<http://www.diva-portal.org>

Postprint

This is the accepted version of a paper presented at *8th International Symposium, ISoLA 2018, Limassol, Cyprus*.

Citation for the original published paper:

Meinke, K. (2018)

Quantitative Safety Analysis of a Coordinated Emergency Brake Protocol for Vehicle Platoons

In: Tiziana Margaria, Bernhard Steffen (ed.), *Leveraging Applications of Formal Methods, Verification and Validation. Distributed Systems - 8th International Symposium, ISoLA 2018* (pp. 386-404). Springer

Lecture Notes in Computer Science

https://doi.org/10.1007/978-3-030-03424-5_26

N.B. When citing this work, cite the original published paper.

Permanent link to this version:

<http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-257965>

Quantitative Safety Analysis of a Coordinated Emergency Brake Protocol for Vehicle Platoons

Carl Bergenhem¹, Karl Meinke^{2**}, Fabian Ström²

1. Qamcom Research and Technology AB,
Falkenbergsg. 3, 41285 Gothenburg, Sweden,

2. School of Electrical Engineering and Computer Science,
KTH Royal Institute of Technology, 100 44 Stockholm, Sweden

Abstract. In this paper, we present a general methodology to estimate safety related parameter values of cooperative cyber-physical system-of-systems. As a case study, we consider a vehicle platoon model equipped with a novel distributed protocol for coordinated emergency braking. The estimation methodology is based on learning-based testing; which is an approach to automated requirements testing that combines machine learning with model checking.

Our methodology takes into account vehicle dynamics, control algorithm design, inter-vehicle communication protocols and environmental factors such as message packet loss rates. Empirical measurements from road testing of vehicle-to-vehicle communication in a platoon are modeled and used in our case study. We demonstrate that the minimum global time headway for our platoon model equipped with the CEBP function scales well with respect to platoon size.

Keywords: vehicle platoon, learning-based testing, Co-CPS, safety boundaries, quantitative analysis, coordinated braking

1 Introduction

A vehicle platoon (or road train) is a collection of vehicles that coordinate and collaborate to reach goals such as traveling to a certain destination, while also improving e.g. safety, fuel economy and driver comfort. One challenge for platoon design is coordination of a platoon-wide emergency brake by means of a distributed protocol (CEBP). The overall goal is to avoid collisions within the platoon while still performing braking as efficiently (i.e. with as high deceleration) as possible. To justify the deployment of a CEBP solution it is necessary to quantitatively analyse its behaviour, especially properties that impact on safety.

In this paper, we introduce a new methodology to estimate quantitative parameters related to safety properties of cooperating cyber-physical systems (Co-CPS). Our approach is based on the method of learning-based testing (LBT) [23]. We illustrate this methodology by estimating safety related parameters of

^{**} corresponding author karlm@kth.se

a platoon model that includes a novel CEBP algorithm. This case study is in many ways generic. It therefore supports the claim that our parameter estimation methodology could be extended to a wider variety of cyber-physical system-of-systems through the use of simulators and virtualised environment modeling. This is one goal of the EU project *Safe Cooperating Cyber-Physical Systems using Wireless Communication (SafeCOP¹)*.

In a platoon, the lead vehicle can be manually driven and the followers (one or more) follow the leader automatically; using control algorithms for longitudinal and lateral motion. The target inter-vehicle headway is small enough (e.g. <1 s) that dependable communication is required for the platoon to be safe. A platoon capable vehicle has the technologies (e.g. communication) to lead or follow in a platoon. Issues concerning positioning, e.g. accuracy and reliability of GPS and security, are out of scope here.

A platooning system can be considered to be a cooperative cyber-physical system-of-systems (Co-CPS). This is because vehicle-to-vehicle (V2V) communication is an enabler for the technology [33]. Failures in a platoon (e.g. poor V2V communication) could potentially cause physical harm. Safety analysis for Co-CPS introduces many technical challenges. Basic problems include the system size, and the existence of black-box third-party components, which can make it technically infeasible to perform a full static analysis (see e.g. the conclusions on platooning of [17]).

For this reason, learning-based testing (LBT) [21] is an interesting contribution to safety studies of Co-CPS. LBT combines promising aspects of both testing, simulation and model based analysis. By inferring black-box abstractions of a complex system, as well as using parallel simulation to accelerate learning, we can obtain approximate but accurate results with a good degree of scalability.

LBT uses machine learning to reverse engineer multi-vehicle system-of-system (SoS) models. These SoS models can then be subject to glass-box analysis techniques, such as model checking, to check violation of safety requirements. Previously in [22], we have used LBT to analyse platooning systems from the perspective of *qualitative safety properties*, such as vehicle collisions. In this paper we extend the scope of LBT to *quantitative estimation of safety related parameters*. We show how to use LBT to numerically estimate an *minimum value of an SoS parameter* such that a given system safety property is not violated. This will typically be a parameter that can be tuned to optimise a specific product for some desired performance. Thus it might be overtuned in a way that can compromise safety or is inappropriate for an environment in some (possibly rare) scenario.

A pertinent example of parameter estimation arises in our platooning case study. Here inter-vehicle distance and time gaps are typically reduced to a minimum in order to save fuel. The question arises: *what is the minimum value that could be chosen for all inter-vehicle gaps such that no crashes occur due to vehicles being too close?* This minimum value is influenced by many factors, not

¹ See www.safecop.eu.

only in the vehicle design itself, but also by environmental factors such as V2V communication packet loss.

Our approach to quantitative parameter estimation involves performing multiple LBT sessions to efficiently refine an estimate interval. This computationally intensive analysis becomes more feasible when simulators, models and the appropriate machine learning algorithms are executed on inexpensive multi-core hardware, which is increasingly available. We define a specific method for parameter estimation using LBT. We then illustrate it by applying it to study our distributed CEBP algorithm integrated in a platoon simulator. The CEBP algorithm is an exemplar of the Co-CPS paradigm of decentralised distributed control. An optimal design for a CEBP is influenced by many factors such as pre-existing platoon control algorithms, underlying physical dynamics models, inter-vehicle communication protocols and environmental features.

