EL2450 Hybrid and Embedded Control

Lecture 13: Hybrid systems bisimulations

- Definition of timed, multi-rate, and rectangular automata
- Over-approximations of reachable sets
- Model-checking of transition systems over LTL specifications

Today's Goal

You should be able to

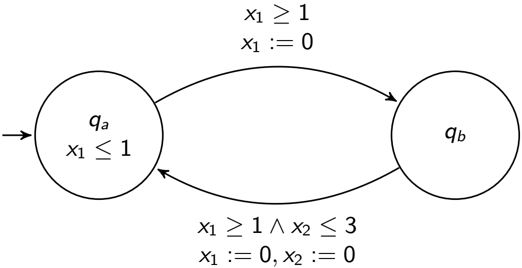
- model a timed automaton
- translate a multi-rate and rectangular automaton to a timed automaton
- understand the concept of reachability analysis for hybrid systems through reach set over-approximation
- understand the model checking process for FTS under LTL specifications

Timed Automata

Timed automata are a subclass of hybrid automata
$TA = (Q, X, \mathrm{Init}, f, D, E, G, R)$

- $Q = \{q_1, \dots, q_m\}$, $X = \mathbb{R}^n_+$, Init $\subseteq Q \times \{0\}^n$
- $f(q, x) = (1, \dots, 1)$: "clock" dynamics
- $E \subseteq Q \times Q$
- $D(q), G(e)$ are *rectangular* sets, i.e., they are finite boolean combinations of constraints of the form $x_i \bowtie a_i$, where $\bowtie \in \{<, \leq, =, \geq, >\}$, and a_i is a positive integer.
- $R(e, x) = \{y\}$, where $y_i = 0$ or $y_i = x_i$ for all $1 \leq i \leq n$.

Example: Timed Automaton

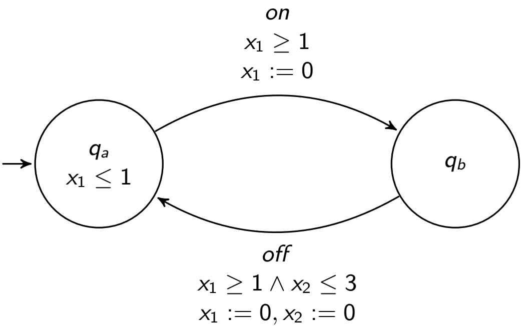


Labeled Timed Automata

Labeled timed automata are a subclass of hybrid automata $TA = (Q, X, \mathrm{Init}, f, Act, D, E, G, R)$

- $Q = \{q_1, \ldots, q_m\}$, $X = \mathbb{R}^n_+$, Init $\subseteq Q \times \{0\}^n$
- $f(q, x) = (1, \ldots, 1)$: "clock" dynamics
- Act **is the set of events**
- $E \subseteq Q \times Act \times Q$
- $D(q), G(e)$ are *rectangular* sets, i.e., they are finite boolean combinations of constraints of the form $x_i \bowtie a_i$, where $\bowtie \in \{<, \leq, =, \geq, >\}$, and a_i is a positive integer.
- $R(e, x) = \{y\}$, where $y_i = 0$ or $y_i = x_i$ for all $1 \leq i \leq n$.

Example: Labeled Timed Automaton



Timed Automata as Transition Systems

A timed automaton $TA = (Q, X, \mathrm{Init}, f, Act, D, E, G, R)$ can be interpreted a transition system $T_{TA} = (S, \Sigma, \rightarrow, S_0 = \mathrm{Init})$:

- $S = Q \times X$ and $(q, x) \in S$ denotes the state
- $\Sigma = Act \cup \mathrm{Time}$ where the generators Act are the event names and Time the continuous evolution
- $(q, x) \xrightarrow{\sigma} (q', y)$ for $\sigma \in Act$ if
  - there exists $(q, \sigma, q') \in E$, and
  - x satisfies the guard $G(q, \sigma, q')$, $\{y\} = R((q, \sigma, q'), x)$, and y satisfies the domain $D(q')$.
- $(q, x) \xrightarrow{\mathrm{Time}} (q, x')$ if $x' = x + \mathrm{Time}$, and x' satisfies $D(q)$.

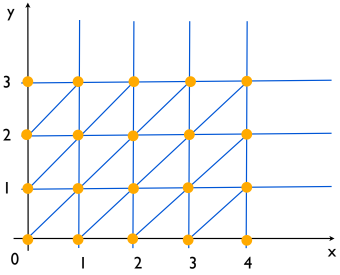
Region Equivalence for One Variable

Let $\equiv$ be an equivalence, where $x \equiv x'$ iff

- $x > M \wedge x' > M$, where M is the largest value associated with x that appears in TA, or
- $x \leq M_i, x' \leq M$, $\lfloor x \rfloor = \lfloor x' \rfloor$, and $frac(x) = 0 \Leftrightarrow frac(x') = 0$

