Received June 26, 2019, accepted August 12, 2019, date of publication August 14, 2019, date of current version August 26, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2935423

Model Checking Longitudinal Control in Vehicle Platoon Systems

CONG PENG[1,2], MARCELLO M. BONSANGUE[2], AND ZHONGWEI XU[1]
[1]School of Electronic Information Engineering, Tongji University, Shanghai 201804, China
[2]Leiden Institute of Advanced Computer Science, Leiden University, 2333 Leiden, The Netherlands

Corresponding author: Zhongwei Xu (xuzhongweish@163.com)

This work was supported in part by the National Science and Technology Key Plan of China under Grant 2017YFB120110508, in part by the National Natural Science Foundation of China under Grant U1734211, and in part by the National CSC Scholarship.

ABSTRACT With a steadily growing number of vehicles, our roads are getting more and more crowded. As a consequence, traffic jams are becoming common. Vehicle platoon systems form a possible solution in the short term. It consists of a number of vehicles automatically following a leader vehicle, in-line, one after another at a short but safe distance. Ideally, all vehicles have to maintain the same speed, so as to have a better usage of the road by minimizing the distance between two vehicles. In this paper we present a timed automata model of a vehicle platoon system with the goal of finding a minimal but guaranteed safe distance between two vehicles under variable speed. Contrary to other models based on cooperative adaptive cruise control, we assume no (Internet) communication among different vehicles or road system. Instead of such global perspective we rather take a local point of view: each vehicle relies on its own sensors to dynamically calculate and maintain a safe distance with the preceding member of the platoon. We use the model checker UPPAAL to verify that the system does not deadlock, and most importantly, that it is safe, avoiding crashes at all time.

INDEX TERMS Model checking, vehicle safety, modeling of platoon systems, formal verification.

I. INTRODUCTION

The impact of the growing number of vehicles on the environment and human safety makes the usage of automotive and AI systems of paramount importance. Over the last few years, several solutions have been proposed ranging from the encouragement of the use of public and clean energy transportation, shared car and the adoption of smart circulation such as smart traffic lights and intelligent roads [1]. Most of them have shown positive effects but have not solved the congestion and traffic jam problems completely due to an endless growing number of vehicles. There is an urgent need for effective solutions in the short term. Autonomous-driving technology is the most promising approach towards a more safe, fluid and naturally friendly traffic. But this technology is, however, not yet ready to be used immediately because of, among other things, of safety problems. One of the most interesting and ready to be used technology in this context is given by platoon systems.

A platoon system is a group of autonomous vehicles driving on the same direction at almost the same speed (see Fig. 1 for a demonstration of a platoon system consisting of eight cars from the PATH project).

FIGURE 1. Vehicle platoon system.

Automation makes it possible for vehicles to travel together very close yet safely. As a consequence, platoon systems make it possible for a higher roadway throughput and a better traffic flow [2]. Furthermore, because less acceleration

The associate editor coordinating the review of this article and approving it for publication was Shichao Liu.

is needed, platoon systems reduce the consumption of fuel, contributing to less pollution and increasing the safety and comfort of the driver.

Speed and distance control of each vehicle is done through a fully automated *longitudinal control system*, which maintains a safe distance with respect to the vehicle in front. As every control system consists of a cycle of three phases: *Sensing*, *Planning* and *Acting*. A sensor continually checks the space and speed of the vehicle and passes these data to the control unit. The control unit processes the data and plans an action that is fed to either the brake or throttle actuators. The main challenge is to avoid collision, a goal that, however, cannot be fully guaranteed (e.g. a vehicle can hardly be responsible for being hit from the back). It is therefore reasonable to assume that platoon systems should have three orders of magnitude less collisions than usual [3], meaning that they should avoid collisions for which the vehicle is responsible. Machine learning or data-driven algorithms for the planning phase that guarantee such a level of safety would need much more data than is available today. The alternative that we explore in this paper is to formally verify safety of a model of the control algorithm.

Besides the longitudinal control system, every vehicle in a platoon system also has a *lateral control system* for following the vehicle in front while maintaining constant distance from the center of the lane. It uses sensors to determine the position of the vehicle which feed the control unit, which in turn decides on which actions need to be taken by the steering actuators. The major challenge here is to guarantee that the system work in case of low visibility.

The second problem we attack in this paper is scalability, by proving platoon safety independently for each vehicle in the platoon. Contrary to our model, existing formal models of platoon systems assume that vehicles communicate with each other and the leader through dedicated communication protocols [4] such as DSRC (Dedicated Short Range Communications). But the limited communication range of DSRC drastically limits the length of the platoon [5]. To solve this problem [6] uses a decentralized platoon maintenance algorithm, in which every vehicle only communicates with its neighbor so that the limit of platoon length can be broken, but safety guarantee of the longitudinal control algorithm has not been formally verified.

Besides the above vehicle to vehicle (V2V) communication, few models concentrate on the vehicle to infrastructure (V2I) communication, relying on road side communication unit (RSU). Examples include [7]–[9] which discuss the design of safe and efficient controllers for regulation of vehicles in automated highways using game theoretic methods. While promising, V2I communication requires high cost equipment to mitigate communication deficiencies in platoon driving scenarios: the limited connection time [10], unfairness in service time [11] and high transmission error [12]. Our model is simpler, does not rely on any external communication, and is easily implementable through standard distance sensor equipment.

The third problem we need to focus on is the longitudinal control problem, which has an important implication in current platoon maintenance research: either by relying on wireless network communication or on prediction and control methods. In [9] a multi-object control planning is introduced, taking into account not only the leading vehicle, but also shared situational knowledge between vehicles and infrastructure, so that the system is able to track vehicles directly ahead.

Most of the existing formal models for longitudinal control units are based on the so-called string model, a mathematical model suitable for studying stability problems. For example, [13] investigates sufficient conditions for stability in terms of bilinear matrix inequality, based on vehicles that use wireless communication in a platoon. Also [14] studies stability of a longitudinal control system designed for vehicles equipped with an adaptive cruise control. While stability is shown to be possible at low velocity, verification of safety is not studied.

A. OTHER MODELS OF PLATOON SYSTEMS

Despite the importance of safety, most of the current work on longitudinal control units concentrates on optimization and control methods. For example, [15] presents a cooperative shared control driver assistance system which supports the driver in the longitudinal vehicle control. Reference [16] proposes a class of robust longitudinal control units for each vehicle except the leading vehicle, based on an ideal swarm model. The focus is on the collision avoidance problem, but the model is not formally verified. Reference [17] presents a distributed H-Infinity control method for multi-vehicle systems with identical dynamic controllers and rigid formation geometry. Also this work pays attention only to robust and string stability rather than safety property.