Although many safety hazards impacted by CEBP could be studied, in this paper we focus on the safety hazard due to *message packet loss* arising from radio interference. We estimate the *minimum global time headway* for different platoon sizes under both perfect communication and stochastic packet loss. This is the minimum time headway between all platoon vehicles that allows collision free motion. By extending the learning time of LBT, we can improve the reliability of this estimate to any given level.

The stochastic packet loss model we use is based on empirical data from V2V communication measurement during physical road tests with a platoon. This stochastic packet loss model, a communication protocol model and a CEBP implementation are then integrated with the platoon simulator described in [22] to model communication and vehicle dynamics performance. The main emphasis of our work however is on the analysis methodology itself, and not the problem of fully accurate platoon modeling. Since we use black-box learning methods, only platoon behavior, and not architecture or code structure are inferred. Thus our LBT approach can be transferred to more complex platoon models without difficulty.

1.1 Related Work

A platooning system for trucks with focus on fuel efficiency is presented in [20]. A brief survey of other vehicle platooning systems is given in [3]. Cooperative adaptive cruise control (CACC) is a similar technology to platooning, but has its focus entirely on maintaining steady-state longitudinal control. Emergency braking in a platoon is also studied in [13]. Here, a dedicated communication protocol and a novel controller (including control topology), that takes into account packet losses, is investigated. Assumptions of bounded packet losses are made to be able derive bounds of headway. In [32] different CACC strategies are evaluated regarding headway using simulation. Several different parameters associated with uncertainty are considered, including packet loss. An event-triggered control scheme and communication strategy is developed for platooning in [8].

Examples of static analysis applied to platooning problems where the collision free property is studied are [7,9]. In [17] it is shown that verifying vehicle code

does not scale well to the entire system-of-systems, and a mixed top-down and bottom up verification strategy are applied.

Some (but not all) of the problems encountered in message packet loss in Section 6 are related to compression waves within platoon simulations. Hence they are somewhat related to the well-known phenomenon of string instability. The effects of string stability and a networked control system have been studied in [26]. Here an analytical approach of string stability is presented for a CACC application; where each vehicle is controlled by its predecessor. Quantitative results are given through an approach based on an analytical method. Communication deficiencies are described in terms of a Maximum Allowable Transmission Interval and Maximum Allowable Delay, rather than as a stochastic model of packet loss. Safety is interpreted as string stability, rather than the crash condition of zero distance between vehicles.

In [31] an analytical framework is presented which links the wireless channel characteristics with the probability of crash in a two vehicle emergency-brake scenario. The maximum tolerable delay, between the beginning of the emergency braking by the preceding vehicle and the moment the following vehicle starts braking, is found. The developed CPS analysis approach is applied to demonstrate how V2V communication packet losses and communication delays impacts safe inter-vehicular distance for specified kinematic parameters of vehicles movements.

1.2 Organisation of the Paper

The rest of the paper is organised as follows. Section 2 presents measurement of V2V communication in a platoon of trucks during road tests, providing the basis of our communication model. Section 3 presents our novel CEBP algorithm. Section 4 presents a methodology for quantitative safety analysis using learning-based testing. Section 5 presents the platoon simulator used for safety analysis of our CEBP algorithm. Section 6 presents the results of our quantitative analysis of the minimum global time headway under conditions of packet loss. Finally, conclusions and future work are given.

2 Road Testing

In this section we describe details and results of a measurement campaign² within the Relcommh project [18] to establish packet loss levels in different platoon driving scenarios. These measurements of V2V communication were done using a platoon of four trucks, (c.f. Figure 1).

The motivation for this section is twofold. On the one hand, we wish to show in Section 6 how the reliability of quantitative safety analysis results for SoS is influenced by the accuracy of environmental modeling. On the other hand,

² The measurements were done while the first author was employed at RISE – The Swedish Research Institute (previously SP – Technical Research Institute of Sweden)

there is a need in the literature to increase understanding of the environment that a platoon is designed for. In the light of results of this section, we can point out some unrealistic assumptions made in the literature. Our measured results suggest that the low packet error rate used in [32] and assumption of no packet loss in [8] are overly optimistic.

In our measurement campaign, at each periodic message broadcast (10 Hz) from the leader truck, the perceived *packet error rate* (PER) at each of the following vehicles was measured. In Table 1, the PER is presented for three different scenarios. Messages were 500 bytes long and 5.9 GHz V2V devices according to ETSI standards [10] were used. Each truck had a left and right antenna from which it could send and receive. Therefore, two PERs are given: communication left-to-left and right-to-right. Differences between the two PERs can be motivated with differences in the immediate surrounding of either side of the vehicle. For example, on the left side of the motorway there is a metal safety barrier that separates the two traffic directions. This may impact PER. A motorway scenario and Tunnel scenarios were measured at 80 km/h vehicle speeds, with 20 m and 20-50 m inter-vehicle distance respectively. In the Parked scenario, the platoon was parked in a platoon formation with a 10 m gap between each truck. The PER between the LV and FV1 is denoted *PERbase*. First-order linear regression was used to calculate the projected average increase in PER for each vehicle hop (right most column in the table). This model was then incorporated into the platooning simulator. One result (11.14 %) could be anomalous as it falls outside the expected trend of increasing PER as the distance between communicating vehicles (LV to FV i) increases.

