Techniques and Tools for Hybrid Systems
Reachability Analysis

Erika Ábrahám
RWTH Aachen University, Germany

Hybrid systems are systems with combined discrete and continuous behaviour, typical examples being physical systems controlled by discrete controllers. Such systems can be found in various fields such as aviation, control engineering, medicine, or the wide field of cyber-physical systems.

The increasing relevance of hybrid systems, especially systems which interact with humans, requires careful design and proper safety verification techniques. Whereas the verification of purely continuous or purely discrete systems are already well-established research areas, the combination of discrete and continuous behaviours brings additional challenges for formal methods.

Logical formalisations are used in theorem-proving-based tools like KeYmaera [12], Ariadne [4], or the Isabelle/HOL-based tool described in [10]. Other tools like dReach [11], iSAT-ODE [6] and HSolver [13] also use logical characterisations but in combination with interval arithmetic and SMT solving. The tool C2E2 [5] uses validated numerical simulation; Bernstein expansion is implemented in [15]. This variety is complemented by approximation methods like hybridization, linearisation and abstraction techniques to increase the applicability of hybrid systems verification.

In this tutorial we focus on flowpipe-construction-based techniques and their implementation. As the reachability problem for hybrid systems is in general undecidable, flowpipe-construction-based reachability analysis techniques usually compute over-approximations of the set of reachable states of hybrid systems: starting from some initial sets, their time trajectories (flowpipes) and successors along discrete transitions (jump successors) are over-approximated in an iterative manner. Some tools in this area are Cora [1], Flow* [3], HyCREATE [9], HySon [2], SoapBox [8], and SpaceEx [7].

The development of such tools is effortful, as datatypes for the underlying state set representations need to be implemented first. Our free and open-source C++ library HyPro [14] provides implementations for the most prominent state set representations, with the aim to offer assistance for the rapid implementation of new algorithms by encapsulating all representation-related issues and allowing the developers to focus on higher-level algorithmic aspects.

In this tutorial we give an introduction to hybrid systems, and to flowpipe-construction-based algorithms for computing their reachable state sets. After discussing theoretical aspects, we shortly describe available tools and introduce

* This work was supported by the German Research Council (DFG) in the context of the HyPro project.
in more detail our HyPro library, explain its functionalities, and give some examples to demonstrate its usage.

References