EPiC Series in Computer Science

Volume 34, 2015, Pages 196–206

ARCH14-15. 1st and 2nd International Workshop on Applied veRification for Continuous and Hybrid Systems

Running SpaceEx on the ARCH14 Benchmarks

Stefano Minopoli[1] and Goran Frehse[1]

VERIMAG, Centre Équation - 2, avenue de Vignate, 38610 GIÈRES
{stefano.minopoli,goran.frehse}@imag.fr

Abstract

In this paper, we present experimental results from running SpaceEx on some of the benchmarks of the ARCH14 workshop. Some of the SpaceEx models were obtained from Matlab/Simulink models with the help of a new translation tool. While some benchmarks could be handled to our satisfaction, several still pose significant challenges. We discuss possible alternatives for handling the problematic cases.

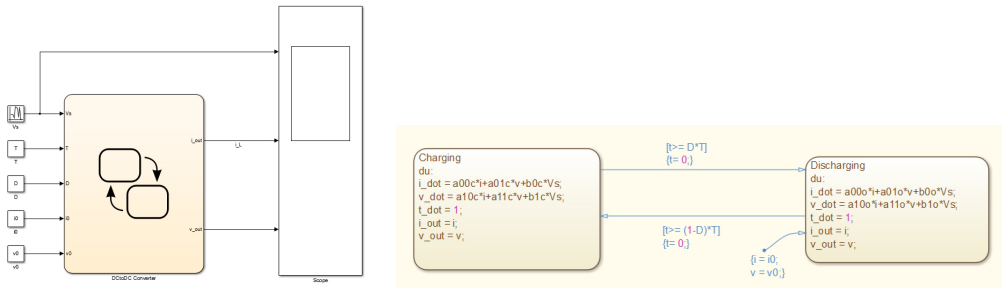
1 Introduction

This paper presents some results from running SpaceEx [5] on some benchmarks of the 2014 Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH14) [6]. For hybrid systems already modeled by Simulink, we used the experimental tool *Simulink to SpaceEx (SL2SX)* to help the process of building the SpaceEx model (publication under submission). The following benchmarks were selected from the ARCH14 pool of benchmarks. The selection criteria were that SpaceEx currently requires piecewise affine dynamics and can handle only reachability problems. The models and configuration files for running the examples in SpaceEx are provided in the attachment.

2 DC-to-DC Switched-Mode Power Converters

This example is related to a model of the *three DC-to-DC switched-mode power converters* described in [10], with continuous dynamics specified by linear ordinary differential equations. A DC-to-DC converter transforms a DC source voltage from one voltage level to another utilizing switches toggled at some (typically kilohertz) frequency with some duty cycle. In particular here we are considering the *buck-boost converter* which, based on the duty cycle of the transistor's switching, can either increase or decrease the source voltage. The system continuous variables consist in currents and voltages. According to the authors, this example is classified as academic, with medium difficulty.

The Simulink model for this system is depicted in Figure 1(a), while Figure 1(b) shows the Stateflow block of the system. We use the tool *SL2SX* to translate the Simulink model into a SpaceEx model. The translation is semi automatic: the tool carries out a network, predisposing all the components needed to model the system, and then some of them need to be refined by the user. In particular, for the main system, the user must delete the network component



