1 (a) An example is $H = (Q, X, \text{Init}, f, \text{Dom}, R)$
with $Q = \{1, 2, 3, 4\}$, $X = \mathbb{R}^2$

$\begin{align*}
1 & \xrightarrow{x_2} 4 \\
2 & \xrightarrow{x_1} f(1, \cdot) = \begin{bmatrix} 1 \\ -3 \end{bmatrix} \\
3 & \xrightarrow{x_1} f(2, \cdot) = \begin{bmatrix} 3 \\ 1 \end{bmatrix} \\
4 & \xrightarrow{x_1} f(3, \cdot) = \begin{bmatrix} -1 \\ 3 \end{bmatrix} \\
5 & \xrightarrow{x_1} f(4, \cdot) = \begin{bmatrix} -3 \\ -1 \end{bmatrix}
\end{align*}$

$\text{Init}$ (away from: $x_1 = 0$ or $x_2 = 0$)

$\text{Dom} = (1, \{x : x_1 > 0, x_2 > 0\}) \cup \langle 2, \{x : x_1 < 0, x_2 > 0\} \rangle \cup \langle 3, \{x : x_1 < 0, x_2 < 0\} \rangle \cup \langle 4, \{x : x_1 > 0, x_2 < 0\} \rangle$

$\begin{align*}
R(1, \{x : x_2 \leq 0\}) &= (4, \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0^-) \\
R(2, \{x : x_1 > 0\}) &= (1, \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0^+) \\
R(3, \{x : x_2 \geq 0\}) &= (2, \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0^+) \\
R(4, \{x : x_1 \leq 0\}) &= (3, \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0^-) \\
R &= \emptyset \text{ otherwise}
\end{align*}$

Since $R(q, x) \neq \emptyset$ for all $(q, x) \in \text{Trans}$, $H$ is non-blocking. Since $|R(q, x)| \leq 1$ for all $(q, x)$, $H$ is deterministic.
1. (b) First, note that for every execution, there exists \( i \) such that \( x(t_i) = (a, b)^T \) where either \( a = 0 \) or \( b = 0 \). Then, \( x(t_i') = (c, d)^T \) where \( (c, d) = \left( \frac{b}{b}, 0 \right) \) if \( a = 0 \) and \( (0, -\frac{a}{3}) \) if \( b = 0 \).

It thus follows that \( H \) accepts a unique infinite execution for every initial state. The execution is zero because for all \( t \in \mathbb{R} \)

\[
W(t) = \frac{d}{dt} (\|x_1(t)\| + \|x_2(t)\|) = -2
\]

where \( W(t) = \|x_1(t)\| + \|x_2(t)\| \) (the \( l_1 \)-norm).

So \( W(t) = 0 \) for some finite \( t \) and thus, the origin is reached in finite time.

2. (a) An example is \( H = (Q, X, \text{Init}, f, \text{Dom}, R) \) where \( Q = \{q_1, q_2\} \), \( X = \mathbb{R}^3 \)

\[
f(q, \cdot) = [0, 0, 0]^T
\]

\[
\text{Init} = (q_1, (1, 0, 0)^T)
\]

\[
\text{Dom} = (q_1, (0, 0, 0)^T)
\]

and

\[
R(q, (x_1, x_2, x_3)) = \begin{cases} 
(q, ((x_1 + x_2)/2, (x_1 + x_2)/2, x_3)) & \text{if } x_1 > x_2 \\
(q, (x_1, (x_2 + x_3)/2, (x_2 + x_3)/2)) & \text{if } x_2 > x_3 \\
\emptyset & \text{otherwise}
\end{cases}
\]
2 (b). Define \( \Delta_i = x_1 (\tau_i) - x_2 (\tau_i) + x_2 (\tau_i) - x_3 (\tau_i) \) for all \( i \) with \( \mathbf{x} = (\tau, q, x^i) \in E_{H^+} (q, (100)^T) \).

Then, \( \Delta_i = 2^{-i} > 0 \) so \( \mathbf{x} \) is an infinite execution. It is Zeno because \( \tau_i = \tau_0 \) for all \( i \).

