AA278A Lecture Notes 4
Spring 2005
Existence of Executions

Claire J. Tomlin

April 7, 2005

The lecture notes for this course are based on the first draft of a research monograph: *Hybrid Systems*. The monograph is copyright the authors:
©John Lygeros, Shankar Sastry, Claire Tomlin
and these lecture notes must not be reproduced without consent of the authors.

1 Reachable States

Reachability is a crucial concept in the study of hybrid systems. Roughly speaking, a state, $(q, x) \in Q \times X$ is called reachable if the hybrid automaton H can "steer" its way to (q, x) while moving along one of its executions. The importance of the concept of reachability is difficult to overstate. In the next section we will show how reachability plays a central role in the derivation of existence and uniqueness conditions for executions. In subsequent sections, reachability will also turn out to be a key concept in the study of safety properties for hybrid systems. In this section we provide some basic definitions and results to motivate subsequent discussion.

Definition 1 (Reachable State) *A state $(q', x') \in Q \times X$ of an autonomous hybrid automaton H is called reachable if there exists a finite execution (τ, q, x) ending in (q', x'), i.e. $\tau = \{[\tau_i, \tau_i']\}_0^N$ with $N < \infty$ and $(q_N(\tau_N'), x_N(\tau_N')) = (q', x')$.*

We use $\text{Reach} \subseteq Q \times X$ to denote the set of all states reachable by H. Clearly, $\text{Init} \subseteq \text{Reach}$ (since we may choose $N = 0$ and $\tau_0' = \tau_0$).

Computing the set of reachable states is in general very difficult. Much of the rest of this course will be devoted to different methods that have been proposed for performing this computation.

Definition 2 (Invariant Set) *Consider an autonomous hybrid automaton H. A set of states $M \subseteq Q \times X$ is called invariant if for all $(q_0, x_0) \in M$, and all (τ, q, x) starting from (q_0, x_0), $(q_i(t), x_i(t)) \in M$ for all $i \leq \langle\tau\rangle$ and $t \in I_i$.*

One requirement imposed on hybrid automata is that the state never leave the domain $Q \times \text{Dom}$. This is often the case when modeling physical systems, where the domain typically encodes hard constraints on the state that should be satisfied along all executions. This requirement can be characterized in terms of the reachable states.

Definition 3 (Domain Preserving) *An autonomous hybrid automaton, H, is called domain preserving if* $\text{Reach} \subseteq Q \times \text{Dom}$.

A simple way to determine whether a hybrid automaton is domain preserving is using induction arguments along its executions: if the initial set of states is in the domain, and the reachable states from these initial states is in the domain, the hybrid automaton is domain preserving.

Example (Water Tank (continued)) Consider again the water tank automaton WT, and assume that $w > \max\{v_1, v_2\}$. Notice that $\text{Init}_{WT} \subseteq \text{Dom}_{WT}$. Here, we can show that WT is domain preserving since Init_{WT} is an invariant set.

Consider an arbitrary initial state $(q_0, x_0) \in \text{Init}_{WT}$ and an arbitrary execution (τ, q, x) starting in this initial state. We argue that $(q_i(t), x_i(t)) \in \text{Init}_{WT}$ for all $i \leq \langle\tau\rangle$, and all

$t \in I_i$ by induction. By assumption, $(q_0, x_0) \in \text{Init}_{WT}$. Assume $(q_i(\tau_i), x_i(\tau_i) \in \text{Init}_{WT}$ for some $i \leq \langle\tau\rangle$. If $\tau_i' > \tau_i$, $(q_i(t), x_i(t)) \in \text{Dom}_{WT}$ for all $t \in [\tau_i, \tau_i')$, by the definition of an execution. The only way the state can leave Init_{WT} along continuous evolution is if $x_1 = r_1$ or $x_2 = r_2$. Assume $q_i(\tau_i) = q_1$ (the case $q_i(\tau_i) = q_1$ is symmetric). Then, since $w > \max\{v_1, v_2\}$, $\dot{x}_1 > 0$. Therefore, the only way the state can leave Init_{WT} is if $x_2 = r_2$. But $x_2 = r_2$ implies a discrete transition takes place, since $\text{Dom}(q_1) = \{x \in \mathbb{R}^2 \mid x_2 \geq r_2\}$ and $\text{Dom}(q_2) = \{x \in \mathbb{R}^2 \mid x_1 \geq r_1\}$ and $\dot{x}_2 = -v_2 < 0$. Therefore, $(q_i(t), x_i(t)) \in \text{Init}_{WT}$ for all $t \in [\tau_i, \tau_i']$. Moreover, $R(q_1, x) = \{(q_2, x)\}$, therefore $(q_i(\tau_{i+1}), x_i(\tau_{i+1}) \in \text{Init}_{WT}$. By induction, Init_{WT} is an invariant set.