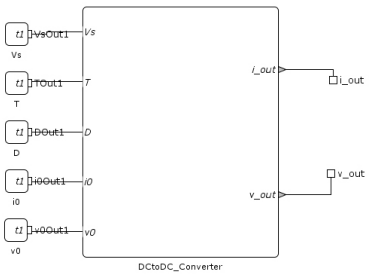
(a) Main system (b) Stateflow block

Figure 1: Simulink diagram for DC-to-DC switched-mode power converters

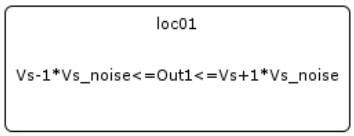
corresponding to the scope block (it is useless for the aims of simulation and verification) and must replace the variable used as parameters in the blocks, with their actual values. The second refinement is related to block types not supported by the current tool version, namely random number and stateflow. The first one provides as output a normally (Gaussian) distributed random signal, by specifying mean and the variance of the signal. Here, this block is used to provide a fixed input voltage V_s (i.e. the mean of the block) whose noise is described by the variance $Vs\_noise = 0.1$. The hybrid automaton shown by Figure 2(b) is our abstraction this block. It consists of a single location, whose invariant guarantees that the values of the output variable is always inside the interval $[Vs - Vs\_noise\, Vs + Vs\_noise]$. Figure 2(c) instead shows a possible hybrid automaton for modeling the stateflow block, whose designs are very similar. Indeed, the automaton consists of the same locations (i.e. *Charging* and *Discharging*), flows and transitions. The only difference concerns the addition of invariants, with the aim of constraining system to fire transitions as-soon-as possible their guards are enabled (according to SLSF must semantics). Figure 2(a) shows the network component corresponding to the SL main system, while Figure 2(c) depicts the hybrid automaton that models the Stateflow block.

Figure 3(a) shows the result of the Simulink simulation, for the voltage during the first 0.004 sec, with input voltage $Vs = 17.5$, current equal to 6 and desired voltage $V_{ref} = 15$. Then, the SpaceEx verification engine is used to compute the reachability set for the voltage. The initial state is set according to the value of the initial and desired voltage, the current and also to stateflow initial conditions (location *Discharging*). The reachable set is computed by the STC algorithm, parametrized with 0.01 as flowpipe tolerance, 4×10^{-6} sec as local time horizon (it needs to be bigger than the time spent inside any location), and 64 octagonal template directions. We obtain essentially the same reachable set by changing the template directions (box instead of octagonal) or by reducing the flowpipe tolerance to 0.001 (which increases the computation time). Figure 3(b) shows the reachability set computed (after 926.0 sec) for the maximum iterations number of 2000. The comparison between this set with the Simulink simulation in Fig. 3(a) shows how the error accumulation affects the reachability analysis.

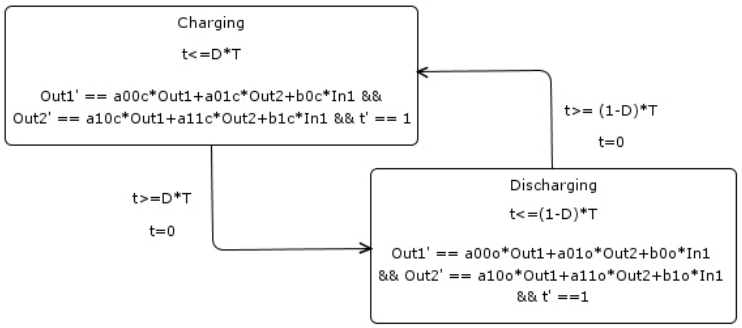
We also investigated the safety properties called *start-up regulation* and *steady-state regulation*. The properties are checked using an additional component, called monitor, which is able to catch when the property under analysis is violated. The system, together with the component for monitoring, is depicted in Figure 4(a).



(a) main system

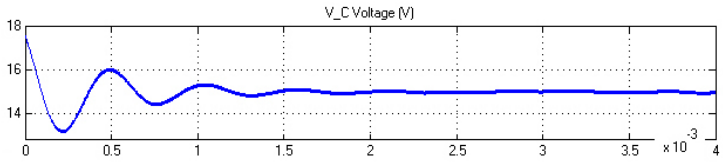


(b) replacement of the random number block

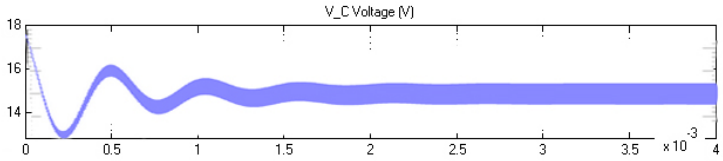


(c) HA model of the stateflow block.

Figure 2: SX model from *SL2SX* after completion by user



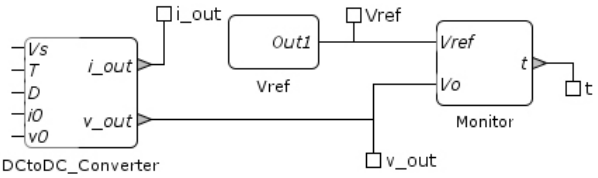
(a) Simulink Simulation over 0.004 sec.



