trans. Veh. Technol. **50**(3), 664–673 (2001)
30. Holmes, B.J., Durham, M.H., Tarry, S.E.: Small aircraft transportation system concept and technologies. J. Aircr. **41**(1), 26–35 (2004)
31. Johnson, C.: Final Report: Review of the BFU Überlingen accident report. Contract C/1.369/HQ/SS/04. Eurocontrol (2004)
32. Johnson, T.T., Mitra, S.: Parameterized verification of distributed cyber-physical systems: an aircraft landing protocol case study. In: Cyber-Physical Systems, pp. 161–170. IEEE (2012)

33. Johnson, T.T., Mitra, S.: A small model theorem for rectangular hybrid automata networks. In: Formal Techniques for Distributed Systems, vol. 7273, pp. 18–34. Springer (2012)
34. Johnson, T.T., Mitra, S.: Invariant synthesis for verification of parameterized cyber-physical systems with applications to aerospace systems. In: Infotech at Aerospace Conference. American Institute of Aeronautics and Astronautics (2013)
35. Kelly, W.E., Valasek, J., Wilt, D., Deaton, J., Alter, K., Davis, R.: The design and evaluation of a traffic situation display for a SATS self controlled area. In: Digital Avionics Systems Conference, pp. 13.A.3–1–13.A.3–12. IEEE (2005)
36. Kulkarni, V.: Modeling and Analysis of Stochastic Systems. Taylor & Francis Group, CRC Press (2016)
37. Kwiatkowska, M., Norman, G., Parker, D.: Controller dependability analysis by probabilistic model checking. Control Eng. Pract. **15**(11), 1427–1434 (2007)
38. Kwiatkowska, M., Norman, G., Parker, D.: Stochastic model checking. In: Formal Methods for Performance Evaluation, vol. 4486, pp. 220–270. Springer (2007)
39. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: verification of probabilistic real-time systems. In: Computer Aided Verification, vol. 6806, pp. 585–591. Springer (2011)
40. Kwiatkowska, M., Norman, G., Parker, D.: PRISM: Probabilistic Symbolic Model Checker. http://www.prismmodelchecker.org (2016)
41. Lakin, M.R., Parker, D., Cardelli, L., Kwiatkowska, M., Phillips, A.: Design and analysis of DNA strand displacement devices using probabilistic model checking. J. R. Soc. Interface **9**(72), 1470–1485 (2012)
42. Lam, W.K.: Hardware Design Verification: Simulation and Formal Method-Based Approaches. Prentice Hall Modern Semiconductor Design Series. Prentice Hall, Upper Saddle River (2005)
43. Lin, C.E., Hung, T.W., Chen, H.Y.: TCAS algorithm for general aviation based on ADS-B. J. Aerosp. Eng. **230**(9), 1569–1591 (2016)
44. Muñoz, C., Dowek, G., Carreño, V.: Modeling and verification of an air traffic concept of operations. Softw. Eng. Notes **29**(4), 175–182 (2004)
45. Muñoz, C., Carreño, V., Dowek, G.: Formal analysis of the operational concept for the small aircraft transportation system. In: Rigorous Development of Complex Fault-Tolerant Systems, vol. 4157, pp. 306–325. Springer (2006)
46. Muñoz, C., Dowek, G.: Hybrid verification of an air traffic operational concept. In: Leveraging Applications of Formal Methods, Verification, and Validation, pp. 1–13. IEEE/NASA (2005)
47. Owre, S., Rushby, J.M., Shankar, N.: PVS: a prototype verification system. In: Automated Deduction, vol. 607, pp. 748–752. Springer (1992)
48. Peters, M.: Capacity analysis of the NASA Langley airport management module. In: Digital Avionics Systems Conference, pp. 4.D.6–1–4.D.6–12. IEEE (2005)
49. Puterman, M.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley, Hoboken (2014)
50. Sardar, M.U.: Towards probabilistic formal analysis of SATS-Simultaneously Moving Aircraft (SATS-SMA). (2016) http://save.seecs.nust.edu.pk/projects/SATS-SMA
51. Sardar, M.U., Afaq, N., Hoque, K.A., Johnson, T.T., Hasan, O.: Probabilistic formal verification of the SATS concept of operation. In: NASA Formal Methods, vol. 9690, pp. 191–205. Springer (2016)
52. Segala, R., Lynch, N.: Probabilistic simulations for probabilistic processes. Nord. J. Comput. **2**(2), 250–273 (1995)
53. Shortle, J.F., Xie, R., Chen, C., Donohue, G.L.: Estimating collision probabilities of landing airplanes at non-towered aiports. In: Transportation Research Board (2003)
54. Siminiceanu, R.I., Ciardo, G.: Formal verification of the nasa runway safety monitor. Int. J. Softw. Tools Technol. Transf. **9**(1), 63–76 (2007)
55. Umeno, S., Lynch, N.: Proving safety properties of an aircraft landing protocol using I/O automata and the PVS theorem prover: a case study. In: Formal Methods, vol. 4085, pp. 64–80. Springer (2006)
56. Viken, S.A., Brooks, F.M.: Demonstration of four operating capabilities to enable a small aircraft transportation system. In: Digital Avionics Systems Conference, pp. 13.A.1–1–13.A.1–16. IEEE (2005)
57. von Essen, C., Giannakopoulou, D.: Analyzing the next generation airborne collision avoidance system. In: Tools and Algorithms for the Construction and Analysis of Systems, vol. 8413, pp. 620–635. Springer (2014)
58. Williams, D.M., Consiglio, M., Murdoch, J., Adams, C.: Flight technical error analysis of the SATS higher volume operations simulation and flight experiments. In: Digital Avionics Systems Conference, pp. 13.B.1–1–13.B.1–12. IEEE (2005)
59. Williams, D.M.: Point-to-Point! validation of the small aircraft transportation system higher volume operations concept. In: International Council of the Aeronautical Sciences (2006)