$[x]_{\equiv} = \{x' \mid x \equiv x'\}$

Region Equivalence for Two Variables

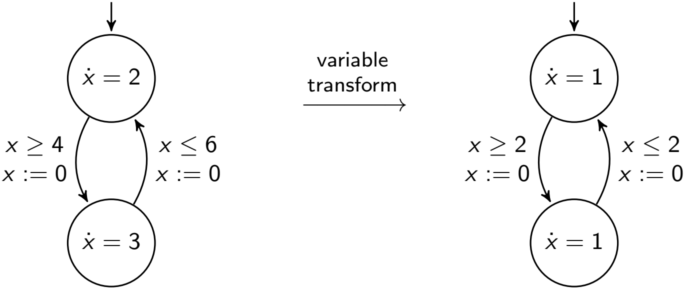


- Equivalence classes: points (orange circles), line segments (blue lines), open sets (between lines)
- $M_x = 4$, $M_y = 3$

Reachability for Timed Automata

- Determine equivalent regions
- Find the quotient transition system of T_{TA}, as $\hat{T}_{TA}$
- Compute reachable set of $\hat{T}_{TA}$
- Same reachability for T_{TA}

Multi-Rate Automata

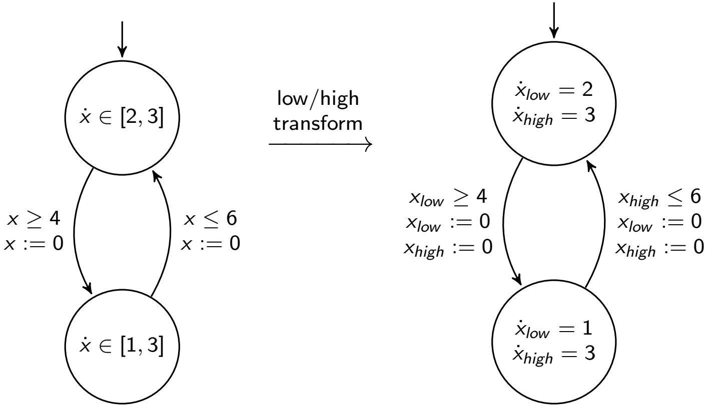


Reachability for Multi-Rate Automata

- Translate into timed automata
- Multi-rate automaton has to be **initialized**, i.e., if a rate of a variable changes along an edge, then the variable has to be reset along the edge

Example: Remove $x := 0$ from the previous example

Rectangular Automata



Reachability for Rectangular Automata

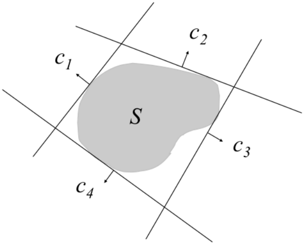
- Translate into multi-rate and then into timed automata
- Rectangular automaton has to be **initialized**, i.e., if a rate of a variable changes along an edge, then the variable has to be reset along the edge

Example: Remove $x := 0$ from the previous example

Over-Approximation of Reach Set

- It is often hard to calculate Reach exactly
- Compute an over-approximation $A \supset \text{Reach}$ instead
- Note that $A \cap B = \emptyset$ implies that $\text{Reach} \cap B = \emptyset$, so safety is guaranteed if the algorithm based on over-approximation terminates

Approximate a Set with a Polyhedron



- Choose normal vectors $c_1, \ldots, c_n$
- Wrap the set in the polyhedron
- Solve an optimization problem

Figure

by R. Alur

Flowpipe Approximation

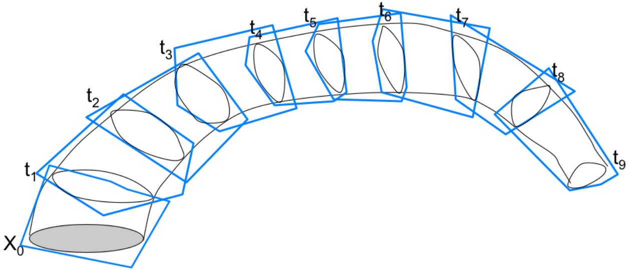


Figure by R. Alur

- Divide Reach(S_0) into $[t_k, t_k + 1]$ segments
- Enclose each segment with a convex polytope
- Reach(S_0) is a union of polytopes

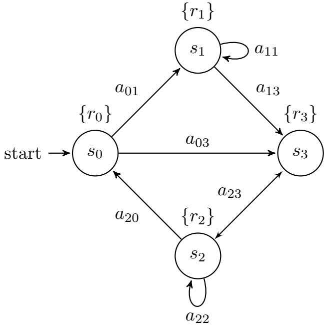
Beyond Reachability Verification

- More complex properties can be expressed and verified,
- Temporal logics, such as LTL, CTL.
- Verification or synthesis of a system and its specification

Model-checking: system model

System modeled as *labeled* finite transition systems (FTS) $\mathcal{T} = (S, \Sigma, \Pi, \rightarrow, L, S_0)$:

- S is a set of states;
- Σ is a set of admissible actions;
- Π is a set of atomic propositions;
- $\rightarrow \in S \times \Sigma \times S$ is a set of transitions;
- $S_0 \subseteq S$ is the set of initial states;
- $L : S \rightarrow 2^{\Pi}$ s.t. $L(s) \subseteq \Pi$ is the set of propositions satisfied by $s \in S$.