Table 1: Packet Error rates (Upper: left-left, Lower: right-right)

	LV to FV1	LV to FV2	LV to FV3	Average increase
Motorway	3.67 %	18.03 %	40.91 %	18.62 %
	2.72 %	5.93 %	22.13 %	9.70 %
Motorway tunnel	6.39 %	5.85 %	11.16 %	2.39 %
	6.82 %	6.74 %	11.47 %	2.32 %
Parked	0.57 %	5.89 %	22.13 %	10.78 %
	2.39 %	14.05 %	11.14 %	4.37 %

In all measured scenarios there were instances of consecutive packet loss (CPL). For the E4 motorway (left to left antenna) scenario the following was found: CPL1=61.53 % (single lost packet), CPL2=36 % (two lost packets in a row), CPL3=1.6 %, CPL4=0.8 %, CPL4..k = 0.87 %. The percentages indicate the distribution of a certain CPL, when there is a packet loss. The largest CPL (longest blackout, k) was eight packets in a row. This implies that the assumed bounds on packet loss in [13] are somewhat optimistic (at most three and five consecutive packets lost are investigated).

We note that the outcome of packet loss measurements depends on several factors such as the radio equipment, antennas, placement and environment. Further details of measurements in the road tests are found in [18].

Fig. 1: Communication scheme in the tests. LV denotes Lead Vehicle. FV_i denotes Following Vehicle i

3 A Coordinated Emergency Brake Protocol

In this section, a protocol for Coordinated Emergency Brake (CEBP) is presented. The goal of the protocol is to coordinate vehicles in an emergency brake scenario to ensure safety (no crashes). An emergency brake can be initiated by any vehicle in the platoon. Here it is assumed that the platoon of N vehicles is formed and no vehicles are joining or leaving. It must be ensured that the last vehicle receives the brake command and actuates first. Braking can commence at the last vehicle directly when it receives the "E-brake request" message. The braking vehicle then sends an acknowledgement (ACK) forward with an "E-brake ACK" message. Preceding vehicles can thus start to brake when the ACK from succeeding vehicles arrives. E.g. FV2 cannot brake until ACK is received from FV3 indicating that it has started to brake. This is illustrated in Figure 2. Each vehicle also maintains a "brake-anyway"-time-out timer. When the timer expires, the vehicle will brake directly and signal this, with an "E-Brake directly" message, to the other vehicles. The value of the time-out corresponds to the expected latency for a returning ACK. Message sending can be done with event-triggered directed broadcast, i.e. there is a sender and an explicit receiver, but the message may be overheard by other vehicles within the platoon. In this case, a vehicle can prepare its brakes in anticipation of the ACK from succeeding vehicle.

We assume that vehicles entering the platoon cannot be sorted according to deceleration capability. Instead, other sorting goals may have priority; such as destination or aerodynamic performance. Not having a sorting procedure at vehicle join implies that a brake strategy, i.e. the description of how vehicles will brake in the event of an emergency brake, must be found in another way. A simple way is to limit braking of the platoon according to the vehicle with least

Fig. 2: An E-brake command from the LV. The acknowledgement then propagates back to the LV – from back to front.

deceleration capability, as is done in [25]. Alternatively, an algorithm could find cliques of vehicles in the platoon that will brake together with a lowest common brake capability. In Figure 2 the actual deceleration capabilities are shown for an example platoon, e.g. -8 m/s^2 for the lead vehicle. As vehicles join the platoon, brake cliques will be formed, e.g. Clique 1 = (LV, FV1, -4 m/s^2), Clique 2 = (FV2, FV3, -5 m/s^2). The agreed deceleration of cliques increases towards the rear, implying that the last clique will brake the most. Note that this implies a voluntary reduction of deceleration capability in some vehicles. An algorithm for finding the brake strategy in the platoon is left for future work. CEBP assumes that a brake strategy has been decided and all vehicle will brake equally. The members and order of the platoon are known.

Our CEBP algorithm has been implemented and integrated into each vehicle in the platooning simulator of [22]. It has been studied using our quantitative safety analysis method described in Section 4 and the results are presented in Section 5.

3.1 Pseudo code

Pseudo code for the CEBP is presented in Algorithm 1. Vehicles are indexed by V_i where $i \equiv 0$ is the lead vehicle (first vehicle, also denoted LV) and $i \equiv 1..N-1$ are the following vehicles (also denoted FV, e.g. where FV1 implies $i = 1$). The last vehicle is V_{N-1} (also denoted e.g. FV3 for $N = 4$). The algorithm, that is described in the pseudo-code, is executed in each vehicle in the platoon. The index i is static in each vehicle, i.e. in each instance of the algorithm. This implies that each vehicle knows its identity and hence its position in the platoon. An E-brake command is assumed to come from an external system or to be manually initiated.

Some comments regarding the code in Algorithm 1 are appropriate: On line 11, directly receiving an “E-brake request” implies that V_i is the last vehicle. This is because any vehicle that requests to E-brake will do so by sending to the last vehicle. On line 25, an ACK is sent by a vehicle that did “brake directly”. This is because there could be preceding vehicles that are waiting for the ACK. If the ACK was not sent then the preceding vehicles can start to brake only after

Algorithm 1 CEBP - Loop in every vehicle

```
1: if Ego Vehicle  $V_i$  wants to e-brake then
2:   send "E-brake request" to the last vehicle in the platoon  $V_{N-1}$ 
3: end if
4: if "E-Brake directly" is received by Ego Vehicle  $V_i$  then
5:   send "E-brake request" to the last vehicle in the platoon  $V_{N-1}$ 
6: end if
7: if Ego Vehicle  $V_i$  (has sent "E-brake request" command) or (overheard "E-brake
   request" or "E-brake ACK" from  $V_j$ ) then
8:   prepare brake system
9:   Start  $\text{Timer}_i$ 
10: end if
11: if "E-brake request" is received by Ego Vehicle  $V_i$  from a preceding vehicle  $V_j$ ,
   where  $j \in \{0..i-1\}$  then
12:   Ego Vehicle  $V_i$  actuate e-brake strategy
13:   send "E-brake ACK" to the next preceding vehicle  $V_{i-1}$ 
14: end if
15: if "E-brake ACK" is received by Ego Vehicle  $V_i$  from next succeeding vehicle  $V_{i+1}$ 
   then
16:   Ego Vehicle  $V_i$  actuate e-brake strategy
17:   Stop  $\text{Timer}_i$ 
18:   if  $i > 0$  and has not already sent an "E-brake ACK" to preceding then
19:     send "E-brake ACK" to the next preceding vehicle  $V_{i-1}$ 
20:   end if
21: end if
22: if  $\text{Timer}_i$  has expired then
23:   Ego Vehicle  $V_i$  actuate e-brake strategy
24:   send "E-Brake directly" to succeeding vehicles  $V_j$ , where  $j \in \{i+1..N-1\}$ 
25:   send "E-brake ACK" to the next preceding vehicle  $V_{i-1}$ 
26: end if
27: if  $\text{Timer}_i$  is started then
28:   decrease  $\text{Timer}_i$ 
29: end if
```