■

It is easy to see that the thermostat automaton T is not domain preserving, since the initial states are unconstrained, and therefore $\text{Reach}_T = \text{Init}_T = X_T \supset \text{Dom}_T$.

2 Transition States

Another important concept in the study of existence properties is the set of states from which continuous evolution is impossible, sometimes referred to as the *transition states*.

For $(\hat{q}, \hat{x}) \in Q \times X$ and some $\epsilon > 0$, consider the set of solutions, $x(\cdot) : [0, \epsilon) \to X$ to the differential equation:

$$\frac{dx}{dt} = f(\hat{q}, x) \text{ with } x(0) = \hat{x}. \tag{1}$$

To characterize the states from which continuous evolution is impossible, we define the set $\text{Trans} \subseteq Q \times X$ by

$$\text{Trans} = \{(\hat{q}, \hat{x}) \in Q \times X \mid \forall x(\cdot)\ \forall \epsilon > 0\ \exists t \in [0, \epsilon) \text{ such that } x(t) \notin \text{Dom}(\hat{q})\}.$$

In words, Trans is the set of states for which continuous evolution along the differential inclusion forces the system to exit the domain instantaneously.

The exact characterization of the set Trans may be quite involved. We conclude the section by giving some suggestions on how this can be done in certain simple cases. We restrict our attention to cases in which the domain is sufficiently smooth. If f is Lipschitz continuous in x, the following is an immediate consequence of the existence of solutions of ordinary differential equations.

Proposition 4 *If an autonomous hybrid automaton is such that f is locally Lipschitz continuous in x, then* $\text{Trans} \cap \overline{Q \times \text{Dom}} \subseteq \partial(q \times \text{Dom})$.

Here, the notation ∂ refers to the boundaries of the domains in each state q. The simplest case is the one in which the domain is an open set.

Proposition 5 *Consider an autonomous hybrid automaton. If $Q \times \text{Dom}$ is open and f is locally Lipschitz continuous in x, then* $\text{Trans} = (Q \times \text{Dom})^c$.

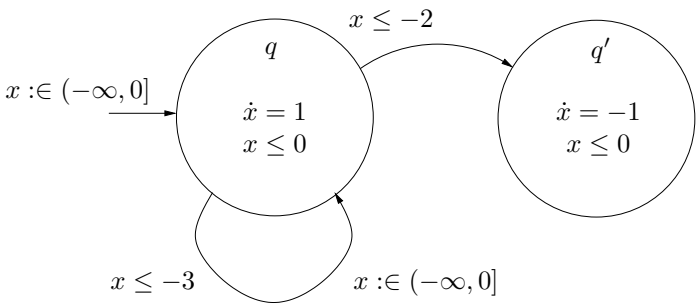

Figure 1: Examples of blocking and non-determinism.

3 Local Existence and Uniqueness

Next, we turn our attention to questions of existence of executions. We derive conditions under which all executions can be extended to infinite executions and conditions under which this extension can be done uniquely. An execution, (τ, q, x), of a hybrid automaton H is called *maximal* if it is not a strict prefix of any other execution of H.

Definition 6 (Non-Blocking and Deterministic) *An autonomous hybrid automaton H is called non-blocking if for all initial states $(q_0, x_0) \in \text{Init}$, there exists an infinite execution starting at (q_0, x_0). It is called deterministic if for all initial states $(q_0, x_0) \in \text{Init}$ there exists at most one maximal execution starting at (q_0, x_0).*

Roughly speaking, the non-blocking property implies that executions exist for all initial states, while the deterministic property implies that the infinite executions (if they exist) are unique. In continuous dynamical systems the existence property is satisfied when the vector field is continuous, while the uniqueness property is satisfied when it is locally Lipschitz continuous. In hybrid systems, however, more things can go wrong. Consider for example the autonomous hybrid automaton of Figure 1. Let (q_0, x_0) denote the initial state, and notice that $q_0 = q$. If $x_0 = -3$, executions starting at (q_0, x_0) can either flow along the vector field, or jump back to q resetting x anywhere in $(-\infty, 0]$, or jump to q' leaving x unchanged. If $x_0 = -2$ executions starting at (q_0, x_0) can either flow along the vector field, or jump to q'. If $x_0 = -1$ executions stating at (q_0, x_0) can only flow along the vector field. Finally, if $x_0 = 0$ there are no executions starting at (q_0, x_0), other than the trivial execution defined over $[\tau_0, \tau_0']$ with $\tau_0 = \tau_0'$. Therefore, the hybrid automaton of Figure 1 accepts no infinite executions for some initial states and multiple infinite executions for others.

Intuitively, a hybrid automaton is non-blocking if for all reachable states for which continuous evolution is impossible a discrete transition is possible. This fact is stated more formally in the following lemma.

