Hybrid Systems:
Foundations, advanced topics and applications

John Lygeros, Shankar Sastry, and Claire Tomlin(order tbd)

February 1, 2012

Preface

The aim of this book is to introduce some fundamental concepts from the area of hybrid systems, that is dynamical systems that involve the interaction of continuous (real valued) states and discrete (finite valued) states.

Various chapters are peppered with simple exercises, meant to enhance the learning experience and highlight subtle points. They should be easy (if not trivial) to solve for students who follow the material. Normally students are not asked to turn in solutions for these exercises. The exercises are meant to help them understand the material not to test their understanding. Students are however encourage them to discuss the solutions with the instructor if they have problems. Lengthier exercises that can form the basis of homework and exam questions are given at the end of most chapters.

Contents

List of Figures

List of Tables

Chapter 1

Motivation and Examples of Hybrid Systems

While rapid progress in embedded hardware and software makes plausible ever more ambitious multi-layer, multi-objective, adaptive, nonlinear control systems, adequate design methodologies and design support lag far behind. Consequently, today most of the cost in control system development is spent on ad-hoc, prohibitively expensive systems integration, and validation techniques that rely almost exclusively on exhaustively testing more or less complete versions of complex nonlinear control systems. The newest research direction in control addresses this bottleneck by focusing on *predictive* and *systematic hierarchical design* methodologies for building an analytical foundation based on *hybrid systems* and a practical set of *software design tools* which support the construction, integration, safety and performance analysis, on-line adaptation and off-line functional evolution of multi-agent hierarchical control systems. *Hybrid* systems refer to the distinguishing fundamental characteristics of software-based control systems, namely, the tight coupling and interaction of discrete with continuous phenomena. Hybridness is characteristic of all embedded control systems because it arises from several sources. First, the high-level, abstract protocol layers of *hierarchical* control designs are discrete as to make it easier to manage system complexity and to accommodate linguistic and qualitative information; the low-level, concrete control laws are naturally continuous. Second, while individual feedback control scenarios are naturally modeled as interconnections of modules characterized by their continuous input/output behavior, *multi-modal* control naturally suggests a state-based view, with states representing discrete control modes; software-based control systems typically encompass an integrated mixture of both

types. Third, every digital *hardware/software implementation* of a control design is ultimately a discrete approximation that interacts through sensors and actuators with a continuous physical environment. The mathematical treatment of hybrid systems is interesting in that it builds on the preceding framework of nonlinear control, but its mathematics is qualitatively distinct from the mathematics of purely discrete or purely continuous phenomena. Over the past several years, we have begun to build basic formal models (*hybrid automata*) for hybrid systems and to develop methods for hybrid control law design, simulation, and verification. Hybrid automata, in particular, integrate diverse models such as differential equations and state machines in a single formalism with a uniform mathematical semantics and novel algorithms for multi-modal control synthesis and for safety and real-time performance analysis.

1.1 Multi-Agent Systems and Hybrid Systems

To a large extent control theory has thus far investigated the paradigm of "Centralized Control". In this paradigm, sensory information is collected from sensors observing a material process that may be distributed over space. This information is transmitted over a communication network to one center, where the commands that guide the process are calculated and transmitted back to the process actuators that implement those commands. In engineering practice, of course, as soon as the process becomes even moderately large, the central control paradigm breaks down. What we find instead is distributed control: A set of control stations, each of whom receives some data and calculates some of the actions. Important examples of distributed control are the Air Traffic Management System, the control system of an interconnected power grid, the telephone network, a chemical process control system, and automated highway transportation systems. Although a centralized control paradigm no longer applies here, control engineers have with great success used its theories and its design and analysis tools to build and operate these distributed control systems. There are two reasons why the paradigm succeeded in practice, even when it failed in principle. First, in each case the complexity and scale of the material process grew incrementally and relatively slowly. Each new increment to the process was controlled using the paradigm, and adjustments were slowly made after extensive (but by no means exhaustive) testing to ensure that the new controller worked in relative harmony with the existing controllers. Second, the processes were operated with a considerable degree of "slack." That is, the process was operated well within its performance limits to permit errors in the extrapolation of test results to untested situations and to tolerate a small degree of disharmony among the controllers. However, in each system mentioned above, there were occasions when the

