Running SpaceEx on the ARCH14 Benchmarks

Stefano Minopoli and Goran Frehse

Université Joseph Fourier - Grenoble 1 / Verimag,
{stefano.minopoli,goran.frehse}@imag.fr

Abstract. In this paper, we present experimental results from running the verification tool 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.

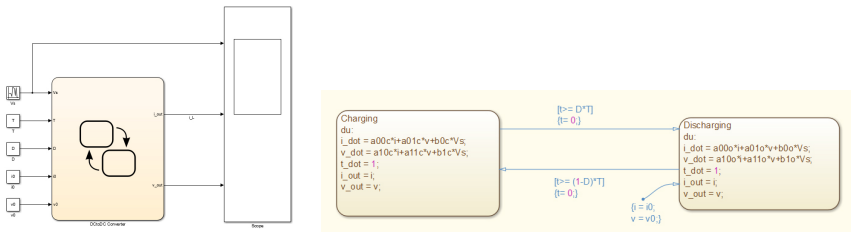
1 Introduction

This paper presents some results from running the verification tool 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 because they have piecewise affine dynamics, are closed (the model includes the controller), and constitute reachability problems. The models and configuration files for running the examples in SpaceEx are available as attachment.

2 DC-to-DC Switched-Mode Power Converters

This example originates from the *DC-to-DC switched-mode power converters* described in [11], 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 by switching at a high (typically kilohertz) frequency between low and high voltage levels. The continuous variables are the currents and voltages in the circuit.

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 new tool *SL2SX* to translate the Simulink model from [11] into a SpaceEx model. The tool translates each Simulink block into a SpaceEx component, preserving the structure and layout of the model. Not all Simulink blocks are supported (some are not readily modeled as hybrid automata), and need to be completed by the user. In this example, the user must delete the network component corresponding to the scope block and must attribute numerical values to the constants in blocks. Figure 2(a)



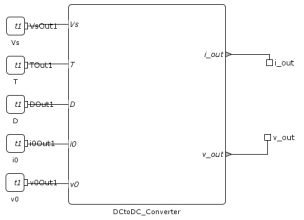
(a) main system (b) Stateflow block

Fig. 1. Simulink diagram for DC-to-DC switched-mode power converters

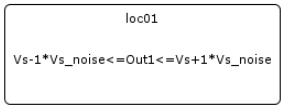
shows the resulting translation of the Simulink main system from Fig. 1(a). The current version of the translator does not support the random number block and the Stateflow block. The hybrid automaton shown in Figure 2(b) is our abstraction for the random number block. It consists of a single location whose invariant guarantees that the values of variable $Out1$ (used to model the block output) are non-deterministically chosen inside an interval, according to block parameters. Figure 2(c) shows a hybrid automaton for modeling the Stateflow block. It consists of the same locations (i.e. *Charging* and *Discharging*), flows and transitions. Invariants have been added to mimick the Simulink/Stateflow must-semantics.[1]

Figure 3(a) shows the result of a Simulink simulation of the voltage level for a time horizon of 0.004 sec. We use SpaceEx to compute the reachable set for the voltage for the same initial conditions. The initial state is set according to Stateflow initial conditions (location *Discharging*, current equal to 6 and voltage equal to 17.5). The reachable set is computed by the STC algorithm, parametrized with 0.01 as flowpipe tolerance, 4×10^{-6} sec as local time horizon, and using 64 octagonal template directions. Figure 3(b) shows the reachable set, which is computed 926.0 sec on a standard laptop. The comparison between this set and the Simulink simulation in Fig. 3(a) shows the error accumulation during the reachability analysis. This benchmark is challenging for the STC algorithm since it switches frequently, every $25\,\mu sec$, 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.

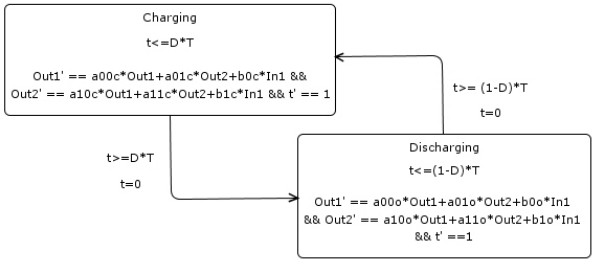
[1] In Simulink/Stateflow, transitions must fire as soon as the state enters the guard set, while in hybrid automata with may-semantics the system may remain in the location as long as the invariant is satisfied. For details, see [10].