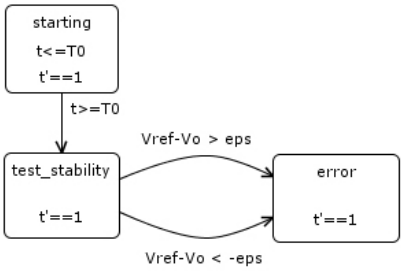
(b) reachability set over 0.004 sec.

Figure 3: Simulation and reachable set for the voltage over the first 0.004 s.

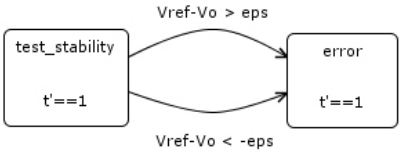
The *start-up regulation* property consists of checking, from the initial condition with output voltage and current equal to zero, whether after some time T_0 the output voltage is always near the desired value V_{ref}. Figure 4(b) shows a hybrid automaton for monitoring this property. It includes a clock t used to measure when T_0 is reached. The monitor then checks whether



(a) Component for monitoring start-up and steady-state safety property



(b) HA model for start-up monitoring



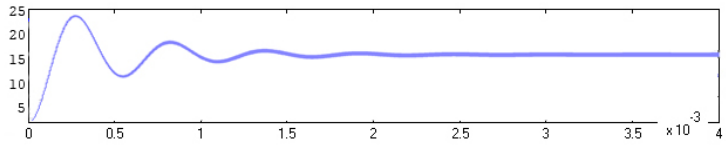
(c) HA model for steady-state monitoring

Figure 4: Model for checking startup and steady-state regulation

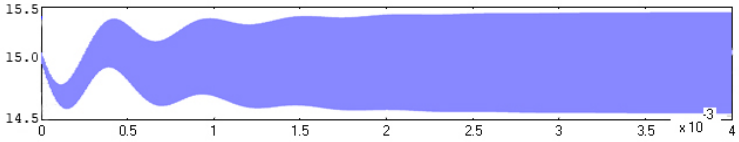
the condition $|V_{ref} - Vo| \leq \epsilon$ holds. If this happens, meaning that the start-up property is violated, the outgoing transitions of location *test_stability* can fire, and the system reaches the location *error*. As an experiment, the property was checked using the SpaceEx engine, for $T_0 = 0.003$ sec, $\epsilon = 0.5$, with initial state as above, and by declaring location *error* as forbidden state, The STC algorithm was used with 0.01 as flowpipe tolerance, 4×10^{-6} sec as local time horizon, and box template directions. It outputs that the forbidden state is not reached, hence the start-up regulation property is valid, i.e. after time $T_0 = 0.003$ sec, the voltage output remains inside the interval $[V_{ref} - \epsilon, V_{ref} + \epsilon]$ up to the local time horizon, as shown by Figure 5(a).

The *steady-state regulation* property consists of checking, starting from the initial state where the output is V_{ref} and the corresponding current (previous experiments establish that it is near 6.5), whether the voltage output remains near the desired value. The monitor used for this case is depicted in Figure 4(c). Similarly to the previous case, if the output exceeds the desired tolerance ϵ, the location *error* is reached. Clearly, the property is violated if this happens. The STC algorithm was used with same setting as in the start-up case, except for the initial state (i.e. here the initial output voltage is V_{ref} and the initial current is 6.5). Given $\epsilon = 0.5$, it returned that the property is valid up to the local time horizon, as shown by Figure 5(b).

As conclusion, we can observe that this benchmark is challenging for the STC algorithm since it switches frequently (every 25 μsec) compared to the overall dynamics, which is on the order of milliseconds. The approximation error incurred during the jump can quickly accumulate and render the analysis useless. Indeed, choosing an interval for the switching times makes the reachability analysis with SpaceEx/STC diverge. We suspect that a discrete-time approach may be more suitable, since the continuous evolution between switching times is relatively minor.