Lemma 7 *H is non-blocking if for all $(q, x) \in \text{Trans} \cap \text{Reach}$, there exists $q' \in Q$ such that $(q, q') \in E$ and $x \in G(q, q')$. If H is deterministic, then it is non-blocking if and only if this condition holds.*

In the case in which H is non-deterministic the conditions of Lemma 7 are not necessary. The reason is that the conditions ensure that a blocking execution exists for some initial conditions, but are not sufficient to show that for some initial conditions all executions will block. For example, in the system of Figure 1 some executions starting at $s_0 = (q, 2)$ are cannot be extended to infinite executions (the ones that start by flowing) while others can (the ones that start by a transition to q').

Intuitively, a hybrid automaton may be non-deterministic if either there is a choice between continuous evolution and discrete transition, or if a discrete transition can lead to multiple destinations. More specifically, the following lemma states that a hybrid automaton is deterministic if and only if (1) each discrete transition has a unique destination, and (2) whenever a discrete transition is possible continuous evolution is impossible.

Lemma 8 *An autonomous hybrid automaton H is deterministic if and only if for all $(q, x) \in$ Reach: (1) if $x \in G(q, q')$ for some $(q, q') \in E$, then $(q, x) \in$ Trans; (2) if $(q, q') \in E$ and $(q, q'') \in E$, where $q' \neq q''$, then $x \neq G(q, q') \cap G(q, q'')$; and (3) if $(q, q') \in E$ and $x \in G(q, q')$ then $R(q, x) = (q', x')$ (ie. the set contains a single element).*

Theorem 9 *(Existence and Uniqueness) A hybrid automaton H accepts a unique infinite execution for each initial state if it satisfies all of the conditions of Lemmas 7 and 8.*

Example (Water Tank (continued)) Recall that

$$\begin{aligned}\text{Reach}_{WT} &= \{(q, x) \in \mathbf{Q} \times \mathbf{X} \mid x_1 \geq r_1 \wedge x_2 \geq r_2\},\\ \text{Trans}_{WT} &= \{q_1\} \times \{x \in \mathbf{X} \mid x_2 \leq r_2\} \cup \{q_2\} \times \{x \in \mathbf{X} \mid x_1 \leq r_1\}.\end{aligned}$$

Therefore,

$$\begin{aligned}\text{Reach}_{WT} \cap \text{Trans}_{WT} =& \{q_1\} \times \{x \in \mathbf{X} \mid x_1 \geq r_1 \wedge x_2 = r_2\} \cup\\ & \{q_2\} \times \{x \in \mathbf{X} \mid x_2 \geq r_2 \wedge x_1 = r_1\}.\end{aligned}$$

Notice that $R_{WT}(s) \neq \emptyset$ for all $s \in \text{Reach}_{WT} \cap \text{Trans}_{WT}$, therefore the conditions of Lemma 7 are satisfied. Moreover, for all $s \in \text{Reach}_{WT}$, $R_{WT}(s)$ contains either no elements (if $s \in \text{Trans}_{WT}^c$), or one element (if $s \in \text{Trans}_{WT}$). Therefore, the conditions of Lemma 8 are also satisfied, and the water tank automaton accepts a unique infinite execution for each initial state. ∎

4 Zeno Executions

The conditions of Theorem 9 ensure that a hybrid automaton accepts infinite executions for all initial states. They do not, however, ensure that the automaton accepts executions defined over arbitrarily long time horizons. We know by assumption of Lipschitz continuity that the state cannot escape to infinity in finite time along continuous evolution. However, the infinite executions may be such that the state takes an infinite number of discrete transitions ($\langle\tau\rangle = \infty$) in finite time (i.e. $|\tau| < \infty$). Executions with this property are known as Zeno executions.

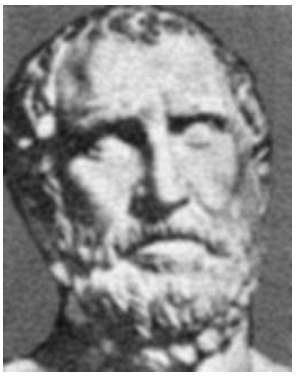

Figure 2: Zeno of Elea

4.1 Examples of Zeno Behavior

The name Zeno executions comes from the ancient Greek philosopher, Zeno of Elea (Figure 2). Born around 490BC, Zeno was a philosopher in the Eleatic school and a student of Parmenides. The teachings of Parmenides rejected the ideas of plurality and change as illusions generated by our senses. The main contribution of Zeno was a series of paradoxes designed to support the view of his mentor by showing that accepting plurality and motion leads to logical contradictions. One of the better known ones is the race of Achilles and the turtle.