material process was stressed to its limits and the disharmony became intolerable, leading to a spectacular loss of efficiency. For example, most air travelers have experienced delays as congestion in one part of the country is transmitted by the control system to other parts. The distributed control system of the interconnected power grid has sometimes failed to respond correctly and caused a small fault in one part of a grid to escalate into a system-wide blackout.

We are now attempting to build control systems for processes that are vastly more complex or that are to be operated much closer to their performance limits in order to achieve much greater efficiency of resource use. The attempt to use the central control paradigm cannot meet this challenge: the material process is already given and it is not practicable to approach its complexity in an incremental fashion as before. Moreover, the communication and computation costs in the central control paradigm would be prohibitive, especially if we insist that the control algorithms be fault-tolerant. What is needed to meet the challenge of control design for a complex, high performance material process, is a new paradigm for distributed control. It must distribute the control functions in a way that avoids the high communication and computation costs of central control, at the same time that it limits complexity. The distributed control must, nevertheless, permit centralized authority over those aspects of the material process that are necessary to achieve the high performance goals. Such a challenge can be met by organizing the distributed control functions in a hierarchical architecture that makes those functions relatively autonomous (which permits using all the tools of central control), while introducing enough coordination and supervision to ensure the harmony of the distributed controllers necessary for high performance. Consistent with this hierarchical organization are sensing hierarchies with fan-in of information from lower to higher levels of the hierarchy and a fan-out of control commands from the higher to the lower levels. Commands and information at the higher levels are usually represented symbolically, calling for discrete event control, while at the lower levels both information and commands are continuous, calling for continuous control laws. Interactions between these levels involves hybrid control. In addition protocols for coordination between individual agents are frequently symbolic, again making for hybrid control laws. The hybrid control systems approach has been successful in the control of some extremely important multi-agent systems such as automated highway systems [1, 2, **?**], air traffic control [3], groups of unmanned aerial vehicles, underwater autonomous vehicles [**?**], mobile offshore platforms, to give a few examples.

1.2 Motivating Applications

1.2.1 Automated Highway Systems (John)

Highway congestion is an increasing problem, especially in and around urban areas. One of the promising solutions considered for this problem is traffic automation, either partial or full. The use of an automated system that performs some or all of the tasks of the driver may reduce or eliminate human errors and hence improve safety. Moreover, as the automatic controller can react to disturbances faster than a human driver, automation may also decrease the average inter-vehicle spacing and hence increase throughput and reduce congestion and delays.

The design of an Automated Highway System (AHS) is an extremely challenging control problem, and a number of alternatives have been proposed for addressing it. One of the most forward-looking AHS designs involves a fully automated highway system that supports platooning of vehicles. The platooning concept [1] assumes that traffic on the highway is organised in groups of tightly spaced vehicles (platoons). The first vehicle of a platoon is called the leader, while the remaining vehicles are called followers. The platooning structure achieves a balance between safety and throughput: it is assumed that the system is safe even if in emergency situations (for example, as a result of a failure) collisions do occur, as long as the relative velocity at impact is low. Of course no collisions should take place during normal operation. This gives rise to two safe spacing policies. The obvious one is that of the leaders, who are assumed to maintain a large inter-platoon spacing (of the order of 30 to 60 meters). The idea is that the leader has enough time to stop without colliding with the last vehicle of the platoon ahead. The more unintuitive spacing policy is that of the followers, who are assumed to maintain tight intra-platoon spacing (of the order of 1 to 5 meters). In case of emergency, collisions among the followers of a platoon may take place, but, because of the tight spacing, they are expected to be at low relative velocities. Recent theoretical, computational and experimental studies have shown that an AHS that supports platooning is not only technologically feasible but, if designed properly, may lead to an improvement of both the safety and the throughput of the highway system, under normal operation.