Since \( x_1 (\tau_i) > x_2 (\tau_i) \) and \( x_2 (\tau_i) > x_3 (\tau_i) \), it follows from \( \lim_{i \to \infty} \Delta_i = 0 \) that

\[
\lim_{i \to \infty} x_1 (\tau_i) = \lim_{i \to \infty} x_2 (\tau_i) = \lim_{i \to \infty} x_3 (\tau_i)
\]

But \( x_1 (\tau_i) + x_2 (\tau_i) + x_3 (\tau_i) = 1 \), so the Zeno state of \( \mathbf{x} \) is thus equal to \( (\frac{1}{3}, \frac{1}{3}, \frac{1}{3}) \).
**Problem 3: Steam Boiler** (Steam Boiler System, Figure 1)

The steam boiler consists of a tank containing water and a heating element that causes the water to boil and escape as steam. The water is replenished by two pumps which at time $t$ pump water into the boiler at rates $u_1(t)$ and $u_2(t)$ respectively. At every time $t$, pump $i$ can either be on ($u_i(t) = P_i$) or off ($u_i(t) = 0$). There is a delay $T_i$ between the time pump $i$ is ordered to switch on and the time $q_i$ switches to $P_i$. There is no delay when the pumps are switched off. We will use three hybrid automata to describe this system, one for the boiler, $B = (Q_B, X_B, V_B, Y_B, Init_B, f_B, h_B, Inv_B, E_B, G_B, R_B)$, and one for each of the pumps, $P_i = (Q_i, X_i, V_i, Y_i, Init_i, f_i, h_i, Inv_i, E_i, G_i, R_i)$.

The boiler automaton is defined by:

- $Q_B = \{q_B\}$, $Q_B = \{BOILING\}$;
- $X_B = \{w, r\}$, $X_B = \mathbb{R}^2$;
- $V_B = \{u_1, u_2, d\}$, $V_B = [0, P_1] \times [0, P_2] \times [-D_1, D_2]$, where $P_1, P_2, D_1, D_2 > 0$;
- $Y_B = \{y_1, y_2\}$, $Y_B = \mathbb{R}^2$;
- $Init = \{BOILING\} \times [0, W] \times [0, R]$, where $W, R > 0$;
- $f(BOILING, w, r, u_1, u_2, d) = \begin{bmatrix} \dot{w} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} u_1 + u_2 - r \\ d \end{bmatrix}$
- $h(BOILING, w, r) = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} w \\ r \end{bmatrix}$
- $Inv(BOILING) = X_B \times V_B = \text{Dom}$
- $E = \emptyset$
Notice that since \( E = \emptyset \), \( G \) and \( R \) are trivial and need not be explicitly defined. The automaton for pump \( i \) is defined by:

- \( Q_i = \{ q_i \}, Q_i = \{ OFF, GOING\_ON, ON \} \)
- \( X_i = \{ T_i \}, X_i = R \)
- \( V_i = \{ c_i \}, V_i = \{ 0, 1 \} \)
- \( Y_i = \{ w_i \}, Y_i = [0, P_i] \)
- \( \text{Init} = \{ OFF \} \times \{ 0 \} \)

\[
f(q_i, T_i, c_i) = \begin{cases} 
1 & \text{if } q_i = GOING\_ON \lor q_i = ON \\
0 & \text{if } q_i = OFF 
\end{cases}
\]

\[
h(q_i, T_i) = \begin{cases} 
0 & \text{if } q_i = OFF \lor q_i = GOING\_ON \\
P_i & \text{if } q_i = ON 
\end{cases}
\]

- \( \text{Dom}: \text{Inv}(q_i) = \begin{cases} 
X_i \times \{ c_i = 0 \} & \text{if } q_i = OFF \\
\{ T_i \leq T_i \} \times \{ c_i = 1 \} & \text{if } q_i = GOING\_ON \\
X_i \times \{ c_i = 1 \} & \text{if } q_i = ON 
\end{cases} \)

