Hybrid Systems: The Early Years

A Tribute to Oded Maler

Rajeev Alur
University of Pennsylvania
Real-Time: Theory in Practice, REX Workshop
Mook, The Netherlands

Meeting: June 1991; Proceedings: Summer 1992

J. W. de Bakker, W.P. de Roever, G. Rozenberg
A Melody of Formalisms for Timed Systems

- Timed automata (Alur, Dill)
- Timed temporal logics (Alur, Henzinger)
- Real-time logic RTL (Jahanian, Mok)
- Timed CSP (Reed, Roscoe)
- Timed transition systems (Henzinger, Manna, Pnueli)
- Timed I/O automata (Lynch, Vaandrager)
- Communicating shared resources (Gerber, Lee)
- TLA (Abadi, Lamport)

and many more

Lively discussion:

How to add continuous dynamics to formal models?
The Grand Challenge for Hybrid Systems!!

Initial position \( y \)
Speed \( u \)
Start time \( \Delta \)

Initial position \( x \)
Speed \( v \)
Start time 0

Does the cat catch the mouse?
We propose a framework for the formal specification and verification of *timed* and *hybrid* systems. For timed systems we propose a specification language that refers to time only through *age* functions which measure the length of the most recent time interval in which a given formula has been continuously true. We then consider hybrid systems, which are systems consisting of a non-trivial mixture of discrete and continuous components, such as a digital controller that controls a continuous environment. The proposed framework extends the temporal logic approach which has proven useful for the formal analysis of discrete systems such as reactive programs. The new framework consists of a semantic model for *hybrid time*, the notion of *phase transition systems*, which extends the formalism of discrete transition systems, an extended version of Statecharts for the specification of hybrid behaviors, and an extended version of temporal logic that enables reasoning about continuous change.
Hybrid Traces: Modeling Executions of Hybrid Systems

Time moment = (real time t, discrete time n)
Lexicographic order over time moments
Phase Transition Systems

- State variables V partitioned into $V_c$ and $V_d$
- Assertion I specifying initial states
- Set T of (discrete) transitions, where each transition $\tau$ is a function from states to sets of states (transition $\tau$ is enabled in state $s$ if $\tau(s)$ is non-empty)
- Lower bound $l_\tau$ for each transition $\tau$ (specifies how long a transition must be enabled before it is taken)
- Upper bound $u_\tau$ for each transition $\tau$ (specifies a bound on how long a transition can be enabled without being taken)
- Set A of (continuous) activities, where each activity is a conditional differential equation: $\text{guard}(V_d) \rightarrow \{ \frac{dy}{dt} = \exp | y \in V_d \}$

Formal semantics of a phase transition system = Set of hybrid traces
Modeling the Grand Challenge

Figure 3: Specification of Cat and Mouse.
Proof Rules for Verification

Generalization of inductive invariants for fair transition systems
Relies on explicit solution of differential equations
An Invariant for the Cat and Mouse Specification

A more interesting invariant concerns the Cat and Mouse example. Here we may want to determine conditions under which the cat will never catch the mouse. A simple calculation leads to the requirement

\[
\frac{X_0}{V_m} < \Delta + \frac{X_0}{V_c}
\]  

(1)

The condition states that the time it takes the mouse to reach the wall is smaller than the time it takes the cat to reach the wall. Since both run at constant speed, this guarantee that they will not meet, except at the wall, the mouse arriving there first.

Assume that condition (1) is given and that \( \Delta > 0 \). We then would like to establish the invariant

\[
\text{Cat.running} \land (x_c = x_m) \Rightarrow x_m = 0.
\]  

(2)

To prove this invariant we use an auxiliary assertion \( \varphi \) given by a conjunction of the following implications

\[
\begin{align*}
\text{Mouse.rest} & \rightarrow x_m = X_0 \\
\text{Mouse.running} & \rightarrow x_m = X_0 - V_m \cdot \Gamma(\text{Mouse.running}) \geq 0 \\
\text{Mouse.safe} & \rightarrow x_m = 0 \\
\text{Cat.rest} & \rightarrow x_c = X_0 \\
\text{Cat.running} & \rightarrow x_c = X_0 - V_c \cdot \Gamma(\text{Cat.running}) \geq 0 \\
\text{Cat.rest} & \rightarrow \text{Mouse.rest} \lor \text{Mouse.running} \lor \text{Mouse.safe} \\
\text{Cat.running} & \rightarrow \text{Mouse.running} \lor \text{Mouse.safe} \\
\text{Mouse.rest} \land \text{Cat.rest} & \rightarrow \Gamma(\text{Mouse.rest}) = \Gamma(\text{Cat.rest}) = 0 \\
\text{Mouse.running} \land \text{Cat.rest} & \rightarrow \Gamma(\text{Mouse.running}) = \Gamma(\text{Cat.rest}) \leq \Delta \\
\text{Mouse.running} \land \text{Cat.running} & \rightarrow \Gamma(\text{Mouse.running}) = \Gamma(\text{Cat.running}) + \Delta
\end{align*}
\]  