Implementation of the platooning concept requires automatic vehicle control, since human drivers are not fast and reliable enough to produce the necessary inputs. To manage the complexity of the design process a hierarchical controller is used. The controller is organised in four layers (Figure 1.1). The top two layers, called network and link, reside on the roadside and are primarily concerned with throughput maximisation, while the bottom two, called coordination and regulation, reside on the vehicles and are primarily concerned with safety. The physical layer is not part of

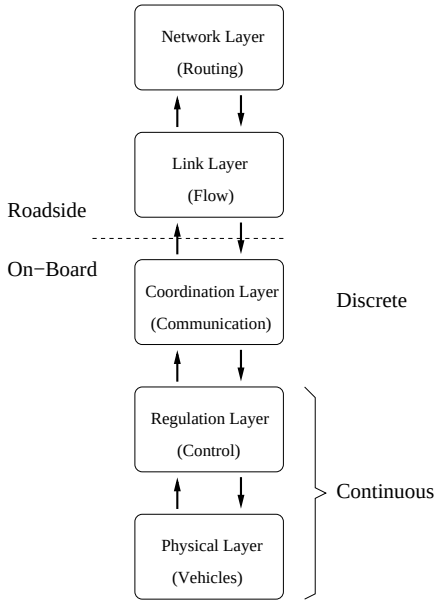

Figure 1.1. The AHS control hierarchy.

the controller. It contains the "plant", i.e. the vehicles and highway, with their sensors, actuators and communication equipment.

The network layer is responsible for the flow of traffic on the entire highway system, for example, several highways around an urban area. Its task is to prevent congestion and maximise throughput by dynamically routing traffic. The link layer coordinates the operation of sections (links) of the highway (for example the highway segment between two exits). Its primary concern is to maximise the throughput of the link. With these criteria in mind, it calculates an optimum platoon size and an optimum velocity and decides which lanes the vehicles should follow. It also monitors incidents and diverts traffic away from them, in an attempt to minimise their impact on traffic flow.

The coordination layer coordinates the operation of neighbouring platoons by choosing manoeuvres that the platoons need to carry out. For normal operation, these manoeuvres are *join* to join two platoons into one, *split* to break up one platoon into two, *lane change*, *entry* and *exit*. The coordination layer is primarily a discrete controller. It uses communication protocols, in the form of finite state machines, to coordinate the execution of these manoeuvres between neighbouring vehicles.

The regulation layer receives the coordination layer commands and readings from the vehicle sensors and generates throttle, steering and braking commands for the vehicle actuators. For this purpose it utilises a number of continuous time feedback control laws that use the readings provided

by the sensors to calculate the actuator inputs required for a particular manoeuvre. In addition to the control laws needed for the manoeuvres, the regulation layer makes use of two *default* controllers, one for leader and one for follower operation.

The interaction between the coordination layer (which is primarily discrete) and the regulation layer (which is primarily continuous) gives rise to interesting hybrid dynamics. To ensure the safety of the AHS, one needs to verify that the closed loop hybrid system does not enter a bad region of its state space (e.g. does not allow any two vehicles to collide at high relative velocity). This issue can be addressed by posing the problem as a game between the control applied by one vehicle and the disturbance generated by neighbouring vehicles. It can be shown that information available through discrete coordination can be used together with appropriate continuous controllers to ensure the safety of the closed loop hybrid system.

1.2.2 Air Traffic Management (Claire)

The introduction of advanced automation into manually operated systems has been extremely successful in increasing the performance and flexibility of such systems, as well as significantly reducing the workload of the human operator. Examples include the automation of mechanical assembly plants, of the telephone system, of the interconnected power grid, as well as transportation system automation such as controllers in high speed trains, automatic braking systems in automobiles, and avionics on board a commercial jet, the results could be disastrous.