> Achilles, a renowned runner, was challenged by the turtle to a race. Being a fair sportsman, Achilles decided to give the turtle a 100 meter head-start. To overtake the turtle, Achilles will have to first cover half the distance separating them, i.e. 50 meters. To cover the remaining 50 meters, he will first have to cover half that distance, i.e. 25 meters, and so on, ad infinitum. Covering each one of the segments in this series requires a non zero amount of time. Since there is an infinite number of segments, Achilles will never overtake the turtle.

This paradox may seem simple minded to the modern reader, but it was not until the beginning of the 20^{th} century that it was resolved satisfactorily by mathematicians and philosophers. And it was not until the end of the 20^{th} century that it turned out to be a practical problem, in the area of hybrid systems.

To motivate the subsequent discussion we list a few examples to illustrate different aspects of the Zeno phenomenon.

Example (Chattering System) Consider the autonomous hybrid automaton of Figure 3. It is easy to show that the hybrid automaton accepts a unique infinite execution for all initial states. However, all infinite executions are Zeno. An execution starting in x_0 at τ_0 reaches $x = 0$ in finite time $\tau_0' = \tau_0 + |x_0|$ and takes an infinite number of transitions from then on, without time progressing further. Thus $\|\tau\| = \tau_0 + |x_0|$. ∎

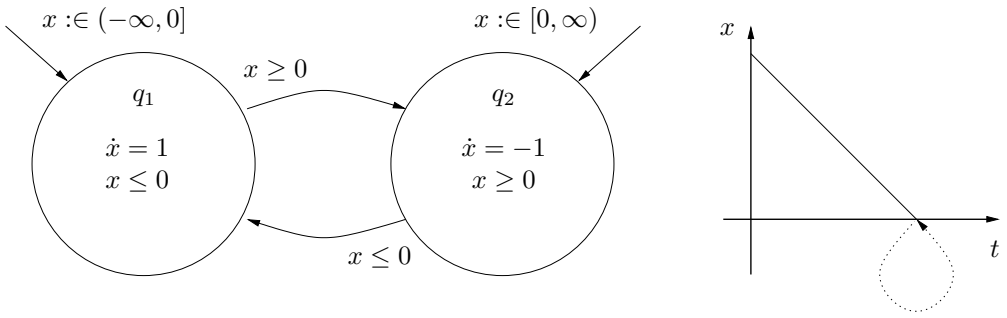

Figure 3: Chattering system.

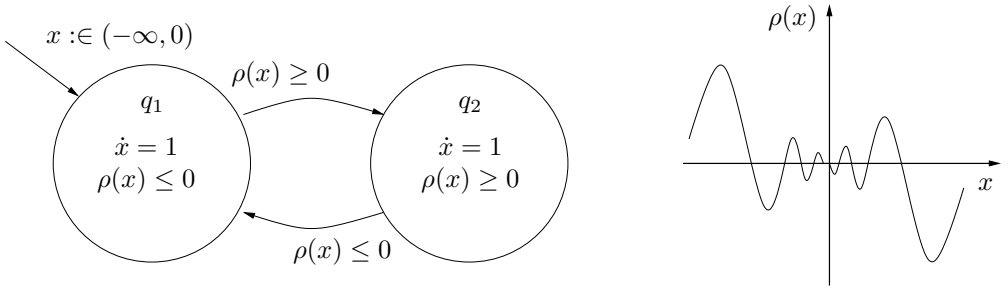

Figure 4: System with a smooth, non-analytic domain.

Example (Non-analytic Domain) Consider the hybrid automaton of Figure 4. Assume that the function $\rho : \mathbb{R} \to \mathbb{R}$ that determines the boundary of the domain is of the form

$$\rho(x) \begin{cases} \sin\left(\frac{1}{x^2}\right) \exp\left(-\frac{1}{x^2}\right) & \text{if } x \neq 0 \\ 0 & x = 0 \end{cases}$$

The function ρ is smooth, but is not analytic in a neighborhood of the origin. It is easy to check that the automaton is non-blocking and deterministic.

For any $\epsilon > 0$, ρ has an infinite number of zero crossings in the interval $(-\epsilon, 0]$. Therefore, the execution of the hybrid automaton with initial state (q_1, x_0) will take an infinite number of discrete transitions in the finite interval $[\tau_0, \tau_0 + |x_0|)$ (notice that $x_0 < 0$).