their time-out counters expire. On line 24 and 25 the messages are repeated e.g. until the algorithm is reset. On line 5 an alternative is possible. Instead of EBR/ACK, a vehicle that receives “E-brake directly” could also do “E-brake directly”.

4 An LBT Methodology for Quantitative Safety Analysis

In this section, we review some fundamental principles of learning-based testing (LBT). We then show how these methods can support a quantitative approach to safety analysis

4.1 Learning-based Testing (LBT)

We begin by reviewing the fundamental principles of learning-based testing (LBT) as these have been implemented in our research tool LBTest. The earliest version of this tool (LBTest 1.x) has been described in [24]. The current tool architecture of LBTest 3.x is presented in Figure 3. This is a concurrent software architecture designed to support LBT on multi-core hardware. Such hardware supports the parallel execution of machine learning queries in multiple threads, where each thread executes a copy SUT_i of the *system under test* (SUT) (c.f. Figure 3). This approach reduces both the simulation time and the learning time, as the learning algorithm itself can also be parallelized. Examples of computation time improvements by such parallelisation have been shown in [22]. By increasing the throughput of data, a larger data set becomes available for machine learning. This increases the accuracy or convergence of the final learned model and hence the reliability of quantitative parameter estimates. For analysing complex Co-CPS behaviors, we believe that concurrency is essential. Since the design of the architecture in Figure 3 has been discussed in [22], we focus on the basic principles of LBT here.

LBTest uses active automaton learning aka. *regular inference* (see e.g. [14]) to generate queries about a black-box SUT. These queries are then executed on the SUT as test cases, and the SUT behaviour is observed for each test case. In an iterative and incremental process, the test cases and the SUT observations are saved and used to build up a behavioral model of the SUT in polynomial time [1]. This model is an automaton or state machine model.

For requirements testing, partial and incomplete models of the SUT can already be subjected, in the early stages of testing, to model checking against a temporal logic requirement specification. Thus, even before the learning process is complete, errors can be found in the SUT. This fact is important for large and complex SUTs such as Co-CPS, where it might not be possible to learn a complete model in any reasonable timescale, even with the use of multi-core technology. In LBTest, propositional linear temporal logic³ (PLTL) is used as

³ Recall that propositional LTL extends basic propositional logic with the temporal modalities $G(\phi)$ (always ϕ), $F(\phi)$ (sometime ϕ) and $X(\phi)$ (next ϕ). Other derived operators and past operators may also be included. See e.g. [12] for details.

Fig. 3: LBTest 3.x concurrent learning architecture

the requirements modeling language. This particular logic has the advantage that test cases can easily be extracted from the model checker, and used to filter out false negatives as we will show. LBTest makes use of a loosely integrated symbolic checker NuSMV [6]. We are also developing a more tightly integrated explicit state model checker for efficiency reasons. These two processes of learning and model checking may be interleaved, an idea first suggested in [27]. Then they incrementally build up a sequence $M_1, M_2, \dots$ of models of the SUT, while generating and executing requirements test cases on each model M_i . However, for large and complex Co-CPS this interleaved approach is too inefficient, and model checking is then only performed on the final model. In Section 6 we have used model checking on the final model only. Thus no *bias* to the model from model checking and counterexample construction can exist.

To separate true negatives (genuine SUT errors) from false negatives (artifacts of an incompletely learned model) it is necessary to validate each counterexample to a requirement generated by the model checker. For this we can: (i) extract a test case representing the counter-example⁴, (ii) execute it on the SUT, (iii) apply an equality test that compares the observed SUT behavior with the predicted bad behavior from the model, and (iv) automatically generate the test verdict (**pass**, **fail**) from step (iii).

The *soundness* of learning-based testing as an analysis method relies on the soundness of the underlying model checker, and the soundness of equality testing.

⁴ Infinite counter-examples to LTL liveness formulas are truncated around the loop, and the weaker test verdict **warning** may be issued.

The *completeness* of LBT as an analysis method relies on the completeness of the underlying model checker, as well as convergence results about the learning algorithms which are used (see [14]). However, within practical case studies of large complex systems it may not be possible for learning to be completed in any reasonable time frame (see e.g. [11]). This problem is significant for Co-CPS. Therefore, development of LBTest has focused on incremental learning algorithms that can generate incomplete approximating models of the SUT in small increments.

To measure the *test coverage* achieved by learning-based testing we currently use a *probably exactly correct* (PEC) model of learning convergence as follows. In Figure 3, a stochastic equivalence checker is shown. This checker empirically estimates the behavioral accuracy of the final learned model M_{final} for replicating the behavior of the SUT on a randomly chosen set of input sequences. For this, the input sequences are executed both on the SUT and the model. We then measure the percentage of behaviorally identical output sequences generated by both. This learning convergence model is more restrictive than the *probably approximately correct* (PAC) convergence model of [30]. There are two motivations for this: (i) our automaton learning framework does not readily support notions of approximate equivalence between data values, and (ii) for software safety analysis exact equality of data values (inputs or outputs) is often a pre-requisite to infer failed test cases.