Many of today's safety critical systems are growing at such a rate that will make manual operation of them extremely difficult if not impossible in the near future. The Air Traffic Control (ATC) system is an example of such a safety critical system. Air traffic in the United States alone is expected to grow by 5% annually for the next 15 years [**?**], and rates across the Pacific Rim are expected to increase by more than 15% a year. Even with today's traffic, ground holds and airborne delays in flights due to congestion in the skies have become so common that airlines automatically pad their flight times with built-in delay times. Aging air traffic control equipment certainly contributes to these delays:the plan view displays used by controllers to look at radar tracks and flight information are the very same that were installed in the early 1970's, and they fail regularly. The computer systems which calculate radar tracks and store flight plans were designed in the 1980's, using software code that was written in 1972. The introduction of new computers, display units, and communication technologies for air traffic controllers will help alleviate the problems caused by failing equipment, yet the Federal Aviation Administration (FAA) admits that any significant improvement will require that many of the basic practices of ATC be automated [**?**]. For example, today's airspace has a rigid route structure based on altitude and on ground-based navigational

"fixes": current practice of air traffic controllers is to route aircraft along predefined paths connecting fixes, to manage the complexity of route planning for several aircraft at once. The rigid structure puts strict constraints on aircraft trajectories, which could otherwise follow wind-optimal or user preferred routes. Also, while a data link between aircraft and ground is being investigated as a replacement for the current voice communication over radio channels between pilot and controller, there is a limit to the amount of information processing that a controller can perform with this data. Studies in [**?**] indicate that, if there is no change to the structure of ATC, then by the year 2015 there could be a major accident every 7 to 10 days.

The result is a perceived need in the air traffic, airline, and avionics communities for a new *architecture*, which integrates new technologies for data storage, processing, communications, and display, into a safe and efficient air traffic management system. The airlines are proponents of a decentralized architecture featuring *free flight*, meaning that each aircraft plans and tracks its own dynamic trajectory with minimal interference from ATC [**?**]. Many people (air traffic controllers in particular) view this as a radical solution, but a recent study funded by NASA [**?**] suggests that distributing some of the control authority to each aircraft would help improve the efficiency of the system as a whole. In [**?**] we proposed an architecture for a new air traffic management system along these lines, in which the aircraft's flight management system uses local sensory information from Global Positioning Systems, Inertial Navigation Systems, and broadcast communication with other aircraft to resolve local conflicts without requesting clearances from ATC. While the degree of decentralization and level of automation in a new air traffic management system are still under debate (since it is very difficult to estimate the increase in efficiency from distributing the control authority), the integrity of any automated functionality in a new air traffic management system depends on a *provably-safe* design, and a high confidence that the control actions won't fail.

In the past, high confidence has been achieved by operating the system well within its performance limits. Extensive testing has been used to validate operations, and any errors occurring from untested situations would be compensated for by this degree of "slack" in the system performance. We would like to maintain high confidence but operate the system much closer to its performance limits. In order to do this, we require accurate models of the system, procedures for verifying that the design is safe to within the accuracy of these models, and procedures for synthesizing control actions for the system, so that safety is maintained.

For about the past six years, researchers in the traditionally distinct fields of control theory and computer science verification have proposed models, and verification and controller synthesis techniques for complex, safety critical systems. The area of *hybrid systems* is loosely defined as the study of systems which involve the interaction of discrete event and continuous time

dynamics, with the purpose of proving properties such as reachability and stability. The discrete event models naturally accommodate linguistic and qualitative information, and are used to model modes of operation of the system, such as the mode of flight of an aircraft, or the interaction and coordination between several aircraft.

1.2.3 Biochemical Networks (Claire)

1.3 Bibliography and Further Reading

Continuous state systems and their properties have been studied extensively in mathematics engineering, economics, biology, etc. The literature is vast and there are a number of excellent textbooks available (see for example [4, 5, 6, 7]). Discrete state systems have also been studied for many years, especially in computer science. Good textbooks include [8, 9, 10].