- \( E = \{(OFF, GOING\_ON), (GOING\_ON, OFF), (GOING\_ON, ON), (ON, OFF)\} \)

- \( G(e) = \begin{cases} 
X_i \times \{ c_i = 1 \} & \text{if } e = (OFF, GOING\_ON) \\
X_i \times \{ c_i = 0 \} & \text{if } e = (GOING\_ON, OFF) \\
\{ T_i \geq T_i \} \times \{ c_i = 1 \} & \text{if } e = (GOING\_ON, ON) \\
X_i \times \{ c_i = 0 \} & \text{if } e = (ON, OFF) 
\end{cases} \)

- \( R(e, T_i, c_i) = \begin{cases} 
\{ 0 \} & \text{if } e = (OFF, GOING\_ON) \\
\{ 0 \} & \text{if } e = (GOING\_ON, OFF) \\
\{ T_i \} & \text{if } e = (GOING\_ON, ON) \\
\{ 0 \} & \text{if } e = (ON, OFF) 
\end{cases} \)
The controller synthesis for this model can be found in [4]. The continuous dynamics of the boiling process are summarized by the differential equations:

\[
\begin{align*}
\dot{\psi} &= p_1 + p_2 - r \\
\dot{r} &= d 
\end{align*}
\]

The dynamics of pump \( i \) are summarized by the open hybrid automaton of Figure 2. Notice that \( p_i \) is both an output variable of the pump and an input variable of the boiling process. For a formal definition of the model (with slight differences in the notation) please refer to Lecture 8.

The composite automaton has 4 continuous, real valued state variables:

\[ z = (u, r, T_1, T_2) \in \mathbb{R}^4 \]

**Solution FROM**

"Controllers for Reachability Specifications for Hybrid Systems" by Lygeros, Tomlin, Sastry

*Automatica 1999.*

**Figure 1:** The Steam Boiler

**Figure 2:** The pump hybrid automaton

9 discrete states:

\[ q \in \{(OFF, OFF), (OFF, GOING.ON), \ldots, (ON, ON)\} \]

2 discrete input variables:

\[ (u_1, u_2) \in \{0, 1\} \times \{0, 1\} = U \]

and one continuous input variable:

\[ d \in [-D_1, D_2] = D \]

As the notation suggests, \( u_1 \) and \( u_2 \) will play the role of controls and \( d \) will play the role of the disturbance. The additional requirement that \( r \in [0, R] \) can be encoded by a state dependent input constraint:

\[ \phi(q, r) = \begin{cases} 
U \times [0, D_2] & \text{if } r \leq 0 \\
U \times D & \text{if } r \in (0, R) \\
U \times [-D_1, 0] & \text{if } r \geq R 
\end{cases} \]

**Proposition 3** If \( \text{Init} \subseteq Q \times \mathbb{R} \times [0, R] \times \mathbb{R}^2 \), then for all \( x^\tau = (r, q, x, u_1, u_2, d) \) and for all \( t \in \tau, \tau(t) \in [0, R] \).

Our goal is to design a controller that keeps the water level in a given range, \([M_1, M_2]\), with \( 0 \leq M_1 < M_2 \). This requirement can easily be encoded by a safety property \((Q U X, \Box F)\) with

\[ F = Q \times [M_1, M_2] \times \mathbb{R}^3 \]
We will try to achieve this goal by treating the situation as a game between \((u_1, u_2)\) and \(d\) over the cost function \(J\). Recall that this involves solving the equation:

\[ J^*(q_0, x_0) = \max_\delta \min_d \left( \min_{x=(r,q,x,d)\in H_d} J(x) \right) \]

Fortunately, for this example, the equation simplifies considerably.

First, notice that the steam boiler system is deterministic, in the sense that for each initial state and each input sequence consistent with \(\phi\) the automaton accepts a unique execution. In this case, we can represent an execution more compactly by \(((q_0, x_0), (u_1, u_2), d)\) with the interpretation that \((u_1, u_2)\) and \(d\) represent the entire sequence for these variables. Moreover, if the memoryless controller we pick is single valued, this implies we need not worry about the innermost minimization.