(a) main system



(b) abstraction of the random number block



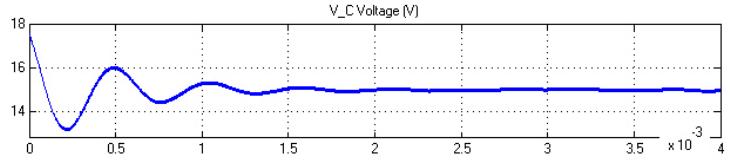
(c) HA model of the stateflow block.

Fig. 2. SX model from *SL2SX* after completion by user

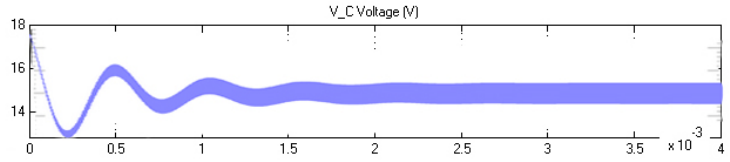
3 Motor-Transmission Drive System

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

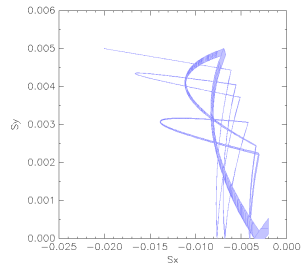


(a) Simulink simulation

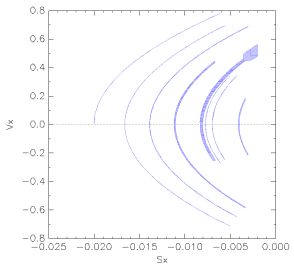


(b) reachable set

Fig. 3. Simulation and reachable set for the voltage over a time horizon of 0.004 s.



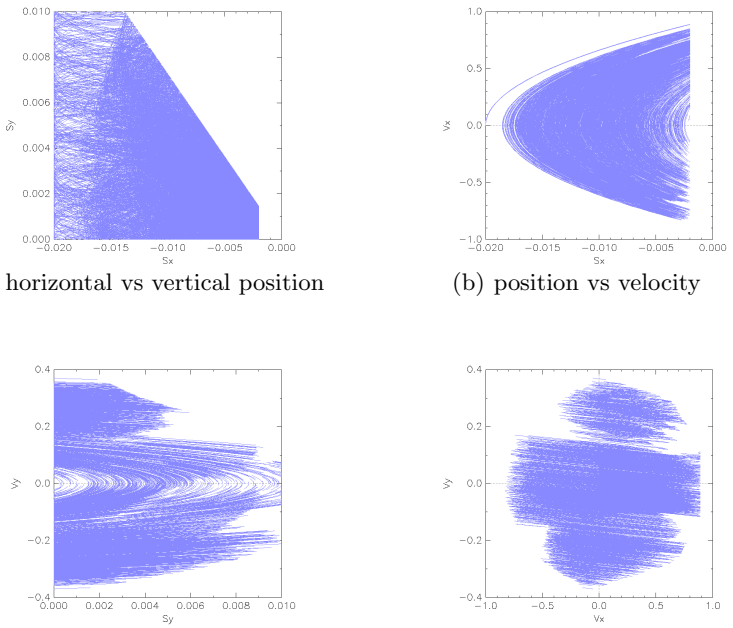
(a) horizontal vs vertical position



(b) position vs velocity

Fig. 4. Trajectories grazing the switching surface amplify the approximation error

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. 4 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 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.



(a) horizontal vs vertical position

(b) position vs velocity

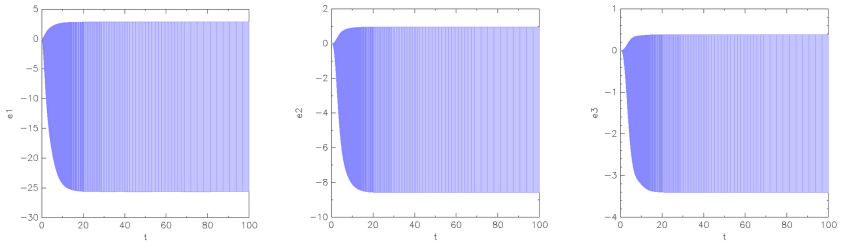
(c) vertical position vs velocity

(d) horizontal vs vertical velocity

Fig. 5. Trajectories obtained from sampling the initial states with 200 random points

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 [12], 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 5 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 $p_x = -0.02$, hor-



(a) $e_1 \geq -25.629m$. (b) $e_2 \geq -8.582m$. (c) $e_3 \geq -3.410m$.