Besides optimization and control methods, [18] evaluates longitudinal safety of platoon systems in dedicated lane on highways. The analysis is based on simulation and it is interesting because it shows that platoon systems significantly improved the longitudinal safety when compared to the base condition.

Similar to us, [19] proposes a virtual spacing policy to replace the unknown spacing policies in platoon systems, but without formal verification of safety properties. The latter is exactly the goal of this paper.

B. OVERVIEW OF OUR CONTRIBUTION

For a clear description of our approach, we summarize our threefold contribution.

I We introduce the requirements for formal longitudinal control of automotive platoons. This allows us to better understand the functioning of platooning mechanisms and, more importantly, to verify essential properties such as the functional correctness and safety of the mechanisms. An important aspect of the mechanisms is that we reserve space for vehicles to join or leave the platoon, while ensuring safety.

II We use a variable distance policy inside the longitudinal control of platoon system, thus optimizing the safe distance based on the speed of the vehicle. And the longitudinal control unit algorithm we used for the distance policy is not rely on classical prediction algorithms, instead, it is based on a dynamical division of the distance in front of the car into five zones.

III Compared with the existing research in longitudinal control of platoon system, we mainly focus on the key point of the whole system, the safety property. In order to make a verification and avoid state explosion, we look only locally one vehicle at the time, meaning that the entire platoon is safe if all its members inside the platoon follow the same control algorithm. We use a timed-automata abstraction and verify the system using the model checker UPPAAL. The model checker UPPAAL can guarantee the verification of time temporal properties for any possible execution of the system unless it stops running. This verification has been carried out under the assumptions that the vehicle control is functioning well and all the vehicle parameters we used, such as the length and type of the vehicles are correct. And it is worth noting that we make the verification based on an abstraction of the real system, as the real system is hard to be verified. Through it is important to continuously improve the precision of the abstraction.

C. PLAN OF THE PAPER

The remainder of the paper is organized as follows. In Section 2 we introduce platoon systems, time automata, and the model checker UPPAAL. In Section 3 we describe the model of the longitudinal control systems in UPPAAL, that we subject to verification in Section 4. We conclude in Section 5, with some remarks and future direction.

II. PRELIMINARIES

In this section, we briefly introduce platoon systems and why it is important to verify their longitudinal control unit. To this end, we use the abstract model of timed automata and the model checking tool UPPAAL based on time automata. Both are briefly we introduced here.

A. PLATOON SYSTEMS

A platoon system is a driving pattern for a line of vehicles driving on the same direction and almost at the same speed one after another.

As shown in Fig. 2, three main scenarios play a role in a platoon system on a highway: maintenance, joining and splitting [20]. In the first scenario the goal of each vehicle is to follow the preceding one maintaining a small but safe distance, except for the first vehicle that drives safely to a certain goal.

Platoon splitting happens when one vehicle leaves the platoon, for example it reaches its own destination. In this case either the preceding car catches up with the new vehicle in front or it becomes the leader of a new platoon.

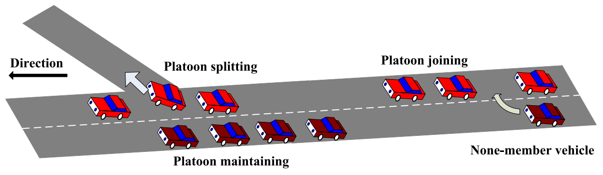


FIGURE 2. Scenarios in the platoon system.

Platoon joining consists of the reverse problem: a non-member vehicle wants to join a platoon. In this case space must be created in the platoon and when it is safe the new vehicle merges with the platoon should be decided.

In this paper, we mainly focus on the platoon maintenance problem. The other two scenarios can be easily added to our model as parts of the future work. Other scenarios, like lane merging and platoon in an urban environment are more complicated and will not be treated here.

In the context of a platoon maintenance problem, there are several aspects that need to be considered:

Vehicle type: The type of vehicles in a platoon has clear effects on the parameters of a platoon system. For instance, one needs to consider the length of the vehicle l, as it is not feasible to consider an uniform length for all vehicles in a platoon (trucks have a significant greater length than cars). Also the maximum acceleration A and maximum speed V_{max} of each vehicle needs to be considered. Further, in order to automatically maintain a safe distance all vehicles in a platoon are assumed to have some sensors continuously measuring the distance with the preceding vehicle. Thus other parameters need to be considered, such as the sensitivity (number of measurement per time unit) of the sensor, that, again, can be different for each vehicle. In order to reduce the number of parameters involved, we will take a local view of the system, considering length, acceleration, max velocity and sensitivity of the sensor in a single vehicle.

Spacing policy: Currently, platoon systems either use a constant distance policy [21], [22], [23] or a variable distance policy [24], [25]. The former one means that the distance between two vehicles inside the platoon must remain the same (up to some minor parameters) despite the change of the vehicle speed. Dynamic spacing policy instead postulates the possibility of changing the distance between two cars depending on a fixed control strategy. In this paper, we use a variable distance policy, thus optimizing the safe distance based on the speed of the vehicle.

Communication method: Vehicles in platoon systems are often assumed to communicate with each other. There are two different types of communication. One is between two vehicles and another is between vehicles and infrastructure. While these are promising lines of research, currently it is not possible to assume direct communication among all vehicles in a platoon because of the instability of wifi connectivity, or because of the high cost in the infrastructure. As a consequence we assume no communication inside and outside the platoon take place, but assume cheaper sensor in each vehicle to continuously measure the distance with the vehicle in front.

Control strategy: The control strategy defines the control algorithms on the vehicle inside the platoon. The verification of such algorithm is the main focus of this paper. We will discuss it in part III.

B. TIMED AUTOMATA IN UPPAAL

Timed automata [26], [27] is a finite state model for the verification of real time systems. It is based on an extension of Büchi automata with time clock and delays. Timed automata has been simplified into timed safety automata (TSA) [28] by combining time constraints with the action constraints in the transitions and by specifying progress properties as invariant conditions local to a state. Based on TSA, several model checking tools have been developed, including UPPAAL [29].