Note: the FTS would be weighted by adding costs on the transitions.

Model-checking: specification

Specification given as linear temporal logic (LTL) formulas over Π:

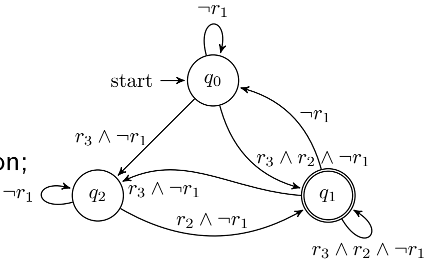
$$\varphi ::= \pi \;|\varphi_1 \wedge \varphi_2 \mid \neg\varphi_1 \mid \mathsf{X}\,\varphi_1 \mid \varphi_1\,\mathsf{U}\,\varphi_2 \mid \mathsf{F}\,\varphi_1 \mid \mathsf{G}\,\varphi_1,$$

see Lecture 9 for details.

Control tasks: Safety: $\mathsf{G}\neg\,\varphi_1$. Order: $\mathsf{F}(\varphi_1 \wedge \mathsf{F}(\varphi_2 \wedge \mathsf{F}\varphi_3))$. Response: $\varphi_1 \to \varphi_2$. Liveness: $\mathsf{GF}\varphi_1$.

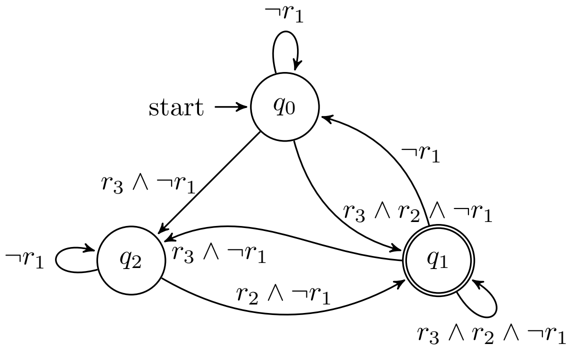
Büchi automaton $\mathcal{A}_\varphi = (Q,\, 2^\Pi,\, \delta,\, Q_0,\, F)$

- Q is a finite set of states;
- 2^Π is an input alphabet;
- $\delta : Q \times 2^\Pi \to 2^Q$ is a transition relation;
- $Q_0,\, F \subseteq Q$ are initial and accepting states.



Model-checking: Büchi Automaton example

Example: $\varphi = (\mathsf{G}\,\mathsf{F}\, r_2) \wedge (\mathsf{G}\,\mathsf{F}\, r_3) \wedge (\mathsf{G}\,\neg\, r_1)$.

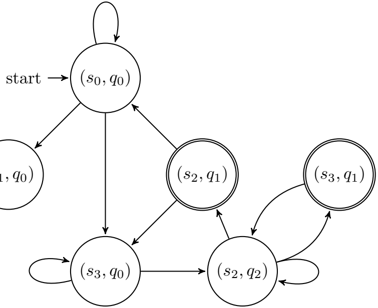


Software: LTL2BA, LTL2dstar, SPOT...

Model-checking: product automaton

The **product** automaton $\mathcal{P} = \mathcal{T} \otimes \mathcal{A}_\varphi = (Q_\mathcal{P},\, \Sigma,\, \delta_\mathcal{P},\, Q_{\mathcal{P},0},\, F_\mathcal{P})$,

- $Q_\mathcal{P} = S \times Q$;
- $\delta_\mathcal{P} \subseteq Q_\mathcal{P} \times \Sigma \times Q_\mathcal{P}$.
  $((s,\, q),\, \sigma,\, (s',\, q')) \in \delta_\mathcal{P}$
  if $(s, \sigma,\, s') \in\longrightarrow$
  and $q' \in \delta(q,\, L(s))$;
- $Q_{\mathcal{P},0} = \{(s_0,\, q_0) \mid q_0 \in Q_0\}$;
- $F_\mathcal{P} = \{(s,\, q_f) \mid s \in S,\, q_f \in F\}$.

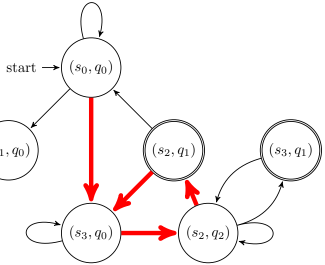


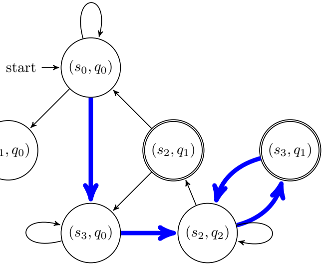
Idea: intersection between language generated by $\mathcal{T}$ and the one accepted by $\mathcal{A}_\varphi$.

Model-checking: graph-search

Graph search for plan prefix and suffix:

- Plan prefix: a path from an initial state to an accepting state;
- Plan suffix: a cycle containing at least an accepting state;
- Combine the plan prefix and suffix (which can be done in different ways).





Next Lecture

Summary

- Summary of the course
- What is in the exam?