4.2 Quantitative Parameter Estimation

A qualitative safety analysis of platooning using LBT was given in [22]. Here we extend this previous approach to quantitative parameter estimation. We are interested to estimate the minimum values of numerical system parameters (such as inter-vehicle distance and time headway) which lie on the boundary between safe and unsafe system behavior.

More precisely, in quantitative parameter estimation, the problem is to estimate the minimum value v_{min} of some continuous SUT parameter p such that an LTL safety property $prop$ is not violated. The parameter p could be an input variable, or a system constant that must be set to an optimal value. Now p may or may not explicitly appear in the formula $prop$ but it should be able to influence its truth value (see e.g. the formula Eq 1 in Section 6).

If we can assume that the safety property $prop$ varies monotonically with p , then this allows us to use a binary chop search to iteratively halve an estimate interval $v_{min} \in [v_{true}^i, v_{false}^i]$ for $i = 0, \dots, n$. Here, v_{true}^i is the current upper bound where $prop$ is true and v_{false}^i is the current lower bound where $prop$ is false. The search begins from two initial endpoints $[v_{true}^0, v_{false}^0]$ that can be obtained by conservatively over-estimating and under-estimating the value of v_{min} .

For a binary chop search, as usual we iterate the boundary search process by refining one of the endpoints. Thus: (i) $v_{true}^{i+1} := v_{true}^i + v_{false}^i/2$ if LBT cannot find a counterexample to $prop$ on the midpoint up to a given learning convergence

value. Otherwise: (ii) $v_{false}^{i+1} := v_{true}^i + v_{false}^i/2$. Then we carry forward into the next iteration the other endpoint $v_{false}^{i+1} := v_{false}^i$ in case (i) and $v_{true}^{i+1} := v_{true}^i$ in case (ii) respectively. This process is iterated until a desired interval accuracy $[v_{true}^n, v_{false}^n]$ is achieved.

Refinement of the boundary v_{true}^i is of course problematic here, since just because a counterexample has not been found by LBT, this does not mean that it does not exist. This is particularly true if the learned models are incomplete. Therefore, we emphasize that our methodology is a parameter estimation technique based on systematic testing, and not a verification technique. As such, our methodology provides an alternative to a traditional Monte-Carlo estimation of v_{min} . However, we believe there are three significant advantages to our approach compared with Monte-Carlo techniques, based on the use of machine learning.

(1) The explicit construction of a model using machine learning gives a more powerful artifact than simply a set of execution traces (as used in Monte Carlo estimation). This model allows us to analyze complex requirements properties, including safety, fairness and liveness issues. These properties cannot be semantically evaluated on traces alone, i.e. they are global properties of a model.

(2) Convergence estimates for the model give more insight into reliability of the estimate for v_{min} than simply measuring the size and statistical significance of a randomly chosen Monte Carlo sample set. This fact is easily demonstrated, for if complete learning succeeds then a Monte Carlo approach is never aware of this and will underestimate the statistical significance of the result. A related aspect to this is the third advantage.

(3) The random query set associated with a Monte Carlo estimate contains significant redundancy when compared with a query set generated by active automaton learning. Said differently, random querying is a very inefficient way to learn the structure of an automaton.

5 A Platooning Simulator

The simulator implements a model for each platoon vehicle behaviour as well as a communication framework for inter-vehicle (V2V) communication modelled on the IEEE 802.11p protocol. The platooning simulator is capable of simulating an N -vehicle platoon travelling in one dimension along a roadway. It is an extension of the simulator presented in [22]. No steering model (i.e. lateral movement) is currently present in the simulator. This extension is part of ongoing research into more general spatio-temporal logic requirements modeling for Co-CPS, see e.g. [19].

5.1 The Vehicle Model

A key control algorithm in the platooning simulator is the longitudinal position controller. For this, we have implemented several published ACC algorithms which control the CACC component of each vehicle (see [29] for detailed descriptions of each). The specific ACC evaluated in Section 6 is Kakade's algorithm

[16], which was chosen for its simplicity and a basic tendency to propagate compression waves. We were interested to know whether this effect, in combination with message packet loss, could disturb emergency braking, and whether LBTest could discover such a problem.

In the simulator there is a detailed model of vehicle braking. This includes a complete industrial model of a brake-by-wire subsystem featuring: (i) global brake torque distribution to individual wheels, (ii) ABS functionality based on slippage detection, and (iii) a friction model for tyres based on slippage rate using common physical parameter values. The simulator also includes e.g. odometry and V2V communication. The most relevant missing models are engine, powertrain and suspension models. While these models could easily be added by using an industrial simulator such as TruckMaker [15] (which is ongoing research) they would not invalidate the basic methodology of this paper.

Environment models in the simulator deal with air resistance and road friction. We assume a constant road friction value for simplicity. A message packet loss model, based on the data of Section 2 was used. To provide deterministic and repeatable behavior (with the exception of packet loss), the simulator is based on synchronous execution of all vehicle components. The fundamental simulation cycle is one millisecond, which provides adequate simulation accuracy for the control algorithms.

5.2 The Communication Model

The communication framework assumes wireless broadcast and point-to-point multi-hop communication between the vehicles in the platoon. A slotted TDMA scheme based on ideas from [5] is implemented: To avoid communication collisions, each vehicle V_i is allowed to transmit only in its own TDMA slot.

As communication is broadcast-based, receiving vehicles can lose packets independently during a broadcast operation. Thus a packet can be received by one vehicle and lost by another. For example a broadcast from the LV is correctly received at FV1 and FV2, but not FV3, see Figure 1. In a platoon of N vehicles, for any sender V_i and receiver V_j (where $0 \leq i, j \leq N - 1$, $i \neq j$) let $d = |i - j|$ correspond to the distance between the sender and receiver. The probability P in percent of a message being lost is $P(\text{message lost}) = PER_{base+increase} \cdot (d - 1)$. Note that with the values from the road test, the probability of message loss (from the LV to the last vehicle) is 100% in a platoon of eight vehicles or more; hence every message is lost (unless e.g. multi-hop communication is used).