We use timed automata to discretize the sensing domain using worst case values instead of continuous actions. The advantage is a better use of resource and a shorter computational time for the verification at the cost of an approximation of the real environment. For the verification of safety of the longitudinal control systems we use the model checker UPPAAL, a tool for modeling, validation and verification of real-time systems [29]. In general, UPPAAL is appropriate for systems that can be modeled as a collection of uncertainty processes with finite control structure and real-valued clocks such as timed safety automata. Different automata communicate through channels and shared data structures. Typical application areas include real-time controllers, communication protocols, and other safety critical systems.

In UPPAAL, a system is modeled as a network of timed automata which act in parallel on a set of global and local discrete variables. The state of a system is defined by the locations of the each automata, the values of the discrete variable, and the time value of each clock.

We give the basic definitions of the syntax and semantics of timed automata, here use the following notations:

Let T be a set of clocks and $C(T)$ a set of conjunctive convex constrains clocks, such as $m \leq t$ or $m-n \leq t$ where m and n are clocks and t is a natural number representing discrete time. A *timed automaton* is a finite directed graph with conditions and resets of valued clocks, i.e. a tuple (L, l_0, E, A, I) where

- L is a finite set of locations,
- $l_0 \in L$ denotes the initial location,
- A is a set of actions, including co-actions for communication and and internal-action for manipulating discrete variables,
- $E \subseteq L \times A \times C(T) \times 2^C \times L$ are the edges between locations which are labelled with an action, a time constraint, and a set of clocks to be reset, and
- $I : L \rightarrow C(T)$ is a function mapping invariant constraints to locations.

The basic idea is that while time passes, an automaton can stay in a certain state until the invariant property holds. If the time constraint of an edge leaving from a location is true, then without any passing of time its action is executed, and all clock in the set labelling the edge are reset to 0. Finally, the execution of an edge moves the automaton to the target location of the edge itself.

As a consequence, timed automata is highly non deterministic, and even if their behaviour can be described by a finite set of location, the system itself can have an infinite set of different states.

Model checking is a technique for automatically verifying formal properties of systems by exploring *all* its state. In the case of timed automata, this is rather challenging because exploring all states results in an infinite transition system. UPPAAL uses an exact finite state abstraction based on convex polyhedra as defined by clock constraints.

UPPAAL implements a symbolic exploration algorithm based on the symbolic semantics of networks of timed automata. In the case of reachability or liveness properties UPPAAL computes successor states symbolically as follows: when an edge is executed, the next state is delayed infinitely (if possible) and the invariant of the state is applied. This computes all the (timed) successor states w.r.t. a given step. Thus computing successors in UPPAAL refer to trying all possible actions followed by delay.

The properties that can be checked by UPPAAL, are defined in a subset of Timed CTL (Computation Tree Logic). We give here just few examples of temporal properties, where r and s are basic state properties on clock and discrete variables:

Possibly The property $E\Diamond r$ holds if there exists an execution with at least one state satisfying the property r.

Invariantly The property $A\Box\ r$ holds if every state in every execution satisfies the property r.

Potentially always The property $E\Box\ r$ holds if every state of some execution satisfies the property r.

Eventually The property $A\Diamond r$ holds if for every execution there is a state which satisfies r.

Leads to $r \rightsquigarrow s$ holds in every execution for which whenever r holds eventually s will holds too. It can also be expressed as $A\Box(r \Rightarrow A\Diamond s)$.

Deadlocks An execution deadlock when after some state there is no successful action that can be taken. This is expressed in UPPAAL by $E\Diamond \mathit{false}$.

III. MODELING OF PLATOON

The first car of a platoon system is the leading vehicle. Except for the leading vehicle, all other vehicle members of the platoon are controlled autonomously according to the following functionalities:

- longitudinal control, which takes care of speed and distance,
- Lateral control, which takes care of lane tracking, lane changing and steering.

Other functionalities such as platoon formation and platoon splitting are part of the maneuver coordinator. We focus only on a fully automatic longitudinal control. It should fulfill the following requirements (driving policies) based on three "common sense" rules:

- The vehicle should keep a safe distance from the vehicle in front so to be able to stop in time in case of need;
- The vehicle should leave (other) vehicles the possibility to cut-in, to enter or to exit the platoon;
- The vehicle should respect the driving rules (for instance max speed).

A. KEEP A SAFE DISTANCE

Full safety for a vehicle, is unfeasible and even impossible. For example there are 108 meters necessary to fully stop a light vehicle driving at 130 km/h with max brake deceleration of 6.096 m/s^2 at response time of 1/100 seconds. Also it is impossible to avoid that other vehicles can always hit the one we are considering from behind or laterally. We therefore focus on just keeping a safe distance from the vehicle in front assuming that it will not crash (stopping in almost zero time).

B. LOCAL RESPONSIBILITY

We focus on a set of local responsibility:

- No inter-vehicle communication: vehicles in a platoon system have no communication with other vehicles or leader;
- No high-way communication: vehicles in a platoon systems have no communication with highway infrastructure;
- No constant speed and spacing: vehicles in a platoon systems have variable speed and spacing parameters so to allow cut-in of a vehicle, i.e. joining, leaving, or passing through a platoon.

C. RESPECT OF DRIVING RULES

In this context, the most important driving rule that needs to be respected is to be within the speed limit. A consequence is that leadership becomes dynamic. For example, after a cut-in in a platoon driving at maximum speed, it is impossible for the vehicle which allows the cut-in to catch up with the vehicle in front. In this case the current vehicle became the leader of a new platoon system.

We can assume that some parameters are fixed by some regulations, while other may depend on each vehicle. Below we show a table with the parameters we consider (rightmost column) and their comparison with standard values for a car driven by human (non-professional pilot):

The control unit of each vehicle in a platoon systems gets input through a sensor measuring the distance with the vehicle in front. On receiving this information the control unit regulates the speed via instructing the throttle or the brakes resulting in an accelerating or decelerating action. Through this feedback control, the distance between each vehicle always stays inside a safe range.

TABLE 1. **Comparison of parameters.**

	Human-based	**Machine-based**
Min reaction time	1 - 2 s	1/100 - 1/10000 s
Max velocity	130 km/h	130 km/h
Max acceleration	6.4 m/s^2	6.4 m/s^2
Max brake	6.1 m/s^2	9.75 m/s^2
Comfortable brake	4.6 m/s^2	4.6 m/s^2
No acceleration	0.75 m/s^2	0.75 m/s^2

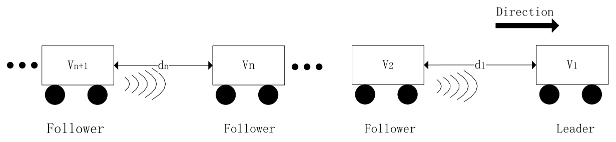


FIGURE 3. **Model of the platoon.**

Let us consider a platoon system where V_1 is the leader vehicle and $V_2, \ldots V_n$ and V_{n+1}, the follower vehicles, as shown in Figure 3.

