Automated Verification of Cyber-Physical Systems

A.A. 2024/2025

Corso di Laurea Magistrale in Informatica

Basic Notions

Igor Melatti

Università degli Studi dell'Aquila

Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica

General Info for This Class

- Automated Verification of Cyber-Physical Systems is an elective course for the Master Degree in Computer Science
- Lecturer: Igor Melatti
- Where to find these slides and more:
  - `https://igormelatti.github.io/aut_ver_cps/20242025/index_eng.html`
  - also on MS Teams: "DT0759: Automated Verification of Cyber-Physical Systems (2024/25)", code **ramh3r4**
- 2 classes every week, 2 hours per class

Rules for Exams

- The exam consists in either reviewing a research paper or working on a project
- Each student may choose one between the two options
- Project: perform verification of a given cyber-physical system
  - also in small teams (max 3 students)
  - each team may choose one among the ones selected by lecturer
  - or may propose one (but wait for lecturer approval!)
  - each team will have to discuss its project with slides
- Paper: read a conference or journal paper and present it with slides
  - each student may choose one among the ones selected by lecturer
  - or may propose one (but wait for lecturer approval!)

Model Checking Problem

- Input: a system $\mathcal{S}$ and (at least) a property φ
  - more precisely, a *model* of $\mathcal{S}$ must be provided
  - that is, $\mathcal{S}$ must be described in some suitable language
- Output:

  PASS $\mathcal{S}$ satisfies φ, i.e., $\mathcal{S} \models \varphi$
  - the system $\mathcal{S}$ is correct w.r.t. the property φ
  - mathematical certification, much better than, e.g., testing

  FAIL $\mathcal{S}$ does not satisfy φ, i.e., $\mathcal{S} \not\models \varphi$
  - the system $\mathcal{S}$ is buggy w.r.t. the property φ
  - a *counterexample* providing evidence of the error is also returned

Model Checking vs. Other Verification Techniques

- Model checking is fully automatic
  - a model checker only needs the description of $\mathcal{S}$ and the property φ
  - "press button and go"
  - this is not true for other verification tools such as proof checkers, which require human intervention in the process
- Model checking is correct for both PASS and FAIL
  - unless the description of $\mathcal{S}$, or the property φ, are wrong
  - this is not true for other verification techniques such as testing, which only guarantees the FAIL result
  - a buggy system may pass all tests, because the error is in some *corner case*

Model Checking Shortcomings

- Only works for finite-state systems
  - typical example: you may verify a system with 3, 4 or 5 processes, but not with n processes, for a generic n
- Requires skilled personnel to write descriptions (and properties)
  - must know both the model checker language and the system
  - however, less skilled than a proof checker user
  - very few exceptions in which the model is automatically extracted from the system
  - also direct translations from digital circuits to NuSMV are available
- Very resource demanding
  - besides PASS and FAIL, also OutOfMem and OutOfTime are expected results...
  - bounded model checking: PASS is limited to execution up to a given number of steps

Model Checking Algorithms

Two main categories:

Explicit visit the graph induced by the description of $\mathcal{S}$

- very good for invariants and LTL model checking of communication protocols
- on-the-fly generation of the graph: only the reachable states are stored, the adjacency matrix is implicitly given by the description of $\mathcal{S}$
- Murphi, SPIN

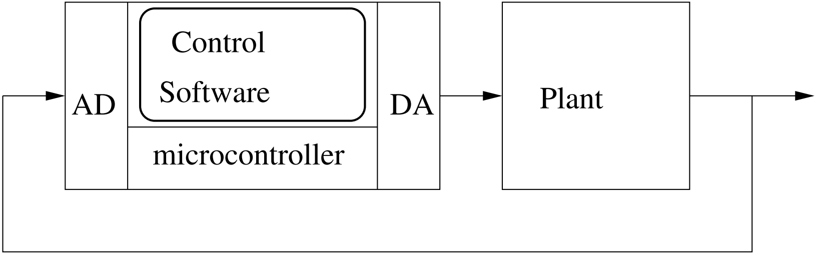
Symbolic represent sets of states and transition relations as OBDDs

- very good for LTL and CTL model checking of hardware-like systems
- all translated into a boolean formula
- also SAT tools may be used (bounded model checking)

Cyber-Physical Systems

- A Cyber-Physical System (CPS) is a system where a physical system is controlled and/or monitored by a software
- They are either partially or fully autonomous
  - we will mainly deal with fully autonomous CPSs
- Examples are everywhere:
  - Internet of Things devices
  - Unmanned Autonomous Vehicles
  - Drones
  - Medical Devices
  - Embedded Systems
  - ...