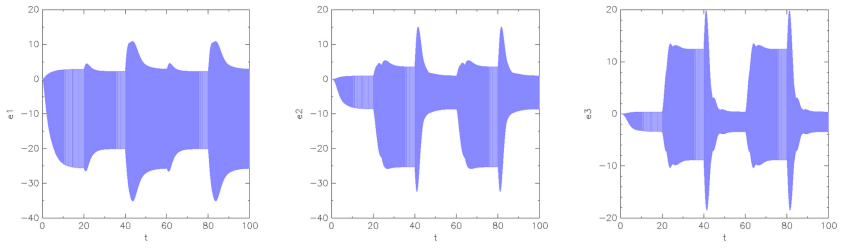
Fig. 6. Bounds on e_1, e_2 and e_3 without communication failure, obtained from the reachable set over time

izontal 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 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 total loss of communication. The leader can change the speed with an acceleration in the range $a_L \in [-9, 1]$ m/s^2. Communication may be subjected to a failure every c_1 seconds and is restored after c_2 seconds. Each vehicle i is modeled by the triple (e_i, v_i, a_i), where e_i represents the difference 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.

The goal is to determine the minimum allowable safe gaps among the vehicles. We use reachability analysis to establish, for each i, the minimum value reachable for e_i. We consider the ideal case without communication failure and the case with possible failure and parameters $c_1 = c_2 = 20\,\text{s}$. The ideal case is modeled by a hybrid automaton with a single location (a purely continuos system), while for the case with breakdown an additional location models the period without communication. The flow is a linear function as given in [9]. For the ideal case, Figure 6 shows the projection of the reachability set over e_1, e_2 and e_3 over the time, computed in 5.9 sec with the STC algorithm, using 0.01 as flowpipe tolerance, octagonal directions, and a local time horizon of 100 s. In this case



(a) $e_1 \geq -35.442m$. (b) $e_2 \geq -32.759m$. (c) $e_3 \geq -19.078m$.

Fig. 7. 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 over time

Table 1. Minimum safe distances for communication failures at different times t.

Failure time	e_1	e_2	e_3
$t = 1s$	$\sim 40m$	$\sim 55m$	$\sim 42m$
$t = 3s$	$\sim 40m$	$\sim 40m$	$\sim 29m$
$t = 4s$	$\sim 38m$	$\sim 32m$	$\sim 20m$
$t = 5s$	$35.4m$	$32.7m$	$19.0m$
$t = 21s$	$35.4m$	$32.7m$	$19.0m$
$t = 22s$	$25.6m$	$8.5m$	$3.4m$

we can establish $min(e_1) = -25.629\,m$, $min(e_2) = -8.582\,m$, and $min(e_3) = -3.410\,m$. The reachable set for the case with possible breakdown is depicted in Figure 7, computed in 188.9 s with the STC algorithm, with flowpipe tolerance 0.1, box directions, and a local time horizon of 21 s. The minimum values for e_1, e_2, and e_3 are, respectively, $-35.442\,m$, $-32.759\,m$, and $-19.078\,m$.

We also investigated other failure patterns for communication. In the *must failure case* when the system is subjected to a communication failure exactly every t seconds. In the *may failure case*, the system can lose communication after t seconds or later. In the following experiments, we found the same safety results for both may and must cases. A may or must failure that happens every $t \geq 22$ s has not impact on the safe distances. Hence, they are equivalent to the ideal case (see Figure 6) when a communication failure can never happen. The analysis with $t \in [5, 21]$ s gives the same minimum safe distances as the case with $t = 20$ s from Figure 7. For cases with $t \leq 4$ s, the minimum safe distances increase as t is reduced. Table 1 shows a summary of the performed experiments (recall that may and must cases are characterized by the same results).

Bibliography

[1] Hongxu Chen, Sayan Mitra, and Guangyu Tian. Motor-transmission drive system. In Frehse and Althoff [6].

[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*, 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 Frehse and Althoff [6].

[10] Stefano Minopoli and Goran Frehse. Non-convex invariants and urgency conditions on linear hybrid automata. In *Formal Modeling and Analysis of Timed Systems*, pages 176–190, 2014.

[11] Luan Viet Nguyen and Taylor T. Johnson. Dc-to-dc switched-mode power converters. In Frehse and Althoff [6].

[12] 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.