Definition 1 (Safe Longitudinal Distance): Let d denote the actual current distances between the front of vehicle V_{i+1} and the back of vehicle V_i, with $0 \leq d \leq d_{max}$, and d_{max} the maximum allowable distance between two vehicles inside a platoon [3]. The distance d is safe if $d > 0$ for any reasonable deceleration (bounded by the maximum usage of the brakes B_{max}) and response time ρ of V_{i+1} with velocity less than v_{max}.

This means that the vehicle V_{i+1} will not collide with V_i even in the worst scenario when V_i is braking at the maximum level and V_{i+1} is moving at the maximum speed. Note that the definition of safe longitudinal distance relies on both local parameters ρ, B_{max}, and v_{max}, and the velocity v_i ($v_i \leq v_{max}$) of the front vehicle V_i as well.

The following lemma calculate the safe longitudinal distance.

Lemma 1: Let V_{i+1} be the vehicle behind vehicle V_i in the longitudinal direction, let the rest parameters be as in the Definition 1. The minimal safe longitudinal distance between the front of vehicle V_{i+1} and the back of vehicle V_i is:

$$d_{min} = \left[v_{max}\rho + \frac{v_{max}^2}{2B_{max}} - \frac{v_i^2}{2B_{max}} \right]_+ .$$

Proof: Let d be the initial distance between V_{i+1} and V_i and assume the worst case scenario that V_{i+1} is moving at maximum velocity v_{max}. The velocity of the front vehicle V_i decreases with a rate B_{max} until arrives to zero (or else it collides with something in front). While, vehicle V_{i+1} keeps the velocity of v_{max} for ρ time units and then slows down with the rate of B_{max} (we assume all the vehicles have the same maximum negative acceleration) until totally stops (or otherwise a crash happens). The distance between the two vehicles will be monotonically decreasing until both

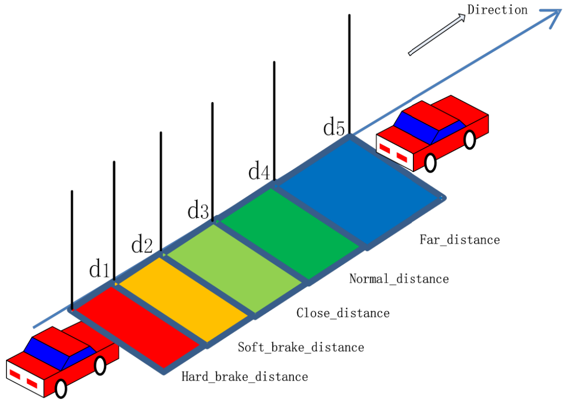


FIGURE 4. **Divided distance of platoon system.**

vehicles stop. A collision is avoided if the distance between the two vehicles is always larger than zero. We thus have

$$d + \frac{v_i^2}{2B_{max}} - \left(v_{max}\rho + \frac{v_{max}^2}{2B_{max}}\right) > 0$$

from which we get the minimum distance $d_{min} < d$, thus concluding the proof.

The minimal distance d_{min} above is based on the maximal use of the brakes by the vehicle following another one which is also using its brake maximally. For instance, for two vehicles, one driven by a human with a reaction time between 1 and 2 seconds, following another at a speed of 130 km/h, the minimal safe longitudinal distance is between 36 and 72 cm.

In reality however, the passengers of the vehicles in a platoon want to have a comfortable experience, with a minimal usage of the maximal brakes. For this reason, in our longitudinal control algorithm we divide the distance between two vehicles in a platoon into 5 zones d_1 to d_5 (see Figure 4) that dictate the actions of the vehicle depending on his continuously measured longitudinal distance d with the front vehicle. We assume this distance d to be measured by a sensor of V_{i+1}, and it needs not rely on classical prediction algorithms.

- When $0 < d \leq d_1$ the vehicle V_{i+1} is in the `Hard _brake zone`, meaning that brakes should be used as hard as possible in order to avoid collision (i.e. $d = 0$) with the front vehicle. In this situation the throttle is not used, we assume that the vehicle decelerates at 6 m/s^2, the maximal negative acceleration acceptable (but not necessarily comfortable) by humans, with the lateral control system taking actions to avoid and correct lateral deviations. Notice that d_1 can be even smaller than the minimal safe distance because the velocity of V_{i+1} can not be maximal in this zone, as it will be decelerating in the previous zones. For instance, in this paper we set d_1 = 20 cm for a vehicle with speed 130 km/h, whereas we have seen that in this case the minimal safe distance is 36 cm.
- If $d_1 < d \leq d_2$ then V_{i+1} is in the `Soft_brake zone`. In this case the control system sends to the actuators the message to use the brakes in a comfortable manner and to not use the throttle. In this paper, considering UPPAAL can only accept integer value, we assume that this means a negative acceleration of 4 m/s^2, so that driver and passenger can have a relatively comfortable ride experience.
- When $d_2 < d \leq d_3$ the vehicle V_{i+1} is in the `Close distance zone`. Inside this range V_{i+1} stays at a safe distance without the need to use the brakes but only by releasing the throttle. We assume that under this circumstance the vehicle decelerates at 1 m/s^2 because of the friction, so that the speed will slow down smoothly and the energy will be saved.
- If $d_3 < d \leq d_4$ then V_{i+1} is in the `Normal distance zone`, using no brakes and remaining at a constant speed (i.e. with acceleration of 0 m/s^2).
- When $d_4 < d \leq d_5$ the vehicle V_{i+1} is in the `Far distance zone`, which means it has to accelerate (if allowed by the speed limit) to get closer to the platoon. Clearly brakes are not needed. For simplicity and considering UPPAAL can only accept integer value, we assume that V_{i+1} accelerates at 6 m/s^2, an acceleration considered acceptable by humans.

The goal of this paper is to identify $d_1 < d_2 < d_3 < d_4 < d_5$ such that no collision with the front car is guaranteed (i.e. it is never the case that the actual distance $d = 0$). We model a control strategy based on the above description and verify its safety in UPPAAL as a parallel composition of a processes describing the behaviour of each follower vehicle in the platoon along with an extra process describing the behaviour of the sensor in front of each follower vehicle.