(3-7)

Assuming that \( \varphi \) has been shown to satisfy premises 12-14, we will show that it implies \( q : (\text{Cat.running} \land (x_c = x_m)) \rightarrow x_m = 0 \). By implication (6), when the cat is running the mouse is in either state running or in state safe. If it is in state safe then, by implication (4), \( x_m = 0 \). If it is in state running, then the assumption \( x_c = x_m \) with implications (3), (5), and (7), leads to \( V_c \cdot \Gamma(\text{Cat.running}) = V_m \cdot (\Gamma(\text{Cat.running}) + \Delta) \). From this, we can conclude \( V_c > V_m \) and

\[
\Gamma(\text{Cat.running}) = \frac{V_m \cdot \Delta}{V_c - V_m}.
\]

On the other hand, from implication (5) we obtain \( X_0 - V_c \cdot \Gamma(\text{Cat.running}) \geq 0 \) which can be written as

\[
\Gamma(\text{Cat.running}) \leq \frac{X_0}{V_c}
\]

Comparing the equality and inequality involving \( \Gamma(\text{Cat.running}) \), we obtain

\[
\frac{V_m \cdot \Delta}{V_c - V_m} \leq \frac{X_0}{V_c}.
\]
Hybrid Systems Workshop
Technical Univ of Denmark, Lyngby

Meeting: October 1992; Proceedings: Summer 1993

R. L. Grossman, A. Nerode, A. P. Ravn, and H. Rischel
We introduce the framework of *hybrid automata* as a model and specification language for hybrid systems. Hybrid automata can be viewed as a generalization of timed automata, in which the behavior of variables is governed in each state by a set of differential equations. We show that many of the examples considered in the workshop can be defined by hybrid automata. While the reachability problem is undecidable even for very restricted classes of hybrid automata, we present two semidecision procedures for verifying safety properties of *piecewise-linear* hybrid automata, in which all variables change at constant rates. The two procedures are based, respectively, on minimizing and computing fix-points on generally infinite state spaces. We show that if the procedures terminate, then they give correct answers. We then demonstrate that for many of the typical workshop examples, the procedures do terminate and thus provide an automatic way for verifying their properties.
Hybrid Systems: Towards a sustained discipline

  P. A. Antsaklis, W. Kohn, A. Nerode, and S. Sastry

- Hybrid Systems III: Verification and Control
  R. Alur, T. A. Henzinger, and E. Sontag

- Hybrid Systems IV, Ithaca, October 1996 (LNCS 1723, 1997)
  P. A. Antsaklis, W. Kohn, A. Nerode, and S. Sastry

- Hybrid and Real-Time Systems, Grenoble, March 1997 (LNCS 1201)
  O. Maler

- Hybrid Systems: Computation and Control (HSCC)
  First International Workshop, Berkeley, April 13-15, 1998
  T. A. Henzinger and S. Sastry (LNCS 1386)
In this paper we discuss the problem of calculating the reachable states of a dynamical system defined by ordinary differential equations or inclusions. We present a prototype system for approximating this set and demonstrate some experimental results.

Test-of-Time Award, HSCC 2019
Example: Airline safety

\[ \dot{x}_1 = -\frac{a_D x_1^2}{m} - g \sin x_2 + \frac{u_1}{m} \]

\[ \dot{x}_2 = \frac{a_L x_1 (1-cx_2)}{m} - \frac{g \cos x_2}{x_1} + \frac{A_L c x_1}{m} u_2 \]

Evolution of velocity and flight path angle

Model based on:

Multiobjective hybrid controller synthesis
J. Lygeros, C. Tomlin, and S. Sastry,
Proc. HART, 1997
Example: Airline safety

\[ \dot{x}_1 = -\frac{a_D x_1^2}{m} - g \sin x_2 + \frac{u_1}{m} \]

\[ \dot{x}_2 = \frac{a_L x_1 (1 - c x_2)}{m} - \frac{g \cos x_2}{x_1} + \frac{A_L c x_1}{m} u_2 \]

Fig. 7. Airplane Safety
Recap: Hybrid Systems, 1990s

Real-Time: Theory in Practice, REX Workshop
Mook, The Netherlands, 1991

Modeling and semantics

Hybrid Systems: Computation and Control
First International Workshop, Berkeley, 1998

Computational tools and case studies