6 A Case Study in Quantitative Safety Analysis

In this section, we present a case study of applying our quantitative parameter estimation method. The aim was to estimate the minimum safe *global time headway* for a platoon which has two modes of behavior: high speed cruising and emergency braking.

The *local time headway* $hw_i(t)$ between two consecutive platoon vehicles V_i and V_{i+1} at time t is the time which would be needed for V_{i+1} to cross the gap which exists between V_i and V_{i+1} at time t ⁵. This local dynamic parameter measures the inter-vehicle gap in terms of time rather than distance. As a runtime parameter to the CACC of V_{i+1} , its driver can set a desired value HW_i for $hw_i(t)$, according to relevant safety and fuel economy criteria. Typical values for HW_i are in the range 1.5 to 2.0 seconds [4]. This desired value HW_i is then maintained by the CACC. Perturbations to $hw_i(t)$ through lead vehicle V_0 actions, will lead to short term deviations of $hw_i(t)$ from HW_i , which should be smoothed out by its CACC.

We are particularly interested to estimate system-of-system parameters. For this purpose, we assume that each platoon vehicle V_i adopts the same common *global time headway* HW , so that $HW = HW_i$. Now we can ask: *what is the smallest value HW_{min} we can choose for HW which ensures safe driving for all vehicles V_i under all possible modes of behavior*⁶? By safe driving, we can assume as a minimum condition crash-free driving, but obviously this criterion could be strengthened. The value HW_{min} we term the *minimum safe global time headway*. An estimate of HW_{min} is easily obtained by LBT if communication between vehicles is perfect, as the SUT is then completely deterministic.

When communication is imperfect then message packet loss is modeled stochastically and the SUT is no longer deterministic. Although most model checkers (including NuSMV) cope well with non-determinism, currently, LBTest uses ML algorithms for deterministic automata only. To address this learning problem we inferred a set of deterministic models which support analysis of the average case behavior of the SUT. This seems pertinent, as the worst case SUT behavior involves catastrophic loss of all message packets. An alternative for future research would be to directly apply ML algorithms for non-deterministic or even probabilistic automata. (See Section 7.)

The integration of two control algorithms for high-speed cruising and emergency braking requires corresponding integration testing to ensure that no unwanted interactions can occur between these algorithms. In principle, high-speed cruising can bring the entire platoon to a state where emergency braking cannot be carried out safely. Such problems (if they occur) might be addressed by choosing a larger global time headway, so that unsafe states were no longer reachable. Thus one way to structure integration testing is to view it as an estimation problem for HW_{min} such that platooning is safe for both cruising and emergency braking with high probability.

To conduct parameter estimation for HW_{min} , the following protocol was implemented in LBTest. As in [22], we focused on emulating the lead driver behavior, since all follower vehicles autonomously adapt to this. Each test case

⁵ Assuming V_{i+1} maintains its speed at time t .

⁶ Clearly HW_{min} is a function of the many individual parameters of each vehicle V_i such as its weight, braking power etc. Different values of HW_{min} will thus be obtained if individual vehicle parameters are changed. For simplicity, we have assumed a *homogeneous platoon*, i.e. all vehicle parameters are the same.

tc for an N -vehicle platoon consisted of a sequence $tc = (r_1, r_2, \dots, r_\lambda)$ of lead driver accelerator, brake or emergency brake commands r_j . Each such command was one of: (i) a *brake command* (-1.88 m/s^2), (ii) an *accelerate command* (1.25 m/s^2), (iii) a *neutral command* (0 m/s^2), or (iii) an *emergency brake command* (-2.22 m/s^2). The initial estimate of HW_{min} was bounded between 0.5 and 2.0 seconds.

For each test case $tc = (r_1, r_2, \dots, r_\lambda)$, the length λ and torque requests r_j were chosen dynamically both by the learning algorithm and the equivalence checker. For efficiency reasons, model checking was not used until after learning was concluded. Thus model checking counterexamples did not influence the analysis. The test case length λ took an average value of 18.3. On average, random test cases amounted to 2.3% of the entire test set. This compares with 100% in the case of Monte Carlo parameter estimation. Thus 97% of test cases were generated deterministically by ML to explore the state space of the SUT. The communication wrapper loaded and executed each test case tc . Each torque request value r_j was maintained constantly for a nominal 5 seconds (5000 simulation cycles). Thus the length of the simulation corresponding to tc was 5λ virtual seconds. The values chosen for λ were sufficient to reach high cruising speeds, in excess of 120 km/h .

The principle SUT output recorded for the test case tc was the time sequence of inter-vehicle gaps $x_{r,0}^i, \dots, x_{r,\lambda}^i$, for each pair of vehicles V_i, V_{i+1} . Here, the time sequence term $x_{r,t}^i$, for $0 \leq t \leq \lambda$, represents the gap between the host-target pair, V_i and V_{i+1} measured at the end⁷ of $5t$ virtual seconds (i.e. $5000t$ simulation cycles). The continuous values of each distance observation $x_{r,t}^i$ were partitioned within the communication wrapper into three discrete equivalence classes:

good, tooClose, crash,

based on host and velocity dependent distance boundaries.

To represent the physical system state of the platoon we also observed the lead vehicle velocity values $v_0^1, \dots, v_\lambda^1$ and acceleration values $a_0^1, \dots, a_\lambda^1$ at the same observation times. These continuous valued observations were partitioned into 1 km/h and 1 km/h^2 equivalence classes.