In order to avoid state explosion, we look at only one local vehicle at the time, meaning that the entire platoon is safe if all its members inside the platoon follow the same control algorithm. The local model is a parallel composition of two timed automata, the *control_system* and the *sensor*. The UPPAAL model of the *control_system* consists of seven states: one for each zone where the vehicle current is, one representing a car that is stopped, and another for a car becoming a leader of the platoon. Each state has a transition looping to itself representing the interaction with *sensor* through channel Dist to receive the current distance. When the current distance changes, transitions may happen.

In order to build a specific UPPAAL model, and because UPPAAL can only accept integer values, we use the following parameters:

- Braking has three values 0, 1 and 2, corresponding to *no brake* (-1 m/s^2), *soft brake* (-4 m/s^2) and *hard brake* (-6 m/s^2);
- We represent the throttle as the integer value *PosAcc* ranging between 0 and 6 (at 0 the vehicle decelerate by friction, at 6 the velocity will reach and eventually remain at the maximum speed). Each increase of *PosAcc*

Algorithm 1 Platoon Model Declaration

```
begin
    $int\ B = 0;$
    $chan\ Dist;$
    $int\ x = 0;$
    $const\ int\ l = 500;$
    $const\ int\ d_1 = 20;$
    $int\ d_2 = 210;$
    $int\ d_3 = 220;$
    $int\ d_4 = 220 + l;$
    $int\ d_5 = (220 + l) * 2 + l;$
    $int\ d = 220;$
    $int\ V = 36;$
    $const\ int\ V_{max} = 36;$
    $const\ int\ t = 1;$
    $clock\ clk1;$
    $int\ z;$
end
```

represents a positive acceleration of 6 m/s^2, until the max speed is reaching;

- We use *clk*1 in the sensor model to indicate the global time of the system, measured in *Ticks* (T), the time needed by the sensor to complete a cycle in sending the distance information to the control system. We set this to 1/100 second, a speed easily satisfied by current sensor technology.

Because UPPAL do not accept rational numbers, we use centimeters as unit of distance rather than meters, as declared in Algorithm 1.

B is the brake value, *Dist* is the communication channel between the control system and the sensor, d is the distance to the front vehicle. As this depends on the environment, we increase or decrease d by a randomly, but reasonable, generated value x. Finally we assume that all the vehicles are at most 5 meters long, thus the length l is initialized to 500 centimeters. We have seen that d_1 is set to 20 cm, but d_2 and d_3 will vary depending on the speed of the vehicle. After running several experiments in UPPAAL to find reasonable values for d_2 and d_3, we set them to 210 and 220 centimeters, respectively. The distances d_4 and d_5 depend both on d_3 and l, so as to allow enough space for non-members vehicles to join the platoon system. The initial distance between a rear vehicle and the preceding one is set to 220 centimeters, which is the threshold between the close distance zone and the normal distance zone. By using model checking these values guarantee that a vehicle in the platoon does not collide with the one in front under reasonable circumstances, for instance, the front vehicle does not fully stop in zero time as a consequence of a crash. Finally, according to the maximum speed limit of the highway, we assume the V_{max} and the initial velocity to be 36 cm/T, in addition, the period of sensor is set to t T (here $t = 1$).

Next we explain the states of the local vehicle model in UPPAAL. Assume a vehicle V_{i+1} stays in the $Normal_dist$ state with the preceding vehicle V_i, and the interval distance is d. When $d < d_3$, the transition condition is fulfilled and thus the control system jump to the state $Close_dist$ assigning *PosAcc* to 0, as shown in Fig. 5. If the distance reduces even more, then the control system will go into the $Hard_brake$ state. However in the case the front vehicle accelerates, the distance will increase and eventually the control system will return into the $Normal_dist$ state again. When the system is in the state $Hard_brake_dist$, it will increase braking, thus using the brakes maximally. In this state, there are also two possibilities. One possibility is to jump back to the previous state, and another one is that after several rounds of decreasing velocity by $V := V - 6$ due to the hard brake value -6 m/s^2, the vehicle will fully stop, and the velocity will become 0 cm/T, thus the vehicle will enter state *Car_stopped_dist*. If the distance remains $d < d_1$, vehicle V_{i+1} will keep inside *Car_stopped_dist* state, until d becomes $d >= d_3$, thus the control system will jump back to *Normal_dist* with the updates $B := 0$ and $PosAcc := 1$, then the vehicle will make an acceleration with $V := V + 3$ until the velocity reach the maximum value.

Above we have given a explanation to the state transition in the loop of distance reducing. Now we explain the loop of distance increasing. When V_i makes a sudden acceleration, the distance may fulfill the condition $d > d_4$, thus V_{i+1} will accelerate through an update of $PosAcc := PosAcc + 1$, and correspondingly the control system will enter Far_dist state. There are two loops on this state, where $PosAcc := PosAcc + 1$ and $V := V + 6$ will be taken until *PosAcc* arrives the value of 6. If the velocity of V_{i+1} reaches the maximum value and still can not catch up with V_i, after matching condition $d > d_5$, the control system will make a transition to *New_leader* state, which means the distance is too far for vehicle V_{i+1} to remain inside the platoon system and thus becomes a new platoon leader or a non-member vehicle.

Next we explain the UPPAAL model of *sensor*. As shown in Fig. 6, there are in total three states: *Car_Moving*, *Car_Stopped* and *Nothing_in_front*. Every state has a self-loop which includes synchronizing on the *Dist* channel. When the sensor receives a request from the control system, the synchronization will act as a feedback that sends the distance information. And based on the distance message, the state of control system will make a state transition or remain in the original state. When the rear vehicle is moving forward, in t T period of time the relative distance between the rear vehicle and the preceding one can be increasing or decreasing. Based on the following three formulas we express this situation:

$$x : int[0, V_{max} * t]; \quad z := x - V * t; \; d := d + z;$$

when the rear vehicle totally stops, the velocity becomes zero, so that the relative distance can only keep increasing or be the same value. Thus based on the model, we use the next two