(a) Start-up regulation for $T_0 = 0.003$ sec and $\epsilon = 0.5$



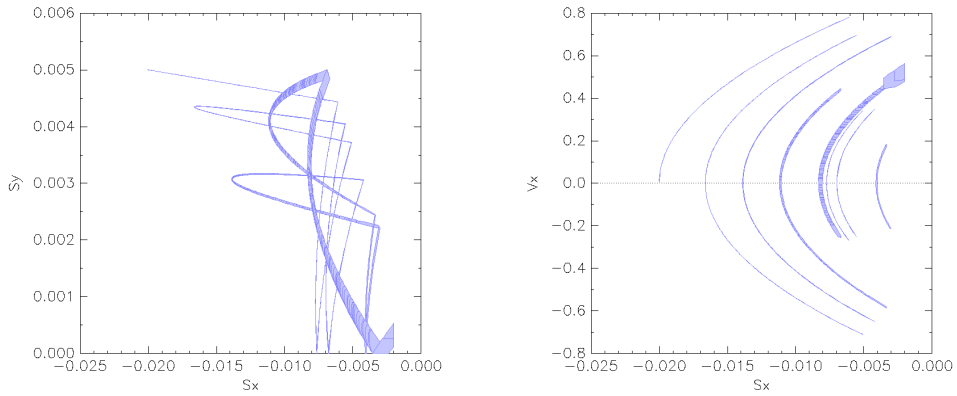
(b) Steady-state regulation for $\epsilon = 0.5$

Figure 5: The reachability set over the first 0.004 s. satisfies the start-up and the steady-state regulations

3 Motor-Transmission Drive System

This example is a model of the *Motor-Transmission Drive System* in [1]. Instead of considering the traditional powertrain, where a clutch disengages the power input of the engine during the shifting process, the rotor of the electric motor is directly connected to the input shaft of the transmission. For shifting gears, a sleeve is pushed by a shift actuator to first disengage from one gear and then to mesh with another gear. If the sleeve arrives at the target gear at an improper angular position, then it can delay the meshing process, or worse still, lead to physical impacts. The impacts make this a hybrid system: the sleeve moves continuously until it hits the gear, at which point its velocity changes (almost) instantaneously and then evolves continuously again. The state of the system consists of the horizontal position and velocity, p_x and v_x, and vertical position and velocity, p_y and v_y. We exploit the symmetry of the original problem to decrease the complexity of the reachability analysis. Note that a gear tooth extends from $p_y = -b$ to $p_y = b$ and is symmetric around $p_y = 0$. Due to symmetry, we can model only the upper half of a gear tooth, i.e., $0 \leq p_y \leq b$, and virtually extend the state space using mirroring: when the state hits the bounds $p_y = 0$ or $p_y = b$, the velocity is updated to $v_y := -v_y$. The resulting trajectories are equivalent to the original trajectories modulo changes in the sign of p_y and multiples of $2b$.

The system can be viewed as a 2D-version of a bouncing ball. The dynamics are two double integrators, which translate the acceleration of the sleeve into its horizontal and angular position. The collision of the sleeve hitting the gears leads to a change in velocity. Because of the stiffness of the gears, this collision is not a simple geometric reflection, i.e., the incoming and outgoing angles may not be the same. The collision equations are such that a minor uncertainty in time and position is amplified by the collision. This is shown in Fig. 6 for a single initial state. The reachable set is computed in SpaceEx with relatively high precision: using the STC scenario, we chose 512 uniformly distributed template directions and a flowpipe error of less than 0.0001. The approximation error remains quite small for the first 6 collisions, but on the