■

Example (Water Tank (continued)) We have already shown that the water tank hybrid automaton accepts a unique infinite execution for each initial state. In addition, if the inflow is greater than each of the outflows but is less than their sum ($\max\{v_1, v_2\} < w < v_1 + v_2$), then all infinite executions are Zeno. One can show that

$$\|\tau\| = \tau_0 + \frac{x_1(\tau_0) + x_2(\tau_0) - r_1 - r_2}{v_1 + v_2 - w}$$

■

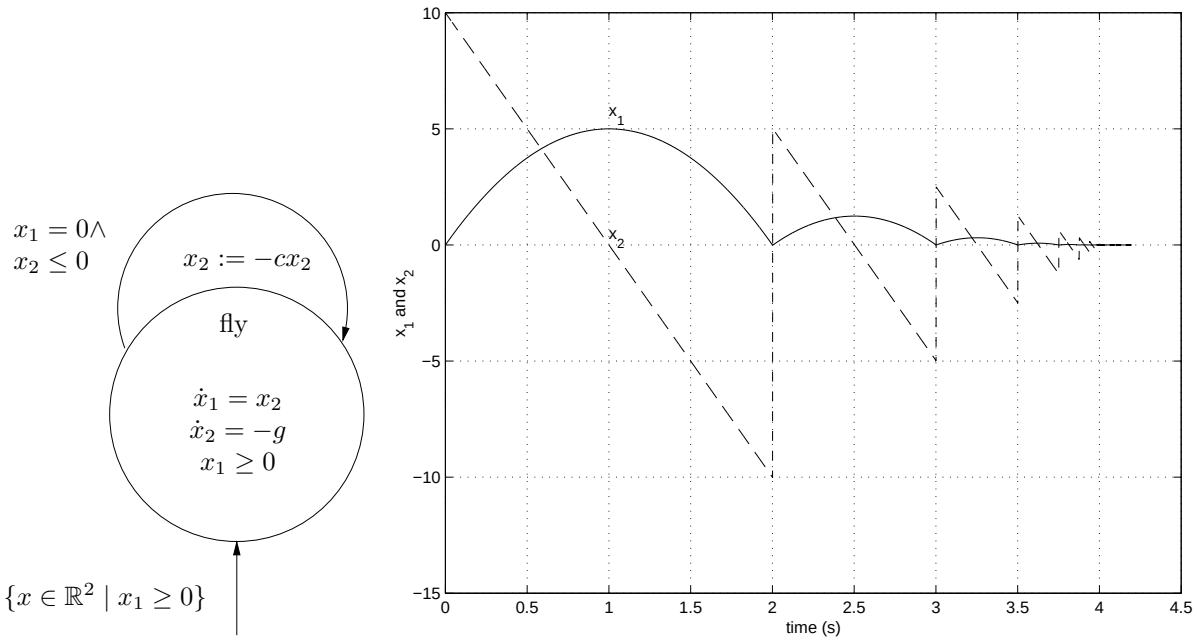

Figure 5: The bouncing ball automaton and one of its Zeno executions.

Example (Bouncing Ball) The boucing ball automaton (Figure 5) is a model of an elastic ball bouncing on a level surface. x_1 denotes the hight of the ball above the surface and x_2 its vertical velocity. It is assumed that the ball loses a fraction $c^2 \in [0, 1]$ of its energy at each bounce. g is the acceleration due to gravity and the mass of the ball is assumed to be 1..

One can show that the bouncing ball hybrid automaton is non-blocking and deterministic. Moreover, if $c < 1$ all infinite executions are Zeno. The first bounce occurs at time:

$$\tau_0' = \tau_1 = \tau_0 + \frac{x_2(\tau_0) + \sqrt{x_2^2(\tau_0) + 2gx_1(\tau_0)}}{g}$$

The second bounce occurs at time:

$$\tau_2 = \tau_1' = \tau_0 + \tau_1 + \frac{2x_2(\tau_1)}{g}$$

More generally, the N^{th} bounce occurs at time:

$$\tau_N = \tau_1' = \tau_0 + \tau_1 + \frac{2x_2(\tau_1)}{g} \sum_{k=1}^{N} c^{k-1}$$

where $x_2(\tau_1) = -cx_2(\tau_0') = c\sqrt{x_2^2(\tau_0) + 2gx_1(\tau_0)}$. Since for $c \in [0, 1)$,

$$\sum_{k=1}^{N} c^{k-1} \to \frac{1}{1-c} \text{ as } N \to \infty \tag{2}$$

we have that

$$\begin{aligned}\|\tau\| &= \tau_0 + \frac{x_2(\tau_0) + \sqrt{x_2^2(\tau_0) + 2gx_1(\tau_0)}}{g} + \frac{2x_2(\tau_1)}{g(1-c)} \\ &= \tau_0 + \frac{x_2(\tau_0)}{g} + \frac{(1+c)\sqrt{x_2^2(\tau_0) + 2gx_1(\tau_0)}}{g(1-c)}\end{aligned}$$

Figure 5 shows a Zeno execution in which $\|\tau\| = 4$. ∎

It can be argued that Zeno behavior is little more than a mathematical curiosity and should not be an issue when dealing with physical systems. However, modeling abstraction, often employed by engineers to simplify models for the purpose of analysis and control, can easily lead to Zeno hybrid models of physical systems, as the water tank and bouncing ball examples illustrate. Zeno executions turn out to be a source of a number of problems in the simulation and analysis of hybrid automata. When faced with Zeno executions, simulation algorithms are likely to stall or produce incorrect results. Analysis algorithms may result in misleading claims as the following example illustrates.