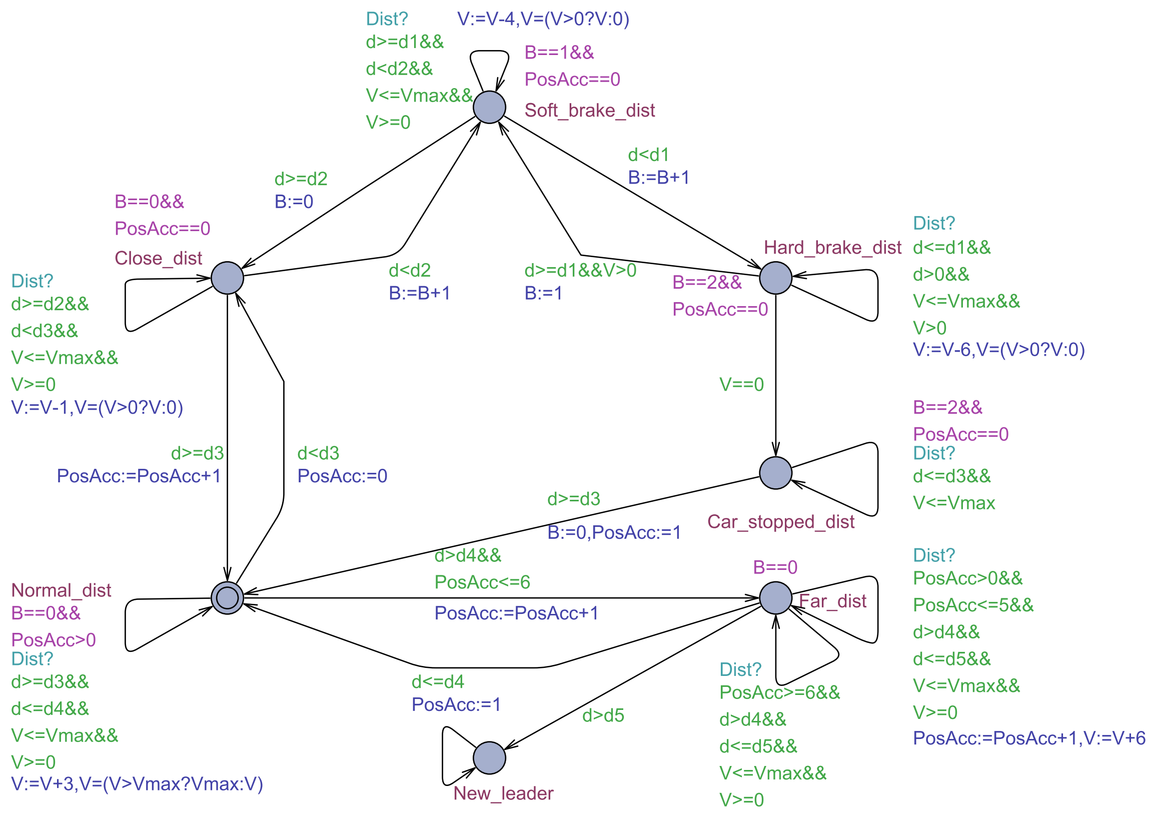


FIGURE 5. Local vehicle model in UPPAAL.

formulas to calculate it:

$$x : int[0, V_{max} * t]; \quad d := d + x;$$

In both situations x is a random value between 0 and $V_{max} * t$ to show the change of distance. If $d > d_5$, the vehicle has already left the platoon, thus the sensor will not receive distance value request from control system, and correspondingly it will not send back the distance information due to save the energy. In this way, the rear vehicle will enter a *Nothing_in_front* state, which automatically becomes a new platoon leader.

IV. VEHICLE MODEL VERIFICATION OF PLATOON SYSTEM USING UPPAAL

As we have timed automata for modelling the control system and the sensor of a vehicle inside a platoon, we can carry out verification of specific properties using UPPAAL. As each vehicle among the followers has the same model, the verification can deal with arbitrary length of platoon by simply adding the number of followers. Here we just verify two properties through the local control system model and the sensor model. These two properties are the **deadlock freedom** problem and the **guarantee safety** problem. The former is to ensure the system never blocks and the latter is to make sure the distance between two vehicles is always strictly positive. Here using the "**brute-force**" approach [3] that check all possible states, UPPAAL needs 3 to 5 minutes for computing the result due to the high amount of data. For the sake of simplicity, the particular parameter for verification has been used. In particular, we set the time period to 1 T and V_{max} to 36 cm/T. If the *control_system* and *sensor* models are correct, this property should be satisfied:

$A \square$ *not deadlock*;

We conduct the verification through the model checking tool UPPAAL. All experiments are carried out on a machine with Intel Core i7 3.40 GHz CPU and 7.7 GB memory under Windows. In order to get accurate results and avoid fluctuation, each model performs 10 runs with an average cutoff time of 186.65 s. Finally, if the verification process finishes successfully, then it means that the "no deadlock" property is satisfied. The other practical property is to guarantee that the distance between rear vehicle and preceding vehicle is always strictly larger than zero. To express this property, we use the

TABLE 2. Definitions of time period and velocity.

Time Period	Velocity
1 T (0.01 s)	36 cm/T = 129.6 km/h
2 T (0.02 s)	24 cm/T = 86.4 km/h
3 T (0.03 s)	18 cm/T = 64.8 km/h
4 T (0.04 s)	15 cm/T = 54 km/h
5 T (0.05 s)	13 cm/T = 46.8 km/h

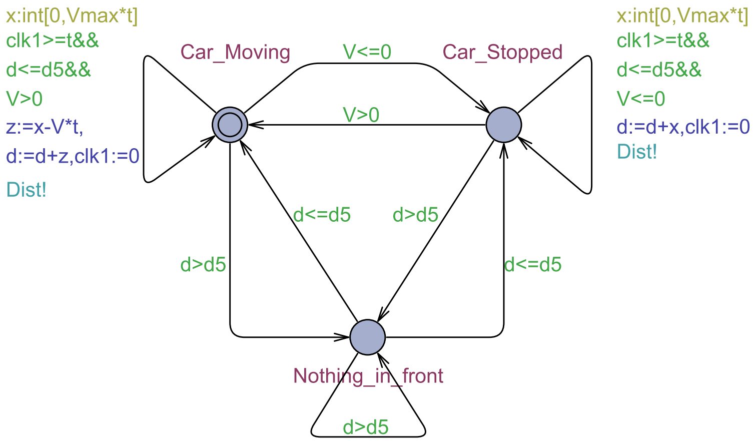


FIGURE 6. Local sensor model in UPPAAL.

TABLE 3. Definitions of distance allocation and velocity.