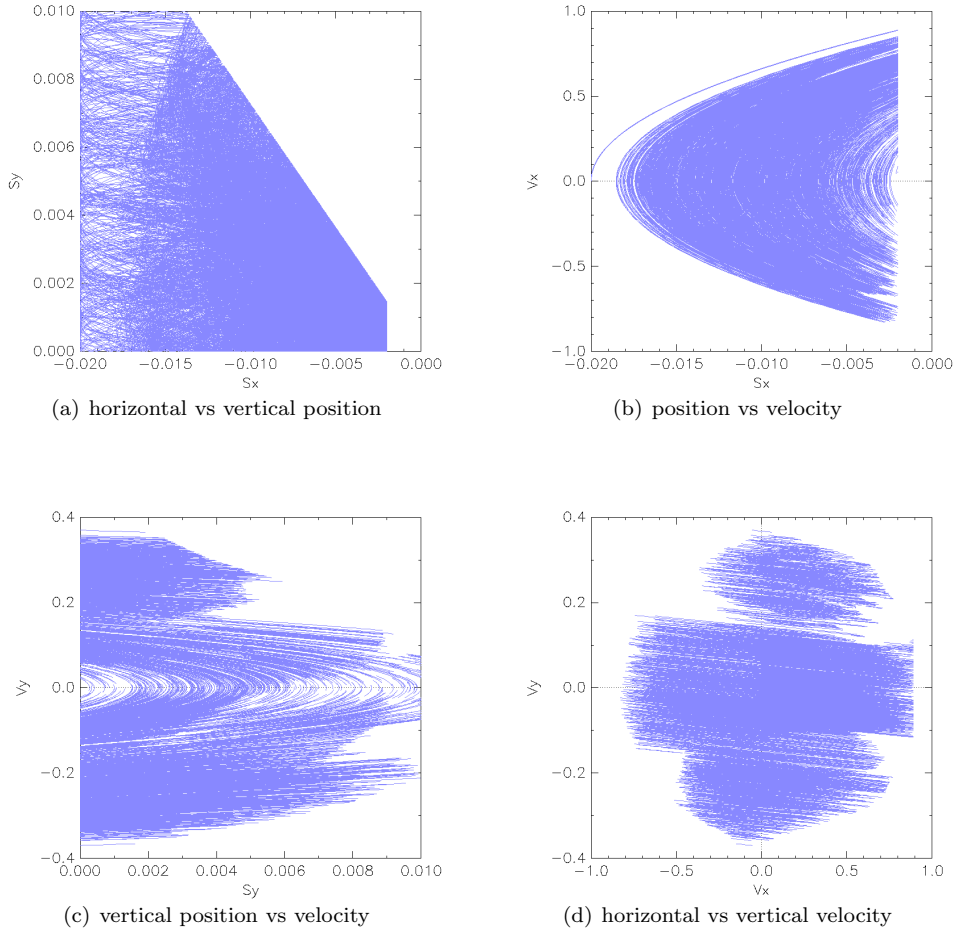
(a) horizontal vs vertical position (b) position vs velocity

Figure 6: Trajectories grazing the switching surface amplify the approximation error

7th and 8th collision it explodes. Further iterations will lead to excessive overapproximation. Note that the result is not much worse (but a lot faster) when using just octagonal directions. Combining the STC algorithm with synthesizing template directions for precise intersection as in [7] might improve the results.

Another problem of this system is the state explosion. The gear geometry is such that a state set can intersect with four guards simultaneously: the upper boundary of the gear, the lower boundary of the gear (in our case, the mirroring surface), the right hand boundary of the gear, and the upper boundary of the model space, which corresponds to the gear hopping to the next tooth. The right hand boundary is a cul-de-sac, since the gear is considered meshed. Each newly found state may therefore trigger four successor states, three of which may themselves spawn new successors. This leads us to believe that the problem is not very suitable to forward reachability. Other techniques, like directly synthesizing an invariant [11], might be more successful.

As an alternative to formal reachability, sampling the state space with simulation runs is possible. Indeed, the system is deterministic apart from the initial condition, so that every initial state leads to a single trajectory. It turns out that all these trajectories are acyclic and eventually end up in the meshed state, so we consider this system very suitable to trajectory-based techniques such as [4, 8, 2, 3]. Figure 7 show the results of running the SpaceEx simulation scenario. It uses the same mechanism as the reachability computations, but replaces the set-based successor computations by simulation trajectories obtained with an ODE solver (CVODE). The initial set of states is sampled randomly for a given number of points. The initial states were the horizontal position $p_x = -0.02$, horizontal velocity $v_x = 0$, the full range of possible vertical positions $p_y \in [0, 0.01]$, and the full range of vertical velocities $v_y \in [-0.08, 0.08]$ given in the benchmark description. A monitor automaton was used to measure the accumulated impact I, the number of hops (jumps from one tooth to another), and the number of collisions with the gear walls. The simulation was run until a fixed-point was reached, i.e., until all states enter



(a) horizontal vs vertical position

(b) position vs velocity

(c) vertical position vs velocity

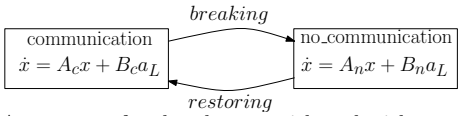
(d) horizontal vs vertical velocity

Figure 7: Trajectories obtained from sampling the initial states with 200 random points

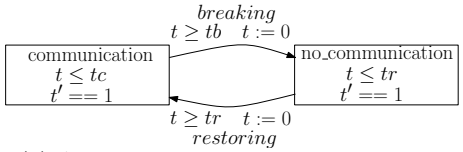
the meshed state. Sampling the initial states with 200 random points, we obtained a bound of $I \leq 48.7$ in the impact, a maximum of 1 hops to other gear teeth, and up to 10 impacts. The complexity of the obtained sets confirms our suspicion that the system is indeed challenging for forward reachability.