Example (Water Tank (continued)) Consider the water tank automaton and assume that $\max\{v_1, v_2\} < w < v_1 + v_2$. We have already shown that under this assumption Init_{WT} is an invariant set. Therefore, along all executions, the water in both tanks remains above the corresponding levels ($x_1 \geq r_1$ and $x_2 \geq r_2$ for all times). ∎

Though mathematically sound, the above argument is misleading from a practical point of view. If the amount of water coming in to the system is less that the amount of water going out, we know that sooner or later at least one of the tanks will have to drain (just as we know that sooner or later Achilles will overtake the turtle). The point is that one would never be able to infer this fact by analyzing the hybrid automaton model of the water tank system.

4.2 Zeno Hybrid Automata

Zeno executions are a fundamentally hybrid phenomenon. They can not appear in purely discrete or purely continuous systems, since they require the interaction of continuous dynamics (in the form of time) and discrete dynamics (in the form of discrete transitions).

Definition 10 *(Zeno Automaton and Zeno Time) An autonomous hybrid automaton H is called Zeno if for some $(q_0, x_0) \in$ Init all infinite executions are Zeno ($\langle\tau\rangle = \infty$ and $\|\tau\| < \infty$). The continuous extent, $\|\tau\|$, of a Zeno execution is called the Zeno time.*

Definition 11 *(Zeno State) Consider an autonomous hybrid automaton H that accepts a Zeno execution. A state (q, x) is called a Zeno state of this execution if there exists a sequence $\{\theta_j\}_{j=1}^{\infty}$ with $\theta_j \in I_{i_j} \in \tau$ and $\lim_{j\to\infty} \theta_j = \|\tau\|$ such that for all $N > 0$ and all $\epsilon > 0$ and $d((q_{i_j}(\theta_j), x_{i_j}(\theta_j)), (q, x)) < \epsilon$ for some $i > N$.*

The Zeno state is a special case of an ω limit point of an infinite execution. The discrete part of the Zeno state consists of a discrete state that is visited infinitely often by a Zeno execution.

4.3 Resolving the Zeno Phenomenon

In both the bouncing ball and water tank automata, the Zeno behavior is due to modeling simplifications. In the water tank example, the switching dynamics associated with the inflow have been abstracted with an ideal switch, and in the bouncing ball example, the bounce dynamics have been replaced with a simple reset map. In these two examples, an infinite number of transitions takes place in the time interval $(\tau_\infty - \epsilon, \tau_\infty)$ for any $\epsilon > 0$.

The first example above (hybrid automaton modeling chatter) is an example of a Zeno hybrid automaton for which there exists an interval $(\tau_\infty - \epsilon, \tau_\infty)$ in which no transitions take place, while an infinite number of transitions take place at τ_∞. The classical way of analyzing (and controlling!) such systems is by introducing the concept of sliding modes [1, 2].

For the bouncing ball and water tank, we consider an approach known as *regularization*. In the study of differential equations, regularization is a fairly standard process in which one attempts to slightly alter a differential equation whose solution is not well defined, to one whose solution is well defined. The approach can be extended to Zeno hybrid automata, to extend Zeno executions beyond the Zeno time. The general idea and some examples are given below.

- Let H be a non-blocking and deterministic hybrid automaton.
- Assume that for every $(q_0, x_0) \in \text{Init}$ the execution starting at (q_0, x_0) is Zeno.
- We *regularize* H by constructing a family of deterministic, non-blocking, and non-Zeno automata H_ϵ, parameterized by a real valued parameter $\epsilon > 0$, and a continuous map:

$$\phi : Q_\epsilon \times X_\epsilon \to Q \times X \tag{3}$$

  which relates the state of each H_ϵ to the state of H.
- In general $\phi((\tau_\epsilon, q_\epsilon, x_\epsilon)$ will not be an execution of H. However, the construction of H_ϵ is such that H_ϵ tends to H as ϵ tends to 0.

Example (Regularization of bouncing ball): Consider again the bouncing ball automaton of Figure 5.