Next notice that \(J\) can be encoded by means of two real valued cost functions:

\[ J_1(x^0, u_1, u_2, d) = \inf_{t \geq 0} w(t) \quad \text{and} \quad J_2(x^0, u_1, u_2, d) = -\sup_{t \geq 0} w(t) \]

Clearly:

\[ J = 1 \iff (J_1 \geq M_1) \land (J_2 \geq -M_2) \]

The problem of finding a solution to the game over the discrete cost function (known as qualitative game or game of kind) reduces to finding solutions to two real valued games (known as quantitative games or games of degree). Even though there is no obvious benefit to doing this, it allows us to use tools from continuous optimal control to address the problem.

Start with the game over \(J_1\). Guess a possible solution:

\[ u^*_i(q, x) = 1 \text{ for all } (q, x), \quad \text{and} \quad \ D^*(q, x) = \begin{cases} D_2 & \text{if } r < R \\ 0 & \text{if } r = R \end{cases} \]

Notice that both players resort to a feedback strategy (a trivial one).

**Lemma 1** \((u_1^*, u_2^*, d^*)\) is globally a saddle solution for the game between \((u_1, u_2)\) and \(d\) over \(J_1\).

**Proof:** See [4].

A saddle solution is a solution to equation (1) for which the order in which the players make their decisions turns out to be unimportant. In other words, a solution for which for all \((q, x)\):

\[ J^*_1(q, x) = \max_{(u_1, u_2)} \min_d J_1((q, x), (u_1, u_2), d) = \min_{(u_1, u_2)} \max_d J_1((q, x), (u_1, u_2), d) \]

Or, in other words, a solution for which for all \((q, x), u_1, u_2\) and \(d\):

\[ J_1((q, x), (u_1, u_2), d^*) \leq J_1((q, x), (u_1^*, u_2^*), d^*) \leq J_1((q, x), (u_1^*, u_2^*), d) \]

The last definition is usually somewhat easier to to work with. It is in fact used to prove the lemma.

The saddle cost:

\[ J^*_1(q, x) = J_1((q, x), (u_1^*, u_2^*), d^*) \]

Can be computed in closed form. This allows us then to compute the set of states for which there exists a control that for all actions of the disturbance prevents draining. This set turns out to be of the form:

\[ W_1^* = \{(q, x) : J^*_1(q, x) \geq M_1\} = \{(q, x) : w \geq \tilde{w}(r, T_1, T_2)\} \]

Two level sets of this function are shown in Figure 3.

The expression for \(J^*\) also allows us to compute the least restrictive controller that renders the set \(W_1^*\) invariant. It turns out to be unique:
Lemma 2 The feedback controller $g_f^1$ given by:

\begin{align*}
    u_1 &\in \{0, 1\} \text{ and } u_2 \in \{0, 1\} \text{ if } [w > \bar{w}(r, 0, 0)] \lor [w < \bar{w}(r, T_1, T_2)] \\
    u_1 &= 1 \text{ and } u_2 \in \{0, 1\} \text{ if } \bar{w}(r, 0, 0) \geq w > \bar{w}(r, T_1, 0) \\
    u_1 &\in \{0, 1\} \text{ and } u_2 = 1 \text{ if } \bar{w}(r, 0, 0) \geq w > \bar{w}(r, 0, T_2) \\
    u_1 &= 1 \text{ and } u_2 = 1 \text{ if } w = \bar{w}(r, T_1, T_2)
\end{align*}

is the unique, least restrictive, non-blocking, feedback controller that renders $W^*_1$ invariant.

Proof: See [4].

Note that the first term applies to states in the interior of the safe set ($w > \bar{w}(r, 0, 0)$) as well as all the states outside the safe set ($w < \bar{w}(r, T_1, T_2)$). The expression for $\bar{w}$ (see [4]) suggests that $\bar{w}$ is monotone in $T_1$ and $T_2$. Therefore, the condition on the last case is enabled if and only if all other conditions fail. The two middle conditions may overlap, however. Therefore there is some nondeterminism in the choice of safe controls (some states may be safe with either one or the other pump on, but not neither).