By comparison, the study of hybrid systems is relatively recent. One class of approaches to modeling and analysis of hybrid systems has been to extend techniques for finite state automata to include systems with simple continuous dynamics. These approaches generally use one of two analysis techniques: model checking, which verifies a system specification symbolically on all system trajectories, and deductive theorem proving, which proves a specification by induction on all system trajectories. Emphasis is placed on *computability* and *decidability*, or proving that the problem: *Does the system satisfy the specification?* can be solved in a finite number of steps. Models and decidability results have been obtained for timed automata [11], linear hybrid automata [12], and hybrid input/output automata [13]. Linear hybrid automata model or abstract the continuous dynamics by differential inclusions of the form $A\dot{x} \leq b$ and verify properties of the resulting abstracted system [**?**, **?**]. While reachability and eventuality properties for timed automata have been shown to be decidable, the decidability results for linear hybrid automata are fairly narrow. For all but the simplest continuous linear dynamics (two-dimensional rectangular differential inclusions), reachability properties are semi-decidable at best, and in most cases undecidable. Methods for designing discrete controllers for timed and hybrid systems have been developed using this framework [14, 15], and computational tools have been developed for both model checking [16, **?**], and theorem proving [**?**]. A second class of models and analysis techniques for hybrid systems has developed out of research in continuous state space and continuous time dynamical systems and control. The emphasis here has been on extending the standard modeling, reachability and stability analyses, and controller design techniques to capture the interaction between the continuous and discrete dynamics [17, **?**, **?**, **?**, **?**, 18]. Analysis and design techniques extend existing control techniques, such as stability theory [**?**], optimal control [**?**, **?**, 18], and control of discrete event systems

[19, 20], to hybrid systems. One area in which results have been hard to come by is the efficient *computation* of reachable sets for hybrid systems whose dynamics are nonlinear or are of order greater than one. Only recently, some attempts to directly approach this problem have been reported in the literature [21, 22].

The few books that have appeared on the subject to date [23, 24] have a research monograph "flavour" and address specific topics and classes of systems; [24] however also contains a substantial textbook style overview. Another good source of material are the special issues devoted by a number of journals on the topic of hybrid systems [25, 26, 27, 28, 29]. Finally, the series of edited volumes based on the proceedings of hybrid systems workshops form another excellent source of material [**?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**].

Hybrid systems arise naturally in a number of engineering applications, in addition to the ones listed in this chapter. For example, the hybrid paradigm has also been used successfully to address problems in air traffic control [3], automotive control [30], bioengineering [**?**], chemical process control [31, 32], highway systems [1, 33] and manufacturing [34].

Our approach to hybrid systems modeling incorporates accurate, nonlinear models of the continuous dynamics with models for discrete event dynamics. We include continuous and discrete input variables to model both parameters that the designer may control as well as disturbance parameters that the designer must control against. Using analysis based on traditional discrete and continuous optimal control techniques, and on two-person zero-sum game theory for automata and continuous dynamical systems, we derive the Hamilton-Jacobi partial differential equations whose solutions describe *exactly* the boundaries of reachable sets. Only then do we approximate: we use a clever numerical technique to solve this equation. These equations are the heart of our general controller synthesis technique for hybrid systems, in which we calculate feedback control laws for the continuous and discrete variables which guarantee that the hybrid system remains in the "safe subset" of the reachable set. While about 10 years ago such a method would have been prohibitively computationally expensive, advances in computational power and new fast methods for integrating PDEs have made such solutions feasible, even for real-time applications. The result is an analytic and numerical method for computing reachable sets and control laws for hybrid systems, which doesn't require a preprocessing step to approximate the dynamics. We have been successful in computing solutions to finite-time examples, but in our method thus far, we have not addressed considerations of decidability and computational complexity.

Chapter 2
Introduction to Dynamical Systems

In this chapter we begin with a review of dynamical systems, continuous time, discrete time, continuous state and discrete state. By hybrid systems, we mean systems that are hierarchies and interconnections of continuous and discrete state systems in continuous time and discrete time. However, before we launch into the formal definition and study of hybrid systems we start with an overview of the main themes of this book (modeling, reachability, control of hybrid systems) as they manifest themselves in the more classical discrete and continuous systems literature. We will highlight the different classes of dynamics of interest that will be of interest in our study of hybrid systems.