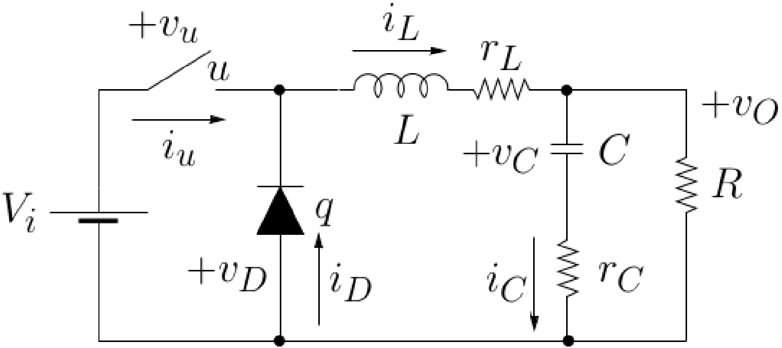
Cyber-Physical Systems with Controllers



CPSs with Controllers: Classical Examples

Buck DC/DC Converter

Buck DC/DC Converter



CPSs with Controllers: Classical Examples

Continuous time dynamics

$$
\begin{aligned}
\dot{i_L} &= a_{1,1} i_L + a_{1,2} v_O + a_{1,3} v_D && (1)\\
\dot{v_O} &= a_{2,1} i_L + a_{2,2} v_O + a_{2,3} v_D && (2)
\end{aligned}
$$

$$
\begin{array}{cccl@{\qquad}cccl}
q & \to & v_D = R_{\mathrm{on}} i_D & (3) & \bar{q} & \to & v_D = R_{\mathrm{off}} i_D & (7)\\
q & \to & i_D \geq 0 & (4) & \bar{q} & \to & v_D \leq 0 & (8)\\
u & \to & v_u = R_{\mathrm{on}} i_u & (5) & \bar{u} & \to & v_u = R_{\mathrm{off}} i_u & (9)\\
v_D & = & v_u - V_{in} & (6) & i_D & = & i_L - i_u & (10)
\end{array}
$$

where:

- i_L, v_O are state variables
- $u \in \{0, 1\}$ is the action

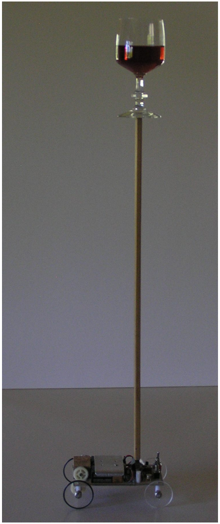
Discrete time dynamics with sampling time T

$$
\begin{aligned}
{i_L}' &= (1 + Ta_{1,1})i_L + Ta_{1,2}v_O + Ta_{1,3}v_D && (11)\\
{v_O}' &= Ta_{2,1}i_L + (1 + Ta_{2,2})v_O + Ta_{2,3}v_D. && (12)
\end{aligned}
$$

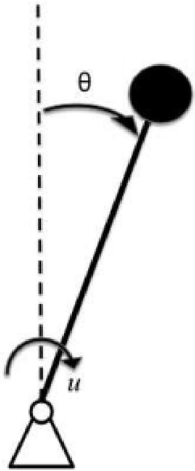
$$
\begin{aligned}
q &\rightarrow v_D = R_{\mathrm{on}} i_D && (13) & \qquad \bar{q} &\rightarrow v_D = R_{\mathrm{off}} i_D && (17)\\
q &\rightarrow i_D \geq 0 && (14) & \bar{q} &\rightarrow v_D \leq 0 && (18)\\
u &\rightarrow v_u = R_{\mathrm{on}} i_u && (15) & \bar{u} &\rightarrow v_u = R_{\mathrm{off}} i_u && (19)\\
v_D &= v_u - V_{in} && (16) & i_D &= i_L - i_u && (20)
\end{aligned}
$$

- Goal: keep v_O in a desired safe interval
  - typically, $5 - 0.01V \leq v_O \geq 5 + 0.01V$
- Notwithstanding the input voltage V_i and the resistance R may vary in some given interval
  - typically, $R = 5 \pm 25\%\Omega$, $V_i = 15 \pm 25\% V$
- Effectively used in laptops: from battery voltage (V_i) to laptop processor voltage (v_O)

Inverted Pendulum



Inverted Pendulum



CPSs with Controllers: Classical Examples

Continuous time dynamics

$$\ddot{\theta} = \frac{g}{l} \sin\theta + \frac{1}{ml^2} Fu$$

where:

- θ is the state variable
- $u \in \{0, 1\}$ is the action
- m, l, F are system parameters

Continuous time dynamics

$$\dot{x}_1 = x_2 \tag{21}$$

$$\dot{x}_2 = \frac{g}{l}\sin x_1 + \frac{1}{ml^2}Fu \tag{22}$$

Discrete time dynamics with sampling time T

$$x_1' = x_1 + Tx_2 \tag{23}$$

$$x_2' = x_2 + T\frac{g}{l}\sin x_1 + T\frac{1}{ml^2}Fu \tag{24}$$

In This Course

To deal with cyber-physical systems:

- Probabilistic Model Checking
  - rather than "are there errors?", it is "is the error probability low enough?"
  - which entails "what is the error probability?"
  - the system is probabilistic, i.e., a Markov Chain
- Statistical Model Checking
  - rather than "are there errors?", it is "is the error probability low enough?"
  - which entails "what is the error probability?"
  - the system may be a non-probabilistic simulator
  - the answer is given with some statistical confidence
  - bridge between testing and verification

In This Course

To deal with cyber-physical systems:

- System Level Formal Verification
  - directly use a simulator instead of describing the system within the model checker
  - this will also need some background on systems simulation
  - bridge between testing and verification
- Automatic Synthesis of Controllers
  - rather than "are there errors in this system?", it is "generate a controller so that errors are avoided"