1. Temporal Regularization: Assume each bounce of the ball takes time $\epsilon > 0$. The regularized automaton is shown in Figure 6. One can shown that the automaton of Figure 6 accepts a unique, non-Zeno execution for each initial state. The state of the regularized system is related to the state of the system by

$$\phi(q_0, (x_1, x_2, x_3)) = \phi(q_1, (x_1, x_2, x_3)) = (q, (x_1, x_2)) \tag{4}$$

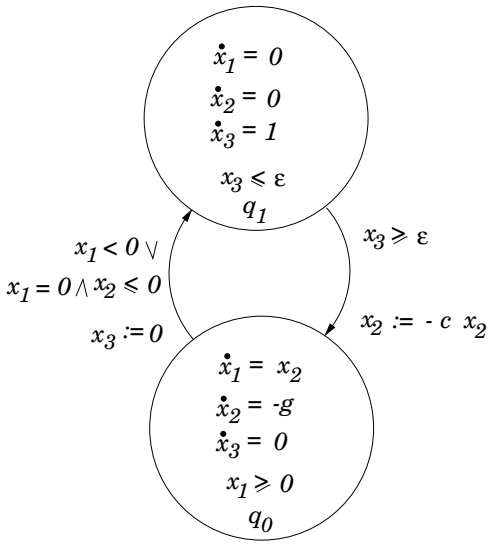

Figure 6: Temporal regularization of bouncing ball automaton.

Figure 7 shows simulation results for the regularized system: x_1, x_2 and q are plotted as functions of time for $\epsilon = 0.1$ and $\epsilon = 0.01$. As ϵ decreases, the execution of the regularized automaton converges to the execution of the actual automaton for $t \in (0, \tau_\infty)$. For $t > \tau_\infty$, the execution converges to the constant $x_1(t) = x_2(t) = 0$.

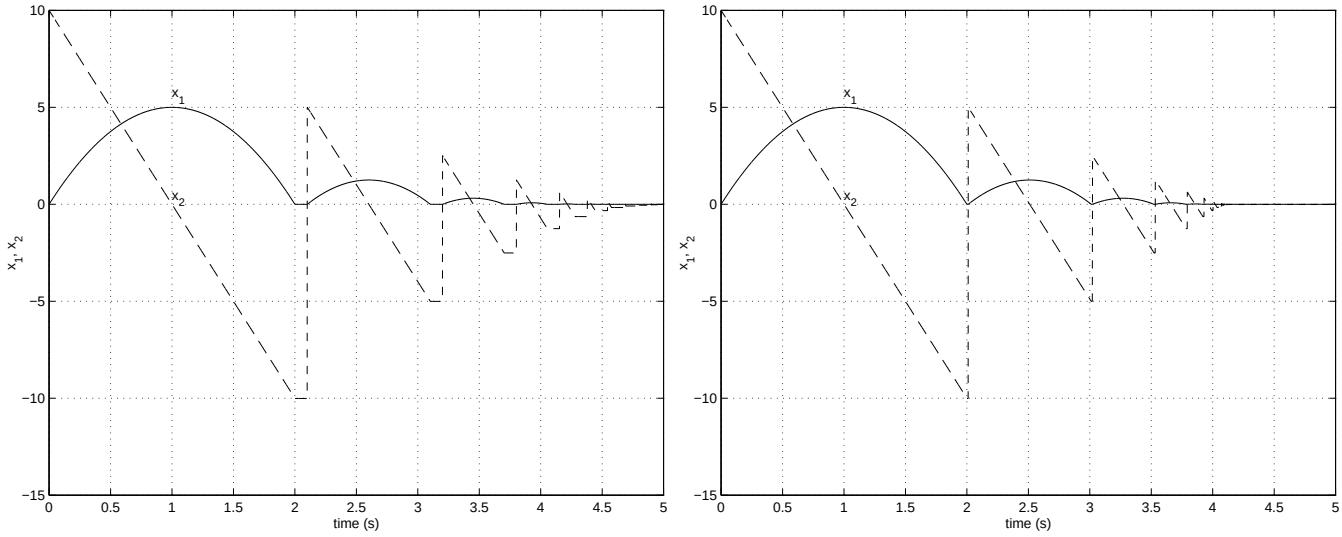

Figure 7: Temporal regularization of bouncing ball automaton ($\epsilon = 0.1$ and $\epsilon = 0.01$).

2. Dynamic Regularization: Assume that the ground is modeled as a stiff spring with spring constant $1/\epsilon$ and no damping. The regularized automaton is shown in Figure 8. One can shown that the automaton of Figure 8 accepts a unique, non-Zeno execution for each initial state. The state of the regularized system is related to the state of the system by

$$\phi(q_0, (x_1, x_2)) = \phi(q_1, (x_1, x_2)) = (q, (x_1, x_2)) \tag{5}$$

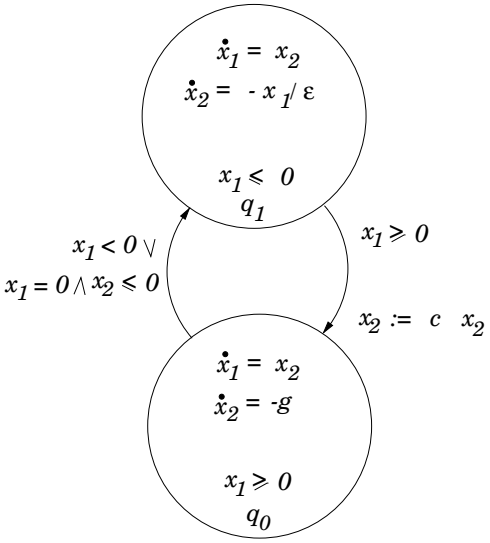

Figure 8: Dynamic regularization of bouncing ball automaton.