4 Networked Cooperative Platoon of Vehicles

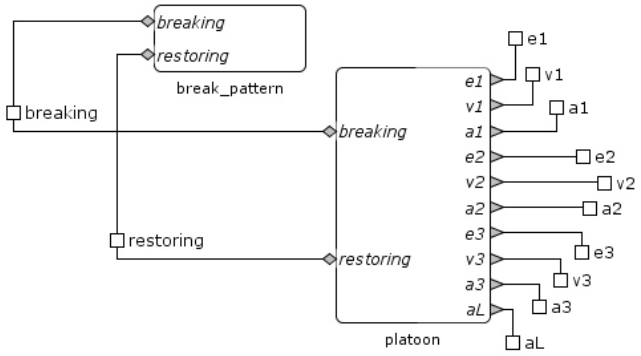
This benchmark consists of a platoon of three controlled vehicles with a manually driven leader [9]. The vehicles exchange information via a communication network, that may be subjected to failure by causing total loss of communication. The leader can proceed by changing speed, according to the possible acceleration $a_L \in [-9, 1] msec^{-2}$, while communication may be subjected to a failure every t_b sec, while when a breakdown happens communication are restored after t_r sec. Each vehicle i is modeled by the triple (e_i, v_i, a_i), where e_i represents the difference



(a) Automaton for the platoon with and without communication.



(b) Automaton for the breakdown pattern.



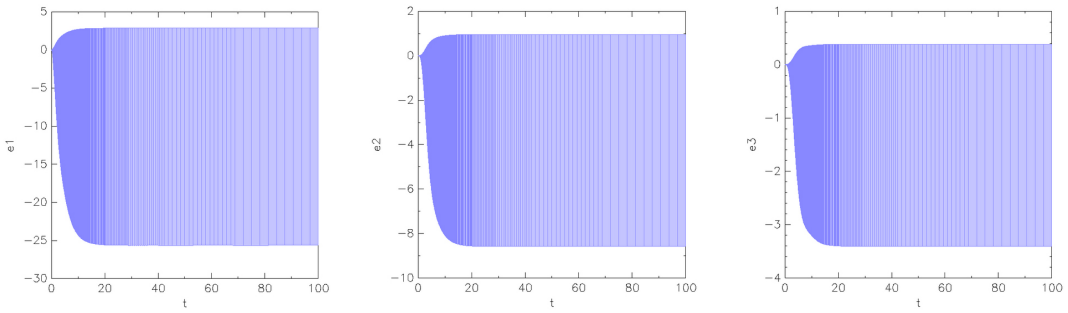
(c) SpaceEx network for modeling the global system.

Figure 8: SpaceEx design for the Networked Cooperative Platoon of Vehicles

between the distance d_i of vehicle i to its predecessor and a reference distance $d_{ref,i}$. Variables v_i and a_i represent, respectively, the relative velocity and acceleration of vehicle i with respect to the predecessor. Let $x = (e_i, v_i, a_i)$ be the vector of the variables, the closed loop system can be described by the following differential equation $\dot{x} = Ax + Ba_L$, where A is a constant system matrix, B is a constant input matrix and a_L the acceleration of the leader. The goal is to determine the minimum allowable safe gaps e_1, e_2 and e_3 among three vehicles, and here is achieved by performing reachability analysis.

The global system is modeled in a compositional way by combining two automata to design, respectively, the breakdown pattern and the trends of the vehicles. Figure 8 shows the SpaceEx model and its components. The automaton *platoon* shown in Figure 8(a) models the dynamics of the platoon. It consists of the location *communication* to model the case with perfect communication among vehicles and the location *no_communication* to model the case with no exchange of information. The flows of the locations are set according matrices A_c, B_c (resp., A_n and B_n) given by [9], for the case of perfect communication (resp., without communication).

