**Context: Formal verification of real-time systems**

Critical systems involve **timing constraints** and concurrency

Bugs can be **dramatic** (risk of loss of lives or huge financial loss)

⇒ need for formal verification

**Objective**

*“Given a model (mathematical representation) of a system and a specification, synthesize timing constraints (parameters) guaranteeing that the system meets its specification”*

**The formalism: parametric timed automata**

Extension of finite-state automata with **clocks** and **parameters**

![Example: a coffee machine modeled using a parametric timed automaton](image)

**IMITATOR in a nutshell**

A **parametric** timed model checker

**Input**

- a **real-time system** modeled using parametric timed automata
- a **specification**

**Output**

- Conditions on the parameters formally guaranteeing the system correctness

**Constraint for which the system meets its specification:**

\[ p_1 + p_2 > 2 \quad p_3 & \quad p_3 > 0 \]

- **Graphical visualization**

**What’s inside?**

Implemented in OCaml

Relying on the Parma Polyhedra Library

Large repository of benchmarks

Distributed under the GNU-GPL license

Try it!  

www.imitator.fr

**Notable applications of IMITATOR**

- **Hardware verification**
  - Collaboration with ST-Microelectronics
- **Scheduling for aerospace**
  - Collaboration with ArianeGroup
- **Scheduling under uncertainty**
  - Solved an industrial challenge by Thales
- **Testing software product lines**
- **Analysis of music scores**
  - IRCAM, Paris, France
- **Monitoring** real-time systems
  - Applications to automotive industry
  - best paper award @ ICECCS 2018

**What’s next?**

- Integration to real-time systems formalisms
  - E.g. Thales’ “Time4sys”
- Beyond timed automata
  - Linear hybrid automata
  - Can represent more subtle variations of speed, temperature, energy...
  - Potential: bring (more) formal methods to automotive industry

**Bibliography**