2.1 Dynamical System Classification

A *dynamical system* describes the **evolution** of *state* variables, typically real valued over *time*. However, we will need to specify precisely what we mean by the terms "evolution", "state" and "time". Some dynamical systems can also be influenced by exogenous inputs, which represent either uncontrollable disturbances or controlled input signals. For example the control system of a modern aircraft may be controlled by inputs such as the amount of thrust, the settings of the wing ailerons and tail flaps, but are influenced by disturbances like wind conditions. Other dynamical systems have outputs, which represent either quantities that can be measured,

or quantities that need to be controlled or regulated. Dynamical systems with both inputs and outputs are sometimes referred to as *control systems*.

Based on the type of their state variables, dynamical systems may be classified into the following categories:

1. **Continuous State:** If the state takes values in Euclidean space $\mathbb{R}^n$ for some $n \geq 1$. We will use $x \in \mathbb{R}^n$ to denote the state of a continuous dynamical system.

2. **Discrete State:** If the state takes values in a finite or countable set $\{q_1, q_2, \ldots\}$. We will use q to denote the state of a discrete system. For example, a light switch is a system whose state takes on two values, $q \in \{ON, OFF\}$. A computer is also a dynamical system whose state takes on a finite (albeit very large) number of values.

3. **Hybrid State Variables:** If a part of the state takes values in $\mathbb{R}^n$, and an other part takes values in a finite set. For example, the closed loop system that is obtained when we use computer control software to control an inverted pendulum is hybrid: part of the state (namely the position, velocity, etc. of the pendulum) is continuous, while another part (namely the state of the variables in the control software) is discrete.

Based on the time set over which the state evolves, dynamical systems can be classified as:

1. **Continuous time:** If the set of times is a subset of the real line $\mathbb{R}$. We will use $t \in \mathbb{R}$ to denote continuous time. For a number of systems of interest to us with continuous state evolving in continuous time, the evolution of the state is described by an **ordinary differential equation** (ODE). An especially simple example is the linear, continuous time system in state space form

$$\dot{x} = Ax.$$

2. **Discrete time:** If the set of time variables is a subset of the integers. We will use $k \in \mathbb{Z}$ to denote discrete time. Typically, the evolution of the state of continuous state, discrete time system is described by a **difference equation**. Think of the linear discrete time system in state space form

$$x_{k+1} = Ax_k.$$

3. **Hybrid time:** This covers the instance when the evolution is over continuous time but there are also discrete "instants" where something "special" happens. Consider for example a continuous system with switches: the dynamics of the systems changes at the switching times.

Continuous state systems can be further classified according to the equations used to describe the evolution of their state

1. **Linear**, if the evolution is governed by a linear differential equation (continuous time) or difference equation (discrete time).
2. **Nonlinear**, if the evolution is governed by a nonlinear differential equation (continuous time) or difference equation (discrete time).

Exercise 2.1 *The linear vs nonlinear classification generally does not apply to discrete state or hybrid systems. Why?*

2.2 Standard Dynamical Systems Examples

We begin with some examples of the following classes of systems:

1. Nonlinear (continuous state), continuous time systems.
2. Nonlinear, discrete time systems.
3. Discrete state, discrete time systems.

Of course, one could also have examples of discrete state, continuous time systems.

2.2.1 Pendulum: Nonlinear, Continuous Time System

Consider a pendulum hanging from a weightless solid rod and moving under gravity (Figure 2.1). Let θ denote the angle that the pendulum makes with the downward vertical, l the length of the pendulum, m its mass, and d the dissipation constant. The evolution of θ is governed by

$$ml\ddot{\theta} + dl\dot{\theta} + mg\sin(\theta) = 0$$

This is a nonlinear, second order, ordinary differential equation (ODE).