During test sessions, each test case constructed by LBTest brought the entire platoon into a high speed cruising mode (using a sequence of non-random or random acceleration and braking commands). The test case would then issue the emergency brake command e followed by a sequence of neutral commands 0 ⁸. By alternating brake and acceleration commands, each test case could establish different global dynamics in the platoon at the moment of emergency braking. For example, by choosing to evaluate the simple PID algorithm for CACC of [16], we were able to observe *compression waves* where some vehicles were decelerating while others were accelerating. When the choice of global time headway HW fell

⁷ It is also possible to use SUT observations between the output cycles by thresholding. This can yield greater accuracy, but this approach was not taken here.

⁸ These terminating neutral commands 0 were redundant by the design of CEBP, but extended the test case until the platoon was stopped.

below the minimum safe global headway HW_{min} then at least one failed test case could be observed. Since some of these failed test cases exhibited compression waves, we concluded that compression is an important non-linear dynamic for certain CACC designs. This observation concurs with the extensive literature regarding *string stability* and ACC design, e.g. [28].

The safety requirement for collision free travel was expressed in LTL as

$$\text{always}(\bigwedge_{i=0}^{N-1} \text{Gap}_i > 0). \quad (\text{Eq 1})$$

This formula expresses that a platoon of size N is safe, since Gap_i represents the i -th inter-vehicle time headway between vehicles V_i and V_{i+1} . Notice that the time headway t is not explicitly represented in this formula. Nevertheless, t clearly influences Requirement Eq 1 as too short a headway leads to crashes. Furthermore, t monotonically influences Eq 1, since every platoon trajectory with a minimum time headway t is also a legitimate trajectory for a minimum time headway of $t' \geq t$. So parameter estimation using a bisection method is valid for this problem.

Fig. 4: Minimum safe global time headway HW_{min} for different platoon sizes N and two packet loss rates.

The minimum safe global time headway HW_{min} was estimated for two different wireless communication scenarios. In the first, perfect data transmission

was assumed in order to derive a baseline time headway value. In the second, the packet loss model (c.f. Section 5) with parameters derived from the measurements of packet loss described in Section 2 was used. *PERbase* and average increase per vehicle hop was chosen from the motorway scenario: 3.67 % and 18.6 % respectively. These values were the basis for a linear regression model to calculate the probability of a packet being lost.

The minimum safe global time headway HW_{min} for these two scenarios was estimated for platoon sizes $N = 2, \dots, 6$ to study its variation with platoon size. The results can be seen in Figure 4. Significant is the observation that in both scenarios HW_{min} reaches a maximum value. This can be interpreted to mean that both the CACC and CEBP algorithms are scalable to large platoon sizes.

7 Conclusions

In this paper we have addressed a challenge in the area of cooperative cyber-physical systems (Co-CPS) which is to quantitatively estimate safety related parameters for a system-of-systems. An inherent problem here is the significant system complexity which calls for novel analysis techniques that can even deal with the case where components may be "black box", i.e. their design and construction are not always known. Thus a black-box approach to parameter estimation based on learning-based testing (LBT) has been applied, and implemented using the tool LBtest.

To illustrate and evaluate our approach we have presented a case study in the area of vehicle platooning. This case study consisted of a platooning simulator integrated with a CEBP - a distributed protocol for coordinated emergency braking. The minimum safe global time headway for this platooning simulator was found for different platoon sizes, both with and without lossy communication.

Future research could expand this case study, for example by considering the effects of time variant communication quality, and compare schemes, such as multi-hop communication, to improve packet reception. This would increase probability of reception, but latency will scale with the number of hops. We could also study the behavior of non-homogeneous platoons.

Future research could also improve the efficiency and accuracy of the LBT algorithms used here in the case of non-deterministic SUT behavior. For such behavior, it is possible to directly implement machine learning algorithms for non-deterministic and probabilistic automata (see e.g. the survey [2]). This would avoid the need to estimate parameter values using several experiments. Furthermore, by learning probabilistic automaton models it may even be possible to estimate the statistical distribution of a parameter value by means of statistical model checkers such as PRISM [34]. Finally, our LBT approach could be empirically compared with Monte Carlo based approaches, regarding accuracy and reliability of parameter estimates.

Acknowledgement

The research leading to these results has been performed in the SafeCOP project, that received funding from the ECSEL Joint Undertaking under grant agreement 692529, and from Vinnova Swedish national funding. The work was partially performed in the Next Generation Electrical Architecture (NGEA) step2 project, funded by the Vinnova FFI-programme. We express special thanks for valuable comments to Magnus Jonsson and Alexey Vinel of Halmstad University.

References

1. Angluin, D.: Learning regular sets from queries and counterexamples. *Inf. Comput.* 75(2), 87–106 (Nov 1987)
2. Bennaceur, A., Meinke, K.: Machine learning for software analysis: Models, methods, and applications. In: *Machine Learning for Dynamic Software Analysis: Potentials and Limits. Lecture Notes in Computer Science*, vol. 11026, pp. 3–49. Springer (2018)
3. Bergenhem, C., Shladover, S., Coelingh, E., Englund, C., Shladover, S., Tsugawa, S.: Overview of platooning systems. In: *Proc. 19th ITS World Congress*, Vienna, Austria (October 2012)
4. van den Bleek, R.: Design of a Hybrid Adaptive Cruise Control Stop-&-Go system. Master’s thesis, Technische Universiteit Eindhoven, Department of Mechanical Engineering (2007)
5. Bohm, A., Jonsson, M., Kunert, K., Vinel, A.: Context-Aware Retransmission Scheme for Increased Reliability in Platooning Applications, pp. 30–42. Springer International Publishing, Cham (2014), http://dx.doi.org/10.1007/978-3-319-06644-8_4
6. Cimatti, A., Clarke, E., Giunchiglia, E., Giunchiglia, F., Pistore, M., Roveri, M., Sebastiani, R., Tacchella, A.: NuSMV 2: An OpenSource Tool for Symbolic Model Checking, pp. 359–364. Springer (2002)
7. Colin, S., Lanoix, A., Kouchnarenko, O., Souquieres, J.: Using CSPIIb Components: Application to a Platoon of Vehicles, pp. 103–118. Springer (2009)
8. Dolk, V.S., Ploeg, J., Heemels, M.: Event-triggered control for string-stable vehicle platooning. *IEEE Transactions on Intelligent Transportation Systems* 18(12), 3486–3500 (Dec 2017)
9. El-Zaher, M., Contet, J., Gruer, P., Gechter, F., Koukam, A.: Compositional verification for reactive multi-agent systems applied to platoon non collision verification. *Stud. Inform. Univ.* 10(3), 119–141 (2012)
10. European Telecommunications Standards Institute: Intelligent Transport Systems (ITS); Access layer specification for Intelligent Transport Systems operating in the 5 GHz frequency band. EN 302 663 V1.2.1, ETSI (July 2013)
11. Feng, L., Lundmark, S., Meinke, K., Niu, F., Sindhu, M.A., Wong, P.Y.H.: Case Studies in Learning-Based Testing, pp. 164–179. Springer (2013)
12. Fisher, M.: *An Introduction to Practical Formal Methods Using Temporal Logic*. Wiley Publishing (2011)
13. Giordano, G., Segata, M., Blanchini, F., Cigno, R.L.: A joint network/control design for cooperative automatic driving. In: *2017 IEEE Vehicular Networking Conference (VNC)*. pp. 167–174 (Nov 2017)