Figure 9 shows simulation results for the regularized system: x_1, x_2 and q are plotted as functions of time for $\epsilon = 0.01$ and $\epsilon = 0.001$.

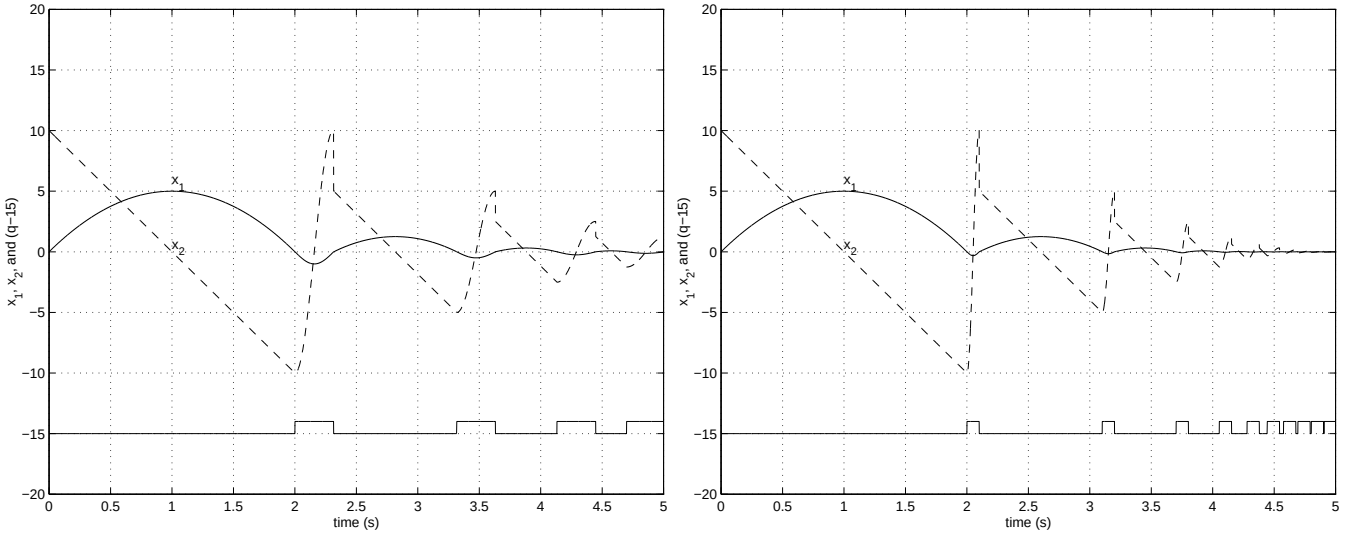

Figure 9: Dynamic regularization of bouncing ball automaton ($\epsilon = 0.01$ and $\epsilon = 0.001$).

Notes and Bibliography

Local existence is arguably the most important of the fundamental properties studied above (and the easiest to check!). One can argue that Zeno is also important, since the presence of Zeno executions makes hybrid automata difficult to simulate, and may lead to misleading claims of safety. However, Zeno hybrid automata can in some sense be very useful (even

realistic) models of physical systems. Therefore, instead of requiring that hybrid automata do not accept Zeno executions, it may be better to deal with Zeno problems when they arise, for example by defining appropriate extensions of Zeno execution beyond the Zeno time. This is the approach taken in [3] motivated by the classical definition of "sliding" flows for discontinuous vector fields.

Uniqueness of execution may also be desirable, since it makes simulation and analysis easier. However, requiring that hybrid automata are deterministic is usually an unnecessarily strict restriction. It is sometimes very desirable to work with non-deterministic hybrid automata, since they allow one to model uncertainty in a physical process and its environment. Moreover, analysis of non-deterministic hybrid automata is not much more difficult. Instead of arguing about *the* execution of the system, one simply has to ensure that *all* possible executions of the system are considered.

References

[1] A. F. Filippov. *Differential equations with discontinuous right hand sides.* Kluwer Academic Publishers, Boston, 1988.

[2] V. I. Utkin. *Sliding Modes in Control Optimization.* Springer Verlag, Berlin, 1992.

[3] K. J. Johansson, M. Egerstedt, J. Lygeros, and S. Sastry. On the regularization of zeno hybrid automata. *Systems and Control Letters*, 38:141–150, 1999.