Time Period	Distance Allocation					Velocity
	d_1	d_2	d_3	d_4	d_5	
1 T (0.01 s)	20 cm	210 cm	220 cm	790 cm	2080 cm	36 cm/T = 129.6 km/h
1 T (0.01 s)	20 cm	130 cm	200 cm	740 cm	1980 cm	30 cm/T = 108 km/h
1 T (0.01 s)	20 cm	70 cm	190 cm	630 cm	1760 cm	24 cm/T = 86.4 km/h
1 T (0.01 s)	20 cm	40 cm	120 cm	620 cm	1740 cm	18 cm/T = 64.8km/h
1 T (0.01 s)	20 cm	30 cm	40 cm	540 cm	1580 cm	12 cm/T = 43.2 km/h

following formula, which uses the property of invariant in UPPAAL:

$A \Box d > 0$;

Similar to the process of verifying the "no deadlock" property, we perform 10 runs for each model with a duration time of 153.85 s. With the proof of this formula, the distance between each two neighboring vehicles will be always larger than zero, which means the crash will never happen inside the platoon based on this model.

In this experiment, the velocity is set to 36 cm/T. Under this experimental setup, we set a group of time periods of the sensor which all satisfy the verified properties. In practice, with the larger period of the sensor, the smaller amount of energy will be required. As a result, we choose the largest one among all values of time periods of the sensor. But here comes a problem – when the velocity slows down, could the time period be even larger under the safety requirement? In order to investigate the result in more depth, we set a group of velocities to indicate the decreasing trend of velocity and conduct the model checking to evaluate whether they can fulfill the safety requirements and find out the suitable time period under each velocity. Under four independent experiments similar to the initial model checking model, we get the results below:

According to the results in Table 2, we know that the time period will be increased with the decrement of the velocity, which also complies with the facts. As velocity slows down, the distance between two vehicles can be larger during one period of time, so the requirements for the frequency of the sensor will be reduced too. Based on the period and velocity setups in Table 2, we conduct a verification of a full platoon system with one leader and two followers, which proves and guarantees that no deadlock and no crash will happen. In the future we can design a variable frequency sensor based on velocity, so, to a certain degree, the energy and consumption can be saved and reduced.

The above table shows the relationship between the time period and the variable velocity. In order to guarantee safety, we adopt the distance allocation method under the maximum allowed velocity of 36 m/s. Thus this group of distance parameters meet the safety requirements of the platoon system at all velocities. If we fix the time period of the sensors, with the decrement of the velocity of the platoon, we can get several groups of distance allocation through several rounds of model checking, as shown in table 3 below.

From table 3 the time periods of sensors in control system have been fixed to 1 T, and the d_1 to d_5 are based on the longitudinal control unit algorithm, also mentioned in Section III. As the velocity slows down, the allocation values of safe longitudinal distance become smaller too. Thus when the whole platoon moves forward in a rather low velocity, the distances among any two neighboring vehicles inside the platoon can be smaller, which can bring a higher roadway throughput.

V. CONCLUSION

Our goal of this formal model is to guarantee that no accident happens inside this platoon system in one hundred percent. Contrary to traditional models based on cooperative adaptive cruise control, we assume no communication among vehicles and the needed distance data can be obtained from sensors. Hence, in our method, the sensors are of great importance. While we can not make sure that the sensor never makes a mistake – for instance safety mistakes may occur under the following two cases: safety-critic miss and safety-critic ghost [3]. Safety-critic miss means a dangerous situation sensed as non-dangerous circumstance by the system which will lead to the neglect of danger and further cause serious accidents. Safety-critic ghost means non-dangerous situation sensed as a danger by the system which will cause unnecessary braking and further affect the comfort experience of the passengers inside the vehicle. The solution is that we use several independent and distributed sensing systems which form a redundant mechanism. In this way, the accuracy of sensing system will be greatly improved.

In this paper, we give a time automata model to the vehicle platoon system and verify its safety. Compared to the existing research works, our main contribution is that we mainly focus on the safety property of the longitudinal control problem in platoon system, where we use UPPAAL as the model checking tool. Different from simulation, we make a rigorous proof of the safety property through the process of model checking which all states have been traversed. As a result, we prove that no crash and no deadlock happen.

In future work, we plan to study if the entire system is stable in a safe state. In addition, if a crash of the leading vehicle causes an abrupt and instantaneous stop, we would like to study how many following vehicles behind it will crash due to wave effect. In addition, we would also like to find the safest position in a platoon based on the control strategy in this paper, in order to improve the ability of the vehicle platoon control system.

REFERENCES

[1] S. Mouelhi, D. Cancila, and A. Ramdane-Cherif, "Distributed object-oriented design of autonomous control systems for connected vehicle platoons," in *Proc. Int. Conf. Eng. Complex Comput. Syst. (ICECCS)*, Nov. 2017, pp. 40–49.

[2] B. van Arem, C. J. G. van Driel, and R. Visser, "The impact of cooperative adaptive cruise control on traffic-flow characteristics," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 4, pp. 429–436, Dec. 2006.

[3] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "On a formal model of safe and scalable self-driving cars," *CoRR*, vol. abs/1708.06374, pp. 1–33, Aug. 2017.

[4] M. Kamali, L. A. Dennis, O. McAree, M. Fisher, and S. M. Veres, "Formal verification of autonomous vehicle platooning," *Sci. Comput. Program.*, vol. 148, pp. 88–106, Nov. 2017.

[5] C. Campolo and A. Molinaro, "Multichannel communications in vehicular ad hoc networks: A survey," *IEEE Commun. Mag.*, vol. 51, no. 5, pp. 158–169, May 2013.

[6] A. Sarker, C. Qiu, and H. Shen, "Quick and autonomous platoon maintenance in vehicle dynamics for distributed vehicle platoon networks," in *Proc. IoTDI*, 2017, pp. 203–208.

[7] A. Puri and P. Varaiya, "Driving safely in smart cars," in *Proc. Amer. Control Conf.*, Jun. 1995, pp. 3597–3599.

[8] J. Lygeros, D. N. Godbole, and S. Sastry, "Verified hybrid controllers for automated vehicles," *IEEE Trans. Autom. Control*, vol. 43, no. 4, pp. 522–539, Apr. 1998.

[9] B. Schaeufele, O. Sawade, D. Pfahl, K. Massow, S. Bunk, B. Henke, and I. Radusch, "Forward-looking automated cooperative longitudinal control: Extending cooperative adaptive cruise control (CACC) with column-wide reach and automated network quality assessment," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst.*, Oct. 2017, pp. 1–6.

[10] J. Zhao, T. Arnold, Y. Zhang, and G. Cao, "Extending drive-thru data access by vehicle-to-vehicle relay," in *Proc. VANET*, 2008, pp. 66–75.