14. De la Higuera, C.: Grammatical inference: learning automata and grammars. Cambridge University Press (2010)
15. IPG Automotive: Brochure about CarMaker, TruckMaker and MotorcycleMaker. <https://ipg-automotive.com/pressmedia/media-library/> (20018), [Online; accessed 11-June-2018]
16. Kakade, R.S.: Automatic Cruise Control System. Master's thesis, Indian Institute of Technology, Department of Systems and Control Engineering, Mumbai (2007)
17. Kamali, M., Dennis, L.A., McAree, O., Fisher, M., Veres, S.M.: Formal verification of autonomous vehicle platooning. *Science of Computer Programming* 148, 88–106 (2017)
18. Karlsson, K., Carlsson, J., Larsson, M., Bergenhem, C.: Evaluation of the v2v channel and diversity potential for platooning trucks. In: *Antennas and Propagation (EuCAP) Proceedings of the 10th European Conference, Davos, Switzerland, 11-15 April, 2016*. (2016)
19. Khosrowjerdi, H., Meinke, K.: Learning-based testing for autonomous systems using spatial and temporal requirements. In: *Proc. 1st International Workshop on Machine Learning and Software Engineering in Symbiosis*. IEEE (2018)
20. Liang, K.Y., Mårtensson, J., Johansson, K.H.: Heavy-duty vehicle platoon formation for fuel efficiency. *IEEE Transactions on Intelligent Transportation Systems* 17(4), 1051–1061 (April 2016)
21. Meinke, K., Sindhu, M.A.: Incremental learning-based testing for reactive systems. In: Gogolla, M., Wolff, B. (eds.) *Tests and Proofs: 5th International Conference, TAP 2011, Proceedings*. pp. 134–151. Springer (2011)
22. Meinke, K.: Learning-based testing of cyber-physical systems-of-systems: A platooning study. In: *Computer Performance Engineering - 14th European Workshop, EPEW 2017, Berlin, Germany, September 7-8, 2017, Proceedings*. pp. 135–151 (2017)
23. Meinke, K., Sindhu, M.A.: Incremental learning-based testing for reactive systems. In: *Tests and Proofs - 5th International Conference, TAP 2011, Zurich, Switzerland, June 30 - July 1, 2011. Proceedings*. pp. 134–151 (2011)
24. Meinke, K., Sindhu, M.A.: Lbtest: A learning-based testing tool for reactive systems. In: *Proceedings of the 2013 IEEE Sixth International Conference on Software Testing, Verification and Validation*. pp. 447–454. ICST '13, IEEE Computer Society (2013)
25. Murthy, D.K., Masrur, A.: Braking in close following platoons: The law of the weakest. In: *2016 Euromicro Conference on Digital System Design (DSD)*. pp. 613–620 (Aug 2016)
26. Oncu, S., Van de Wouw, N., Heemels, M., Nijmeijer, H.: String stability of interconnected vehicles under communication constraints. In: *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*. pp. 2459–2464. IEEE (2012)
27. Peled, D.A., Vardi, M.Y., Yannakakis, M.: Black box checking. In: *Formal Methods for Protocol Engineering and Distributed Systems, FORTE XII / PSTV XIX'99, IFIP TC6 WG6.1*. pp. 225–240 (1999)
28. Swaroop, D., Hedrick, J.: String stability of interconnected systems. *IEEE Transactions on Automatic Control* 41, 349–357 (1996)
29. Trochez, D., Tsakalos, A.: Adaptive Cruise Control Implementation with Constant Range and Constant Time-Gap Policies. Master's thesis, KTH Royal Institute of Technology, EECS School (2017)
30. Valiant, L.G.: A theory of the learnable. *Commun. ACM* 27(11), 1134–1142 (Nov 1984)

31. Vinel, A., Lyamin, N., Isachenkov, P.: Modeling of v2v communications for c-its safety applications: a cps perspective. *IEEE Communications Letters* (2018)
32. van Willigen, W.H., Schut, M.C., Kester, L.J.H.M.: Evaluating adaptive cruise control strategies in worst-case scenarios. In: 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC). pp. 1910–1915 (Oct 2011)
33. Willke, T.L., Tientrakool, P., Maxemchuk, N.F.: A survey of inter-vehicle communication protocols and their applications. *Commun. Surveys Tuts.* 11(2), 3–20 (apr 2009), <http://dx.doi.org/10.1109/SURV.2009.090202>
34. Younes, H.L.S., Kwiatkowska, M.Z., Norman, G., Parker, D.: Numerical vs. statistical probabilistic model checking. *STTT* 8(3), 216–228 (2006)