The switches between locations are controlled by the automaton in Figure 8(b), which drives, via the synchronization labels *breaking* and *restoring*, the automaton *platoon* in changing the mode from the perfect to the broken communication (and vice versa). The breakdown pattern can be easily modeled by setting the value of the additional constant tc, once parameters tb and tr are given. For example, the ideal case (i.e. without communication failures) can be modeled by setting $tc < tb$. This configuration, together with the invariant $t \leq tc$, is such that the guard $t \geq tb$ can never be satisfied, thus preventing the jump to location *no_communication*. Figure 9 shows, for this case, the projection of the reachability set over e_1, e_2 and e_3 on the time, computed in 471 sec by SpaceEx STC algorithm, using 0.1 as flowpipe tolerance on octagonal



(a) $e_1 \geq -25.62m$. (b) $e_2 \geq -8.57m$. (c) $e_3 \geq -3.40m$.

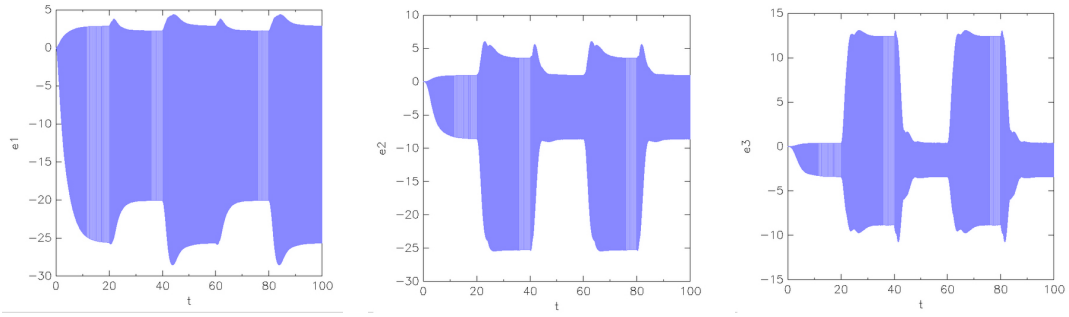
Figure 9: The reachability analysis, for the ideal case, along the first 100 sec. shows the lower bounds on e_1, e_2 and e_3. The plot is obtained by adding a global clock to the model

directions and a local time horizon of 100 sec. In this case we can establish $min(e_1) = -25.62m$, $min(e_2) = -8.57m$ and $min(e_3) = -3.40m$, meaning that the minimum safe distances between consecutive vehicles are $25.62m$, $8.57m$ and $3.40m$ respectively. We obtained the same result by using box directions and by increasing the flowpipe tolerance up to 1.

In the *may failure case*, the system may be subjected to a communication failure after tb seconds, followed by restoration of communications after tr seconds, can be modeled by setting $tc > tb$. In this case, when $t \geq tb$ the automaton may either jump to location *no_communication* or continue with perfect communication. In contrast, the *must failure case* can be modeled by setting $tc = tb$. This forces the system to perform the jump to *no_communication* after tb seconds.

The reachability analysis, with several values of times tb and tr, shows that the safe distances e_i are the same for the may and the must cases. This is because we are interested in a safety property, achieved by the minimization of the variables d_i. But the minimum values of d_i come from the worst case, that trivially is when the system must have a communication failure. Hence, may and must cases require the same safe distances. The case with possible breakdown of 20 sec, every 20 sec, (i.e. $tb = tr = 20$) is depicted by Figure 10, computed in 2390 sec by STC algorithm with 0.1 flowpipe tolerance, 20 sec of local time horizon and octagonal directions. The minimum values for e_1, e_2 and e_3 are respectively, $-28.5m$, $-25.4m$ and $-10.7m$.

Additional experiments have confirmed the general intuition that the minimum safe distances e_i became greater and greater either by reducing the time tb of perfect communication, or by increasing the time tr in which the system can not exchange information. In particular, we performed reachability analysis by changing the values of t_b and t_r such that $t_b + t_r = 40$ (as the previous case of $t_b = t_r = 20$). This has shown that the distance e_1 is less affected by the failure pattern, while in contrast e_3 is the most sensitive. The second remark is that the safe distances remain rather stable when t_b and t_r assume values between 10 and 30, while changes are more evident by increasing t_b from 37 to $t_b = 40$ (i.e. ideal case) and by decreasing t_r from 3 towards 0. Note that the accuracy of the result here, unlike the ideal case, is affected by the choise of the template directions. Indeed, for the case $tb = tr = 20$, the minimum safe distances obtained by using box instead of octagonal directions, are $-35.05m$, $-32.31m$ and $-18.42m$. This is because, while the ideal case is such that no switches happen, here several transitions are fired, each introducing an approximation error.