[11] V. P. Harigovindan, A. V. Babu, and L. Jacob, "Ensuring fair access in IEEE 802.11 p-based vehicle-to-infrastructure networks," *EURASIP J. Wireless Commun. Netw.*, vol. 2012, no. 1, p. 168, 2012.

[12] D. Jia, R. Zhang, K. Lu, J. Wang, Z. Bi, and J. Lei, "Improving the uplink performance of drive-thru Internet via platoon-based cooperative retransmission," *IEEE Trans. Veh. Technol.*, vol. 63, no. 9, pp. 4536–4545, Nov. 2014.

[13] C. Latrech, A. Chaibet, M. Boukhnifer, and S. Glaser, "Integrated longitudinal and lateral networked control system design for vehicle platooning," *sensors*, vol. 18, p. 3085, Sep. 2018.

[14] C. Chang and Z. Yuan, "Combined longitudinal and lateral control of vehicle platoons," in *Proc. Int. Conf. Comput. Syst., Electron. Control (ICCSEC)*, Dec. 2017, pp. 848–853.

[15] S. Mosbach, M. Flad, and S. Hohmann, "Cooperative longitudinal driver assistance system based on shared control," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2017, pp. 1776–1781.

[16] X. Zhao, Y. H. Chen, and H. Zhao, "Robust approximate constraint-following control for autonomous vehicle platoon systems," *Asian J. Control*, vol. 20, no. 4, pp. 1611–1623, 2018.

[17] S. E. Li, F. Gao, K. Li, L.-Y. Wang, K. You, and D. Cao, "Robust longitudinal control of multi-vehicle systems—A distributed H-infinity method," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 9, pp. 2779–2788, Sep. 2018.

[18] M. S. Rahman and M. Abdel-Aty, "Longitudinal safety evaluation of connected vehicles' platooning on expressways," *Accident Anal. Prevention*, vol. 117, pp. 381–391, Aug. 2018.

[19] G. Rödönyi, "An adaptive spacing policy guaranteeing string stability in multi-brandad hocplatoons," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 6, pp. 1902–1912, Jun. 2018.

[20] D. Jia, K. Lu, J. Wang, X. Zhang, and X. Shen, "A survey on platoon-based vehicular cyber-physical systems," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 263–284, 1st Quart., 2016.

[21] P. Seiler, A. Pant, and K. Hedrick, "Disturbance propagation in vehicle strings," *IEEE Trans. Autom. Control*, vol. 49, no. 10, pp. 1835–1841, Oct. 2004.

[22] L. Xiao and F. Gao, "Practical string stability of platoon of adaptive cruise control vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1184–1194, Dec. 2011.

[23] A. Kesting and M. Treiber, "How reaction time, update time, and adaptation time influence the stability of traffic flow," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 23, no. 2, pp. 125–137, 2008.

[24] J. Zhao, M. Oya, and A. El Kamel, "A safety spacing policy and its impact on highway traffic flow," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2009, pp. 960–965.

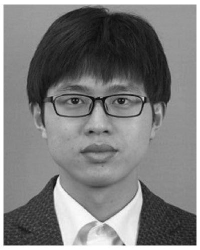
[25] L. Xu, L. Y. Wang, G. Yin, and H. Zhang, "Communication information structures and contents for enhanced safety of highway vehicle platoons," *IEEE Trans. Veh. Technol.*, vol. 63, no. 9, pp. 4206–4220, Nov. 2014.

[26] R. Alur and D. L. Dill, "Automata for modeling real-time systems," in *Proc. 17th Int. Colloq. Automata, Lang., Program.*, 1990, pp. 322–335.

[27] R. Alur and D. L. Dill, "A theory of timed automata," *Theory Comput. Scince*, vol. 126, no. 2, pp. 183–235, 1994.

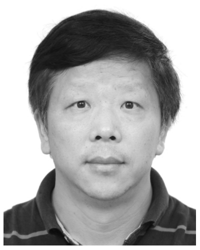
[28] T. A. Henzinger, X. Nicollin, J. Sifakis, and S. Yovine, "Symbolic model checking for real-time systems," *Inf. Comput.*, vol. 111, no. 2, pp. 193–244, 1994.

[29] J. Bengtsson and W. Yi, "Timed automata: Semantics, algorithms and tools," in *Proc. ACPN*, 2003, pp. 87–124.



CONG PENG was born in Wuhan, Hubei, China, in 1988. He received the B.S. degree in electronic information science and technology from Wuhan Textile University, Wuhan, Hubei, China, in 2010, and the M.S. degree in computer architecture from the East China Institute of Computing Technology, Shanghai, China, in 2014. He is currently pursuing the Ph.D. degree in computer science and technology with the School of Electronic Information Engineering, Tongji University, Shanghai. From October 2017 to July 2018, he was a Visiting Scholar with the Theory Group of the Leiden University, Leiden, The Netherlands. His research has been supported by the National Natural Science Fund of China, the National Science and Technology Key Plan of China, and the national CSC Scholarship. His current research interests include model checking, platoon control, and distributed system control.

MARCELLO M. BONSANGUE was born in Caltanissetta, Italy, in 1966. He received the Ph.D. degree in computer science from the Free University of Amsterdam, The Netherlands, in 1996. Since 2011, he has been an Associate Professor with the Leiden Institute of Advanced Computer Science (LIACS), Leiden University, The Netherlands. He is currently a member of the Research Group Cluster foundations of Software Technology, LIACS, and the Research Group Formal methods, Centre for Mathematics and Computer Science (CWI), Amsterdam, The Netherlands. He is also the Director of Education of the LIACS, a Coordinator for the international students and relationships, a member of the LIACS Management Team, and a member of the Scientific Council of LIACS. His current research interests include foundations of computer science and formal methods.



ZHONGWEI XU was born in Jiangsu, China, in June 1964. He received the Ph.D. degree in software engineering from Tongji University (TU), Shanghai, China, in 2005, where he has been a Professor, since 2009. He is currently the Chief of the Interlock Station of Inspection and the Test Center of the China Academy of Railway Sciences. This test center can provide unbiased testing and reporting for approved software products in the China Railway. His research has been supported by the National Natural Science Fund of China, the National Key Technology Research and Development Plan, the National High-Tech Research Development Plan, and the Ministry of Science and Technology of China. He is also an Inventor of six patents, and the author of more than 40 technical papers. His current research interests include automated debugging, testing, and formal verification. He is currently serving on Editorial Board of the *Journal of the China Railway Society*.