Formal Verification of Real-time Systems Under Uncertainty

Étienne André¹ Giuseppe Lipari² and Youcheng Sun²

¹LIPN, CNRS UMR 7030, Université Paris 13, France
²CRIStAL – UMR 9189, Université de Lille, Inria, CNRS, France
Scuola Superiore Sant’Anna, Pisa, Italy

Context: Formal Verification of Real-Time Systems

Critical systems involve timing constants and concurrency

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

⇒ Need for formal verification

Problem: what if the system constants are uncertain or are not yet known?

Solution: parametric verification

- Timing constants become parameters

Objective: derive values for these parameters ensuring the absence of bug (usually under the form of a set of constraints)

Parametric Timed Automata (PTA) [Alur et al., 1993]

- Finite automata (sets of locations and actions) extended with:
  - Clocks: real-valued variables evolving linearly
  - Parameters: unknown constants
- Example: Coffee machine

$$y = p_1 \text{ coffee!}$$

$$\begin{align*}
& \text{pressure} \; \Rightarrow \; x := 0 \\
& \text{sugar} \; \Rightarrow \; y := 0 \\
& \text{cup} \; \Rightarrow \; x := 0
\end{align*}$$

IMITATOR: Parameter Synthesis for Critical Systems

Input: a real-time system modeled by a network of PTA

Output: a constraint over the parameters guaranteeing the system correctness (e.g., non-reachability of some unsafe state)

Several algorithms:

- Non-reachability synthesis
- Parametric language preservation
- Behavioral cartography

TryIMITATOR! [André et al., 2012]

- Entirely written in OCaml
- Graphical outputs (behaviors, parameter constraints, etc.)
- Large repository of benchmarks
  - Asynchronous hardware circuits, scheduling problems, communication protocols, train controllers… and more!
- Available for free under the GNU-GPL license

What’s next?

- Improved optimizations to address scalability
- Distributed and multi-core algorithms
- An input language forIMITATOR dedicated to real-time systems
  - Followed by a translation to PTA

A Case Study: The FMTV Challenge

- A problem proposed by Thales Research & Technology for the video capture in an aerial video system (2014)

- A distributed video processing system (abstract view)

  - CPU1
  - CPU2
  - CPU3
  - CPU4

  sync

  register

  buffer

  τ

  τ

  τ

  τ

\[ T4 \text{process} \leq 40 \text{ ms} \]

\[ T4 \text{wait} \leq 150 \text{ ms} \]

\[ \text{frame} \text{init} \leq 3 \text{ ms} \]

\[ \text{buff} \text{size} \leq 3 \text{ ms} \]

Our Solution: Parametric Analysis [André et al., 2015]

- Task periods are modeled as parameters
  - E.g., \( P4 \text{uncertain} \in [40 - 0.004 \text{ ms}, 40 + 0.004 \text{ ms}] \)
- Another parameter: the end-to-end latency \( E2E \)
  - To focus on the \( E2E \) of an arbitrary frame (denoted as target)
- Some of the PTA modeling the system (for \( n = 1 \))
  - The system status is initialized to be arbitrary so that the worst-case and best-case scenarios for \( E2E \) will be included

PTA model for task \( T4 \)

\[ \text{E2E} \leq 40 \text{ ms} + 0.004 \text{ ms} \]

\[ \text{buffer} \text{size} \leq 150 \text{ ms} \]

\[ \text{frame} \text{init} \leq 40 \text{ ms} \]

\[ \text{buff} \text{size} \leq 3 \text{ ms} \]

- The end-to-end latency results returned byIMITATOR
  - \( E2E \leq 63 \text{ ms}, 145.008 \text{ ms} \) (for \( n = 1 \))
  - \( E2E \leq 63 \text{ ms}, 225.016 \text{ ms} \) (for \( n = 3 \))
- Runtime costs: 7.908 s with \( n = 1 \) and 115.247 s with \( n = 3 \)

Conclusion

- Solved a problem with uncertain timing constants using parametric analysis, which turned out to be an efficient option

References