References


this kind of Zeno behavior. The classical way of analyzing such systems is by introducing the notion of sliding modes [8,18].

4—Regularization—

Solution to Problem 4.

From "On the Regularization of Zeno Hybrid Automata"


Regularization is a standard technique for dealing with differential equations whose solutions are not well defined. We propose a similar approach to extend Zeno executions beyond the Zeno time, primarily for the purpose of simulation. The formal treatment of how to regularize general Zeno hybrid automata is the topic of current research. Here we limit ourselves to specific regularizations of the water tank and bouncing ball automata introduced above. All regularizations are motivated by physical considerations of the underlying systems. For the water tank automaton, it is interesting to notice that different regularizations suggest different extensions of the executions. For the bouncing ball automaton, all extensions considered here are consistent with one another and physical intuition. The regularizations are only presented graphically in this section; see [10] for formal definitions.

Consider a non-blocking and deterministic hybrid automaton $H$ and assume that for every $(q_0, x_0) \in \text{Init}$ the execution $\chi \in \mathcal{H}_{(q_0, x_0)}^{\infty}$ is Zeno. Regularization of $H$ involves constructing a family of deterministic, non-blocking, and non-Zeno automata $H_{\varepsilon}$, parameterized by a real valued parameter, $\varepsilon > 0$, and a continuous map, $\phi : Q_{\varepsilon} \times X_{\varepsilon} \to Q \times X$, relating the state of each $H_{\varepsilon}$ to the state of $H$. Given an execution $\chi_{\varepsilon} = (\tau_{\varepsilon}, q_{\varepsilon}, x_{\varepsilon})$, we use $\phi(\chi_{\varepsilon})$ as a shorthand notation for the collection $(\tau, q, x)$ with $\tau = \tau_{\varepsilon}$, and $(q(t), x(t)) = \phi(q_{\varepsilon}(t), x_{\varepsilon}(t))$ for all $t \in \tau$. Note that in general $\phi(\chi_{\varepsilon})$ will not be an execution of $H$. However, the construction of the family $H_{\varepsilon}$ should be such that $H_{\varepsilon}$ tends to $H$ as $\varepsilon$ tends to 0, in the sense that if $(q_{\varepsilon_{0}}, x_{\varepsilon_{0}}) \in \text{Init}_{\varepsilon}$, then $\phi(q_{\varepsilon_{0}}, x_{\varepsilon_{0}}) \in \text{Init}$, and if $\chi_{\varepsilon}$ is the execution of $H_{\varepsilon}$ with initial condition $(q_{\varepsilon_{0}}, x_{\varepsilon_{0}})$, then $\phi(\chi_{\varepsilon})$ converges to $\chi \in \mathcal{H}_{\phi(q_{\varepsilon_{0}}, x_{\varepsilon_{0}})}^{\infty}$ over all compact subintervals of $[\tau_{0}, \tau_{\infty})$, where the convergence is taken in the Skorohod metric [4].

---

3 Formally, we need to eliminate all "inert" transitions from $\tau$, that is, replace all $[\tau_i, \tau_i'][[\tau_{i+1}, \tau_{i+1}']$ for which $\phi(q_{\varepsilon}(\tau_i'), x_{\varepsilon}(\tau_i')) = \phi(q_{\varepsilon}(\tau_{i+1}), x_{\varepsilon}(\tau_{i+1}))$ by a single interval $[\tau_i, \tau_{i+1}']$.  

11
Water Tank Automaton

We first study temporal and spatial regularizations of the water tank automaton. Throughout, we assume that $\max\{v_1, v_2\} < w < v_1 + v_2$, so that $WT$ is Zeno.