(a) $e_1 \geq -28.5m$. (b) $e_2 \geq -25.4m$. (c) $e_3 \geq -10.7m$.

Figure 10: Lower bounds on e_1, e_2 and e_3, for a possible breakdown every 20 sec with 20 sec of recovering time, obtained from the reachable set along the first 100 sec. The plot is obtained by adding a global clock to the model.

5 Conclusion

In this paper we present SpaceEx models and results for selected benchmarks of ARCH14. For benchmarks with Simulink models we used a prototype of the translation tool *Simulink to SpaceEx (SL2SX)* to help with the process of building the corresponding SpaceEx model. The SpaceEx models are modular, preserving the structure of the source model (i.e. Simulink diagram) or reflecting the physical components and phenomena (collision).

The reachability analysis delivered two types of results. The DC-to-DC benchmark case illustrates how reachability, with the help of a monitor, can be used to verify safety properties. On the other hand, the reachable set of the network platoon is used to establish the minimum safe distances among vehicles, i.e., quantitative measurements. A monitor is a hybrid automaton that switches to an error location, declared as forbidden state, when the property under investigation is violated. The construction of monitors for very simple properties is straightforward, but further investigation is necessary develop monitors for more complex properties and ensure that the monitor has as little adverse impact on the computational cost as possible.

While some benchmarks could be handled to our satisfaction, several still pose significant challenges. In models with a relatively minor continuous evolution between discrete jumps, the approximation error can quickly be accumulated and render the analysis useless. Another limitation is that a potential state space explosion can arise from those models whose state set intersects several guards at the same time.

References

[1] Hongxu Chen, Sayan Mitra, and Guangyu Tian. Motor-transmission drive system. In *1st Workshop on Applied Verification for Continuous and Hybrid Systems (ARCH)*. http://cps-vo.org/node/12109, 2014.

[2] Thao Dang and Tarik Nahhal. Coverage-guided test generation for continuous and hybrid systems. *Formal Methods in System Design*, 34(2):183–213, 2009.

[3] Alexandre Donzé. Breach, a toolbox for verification and parameter synthesis of hybrid systems. In *Computer Aided Verification*, pages 167–170. Springer, 2010.

[4] Georgios E Fainekos, Antoine Girard, and George J Pappas. Temporal logic verification using simulation. In *Formal Modeling and Analysis of Timed Systems*, pages 171–186. Springer, 2006.

[5] G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler. Spaceex: Scalable verification of hybrid systems. In *CAV 11: Proc. of 23rd Conf. on Computer Aided Verification*, pages 379–395, 2011.

[6] Goran Frehse and Matthias Althoff, editors. *1st Workshop on Applied Verification for Continuous and Hybrid Systems (ARCH)*. http://cps-vo.org/group/ARCH/benchmarks, 2014.

[7] Goran Frehse and Rajarshi Ray. Flowpipe-guard intersection for reachability computations with support functions. In *Analysis and Design of Hybrid Systems (ADHS)*, pages 94–101, 2012.

[8] A Agung Julius, Georgios E Fainekos, Madhukar Anand, Insup Lee, and George J Pappas. Robust test generation and coverage for hybrid systems. In *Hybrid Systems: Computation and Control*, pages 329–342. Springer, 2007.

[9] Ibtissem Ben Makhlouf and Stefan Kowalewski. Networked cooperative platoon of vehicles for testing methods and verification tools. In *1st Workshop on Applied Verification for Continuous and Hybrid Systems (ARCH)*. http://cps-vo.org/node/12115, 2014.

[10] Luan Viet Nguyen and Taylor T. Johnson. Dc-to-dc switched-mode power converters. In *1st Workshop on Applied Verification for Continuous and Hybrid Systems (ARCH)*. http://cps-vo.org/node/12113, 2014.

[11] SV Rakovic, P Grieder, M Kvasnica, DQ Mayne, and M Morari. Computation of invariant sets for piecewise affine discrete time systems subject to bounded disturbances. In *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, volume 2, pages 1418–1423. IEEE, 2004.