Physically, temporal regularization represents a situation where there is a delay, $\epsilon > 0$, between the time the inflow is commanded to switch from one tank to the other and the time the switch actually takes place. The temporal regularization of the water tank automaton, $WT^T_\epsilon$, is shown in Figure 3. It is easy to show that $WT^T_\epsilon$ accepts a unique non-Zeno execution for each initial state. Overloading the notation somewhat, we can express the relation between the states of $WT^T_\epsilon$ and the states of $WT$ through the map $\phi(q_i, (x_1, x_2, x_3)) = \phi(q'_i, (x_1, x_2, x_3)) = (q_i, (x_1, x_2))$, for $i = 1, 2$. If we set $r_1 = r_2 = 1, v_1 = 2, v_2 = 3$, and $w = 4$, and assume that initially $x_1(0) = x_2(0) = 2$ and $q(0) = q_1$, then $\tau_\infty = 2$. Figure 4 shows simulation results for $WT^T_\epsilon$; $x_1$ and $x_2$ are plotted as functions of time for two values of $\epsilon$, 0.1 and 0.01. Note that as $\epsilon$ decreases, the execution of $WT^T_\epsilon$ converges over the interval $(\tau_0, \tau_\infty) = (0, 2)$ to the execution of $WT$, in the sense discussed above. For $t > \tau_\infty$, the continuous part of the execution of $WT^T_\epsilon$ tends to $\left(x_1(t), x_2(t)\right) = \left(1, 1 - (t - \tau_\infty)\right)$.

The spatial regularization of the water tank automaton corresponds to a situation where the measurement of $x_1$ and $x_2$ is based on floats, which have to move a certain distance $\epsilon$ to register a change. It can be implemented by introducing a minimum deviation in the continuous state variables between the discrete transitions. The regularized automaton, $WT^T_\epsilon$, is presented in Figure 5. Again one can show that $WT^S_\epsilon$ accepts a unique non-Zeno execution for each initial state. We can relate the state of $WT^S_\epsilon$ to the state of $WT$ through $\phi(q_i, (x_1, x_2, x_3, x_4)) = (q_i, (x_1, x_2))$, for $i = 1, 2$. Figure 6 shows simulation results for $WT^S_\epsilon$ with $\epsilon = 0.1$ and 0.01 and the parameters given above. As for the temporal regularization, the execution of $WT^S_\epsilon$ converges to the execution of $WT$ over the interval $(\tau_0, \tau_\infty)$. For $t > \tau_\infty$, however, the execution converges to $x_1(t) = x_2(t) = -(t - \tau_\infty)/2 + 1$, which is different from the limit in the case of temporal regularization.

Bouncing Ball Automaton

Next, we consider temporal and dynamic regularizations of the bouncing ball automaton. Throughout we assume $c > 1$ so that $BB$ is Zeno.
\[ \begin{align*}
&x_2 \leq r_2 \\
&x_3 := 0 \\
&x_3 \geq \epsilon \\
&x_1 = w - v_1 \\
&x_2 = -v_2 \\
&x_3 = 0 \\
&x_2 \geq r_2 \\
&x_3 \geq \epsilon \\
&x_1 = -v_1 \\
&x_2 = w - v_2 \\
&x_3 = 0 \\
&x_1 \leq r_1 \\
&x_3 := 0
\end{align*} \]

Fig. 3. Temporal regularization of the water tank automaton.

![Graph showing time delay effects](image)

Fig. 4. Simulation of the temporally regularized water tank automaton.
Fig. 5. Spatial regularization of the water tank automaton.

\[
\begin{align*}
\dot{x}_1 &= w - v_1 \\
\dot{x}_2 &= -v_2 \\
\dot{x}_3 &= \dot{x}_4 = 0 \\
x_2 &\geq x_4 - \epsilon \\
x_4 &= x_2 \\
\dot{x}_1 &= -v_1 \\
\dot{x}_2 &= w - v_2 \\
\dot{x}_3 &= \dot{x}_4 = 0 \\
x_1 &\geq x_3 - \epsilon \\
x_1 \leq x_3 - \epsilon \\
x_3 &= x_1
\end{align*}
\]

Fig. 6. Simulation of the spatially regularized water tank automaton.