Exercise 2.2 *Derive this equation from Newton's laws. What is it that makes this ODE nonlinear?*

To determine how the pendulum is going to move, i.e. determine θ as a function of time, we would like to find a solution to this ODE . Assuming that at time $t = 0$ the pendulum starts as some initial position θ_0 and with some initial velocity $\dot{\theta}_0$, "solving the ODE" means finding a function of time

$$\theta(\cdot) : \mathbb{R} \to \mathbb{R}$$

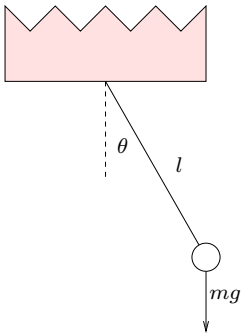

Figure 2.1. The pendulum

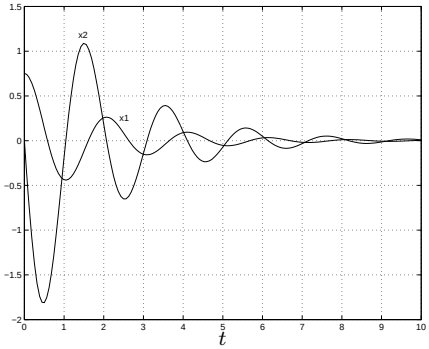

Figure 2.2. Trajectory of the pendulum.

such that

$$\begin{aligned} \theta(0) &= \theta_0 \\ \dot{\theta}(0) &= \dot{\theta}_0 \\ ml\ddot{\theta}(t) + dl\dot{\theta}(t) + mg\sin(\theta(t)) &= 0,\ \forall t \in \mathbb{R} \end{aligned}$$

Such a function is known as a *trajectory* (or *solution*) of the system. At this stage it is unclear if one, none or multiple trajectories exist for this initial condition. *Existence* and *uniqueness* of trajectories are both desirable properties for ODE that are used to model physical systems.

For nonlinear systems, even if a unique trajectory exists for the given initial condition, it is usually difficult to construct explicitly. Frequently solutions of ODE can only be approximated by *simulation*. Figure 2.2 shows a simulated trajectory of the pendulum for $l = 1$, $m = 1$, $d = 1$, $g = 9.8$, $\theta(0) = 0.75$ and $\dot{\theta}(0) = 0$.

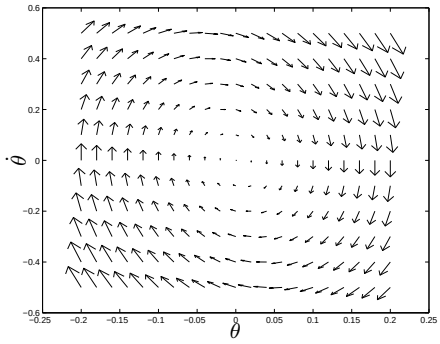

Figure 2.3. The pendulum vector field.

To simplify the notation we typically write dynamical system ODE in *state space* form

$$\dot{x} = f(x)$$

where x is now a vector in $\mathbb{R}^n$ for some appropriate $n \geq 1$. The easiest way to do this for the pendulum is to set

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}$$

which gives rise to the state space equations

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{g}{l}\sin(x_1) - \frac{d}{m}x_2 \end{bmatrix} = f(x)$$

The vector

$$x \in \mathbb{R}^2$$

is called the *state* of the system. The size of the state vector (in this case $n = 2$) is called the *dimension* of the system. Notice that the dimension is the same as the order of the original ODE. The function

$$f(\cdot) : \mathbb{R}^2 \to \mathbb{R}^2$$

which describes the dynamics is called a *vector field*, because is assigns a "velocity" vector to each state vector. Figure 2.3 shows the vector field of the pendulum.

Exercise 2.3 *Other choices are possible for the state vector. For example, for the pendulum one can use $x_1 = \theta^3 + \dot{\theta}$ and $x_2 = \dot{\theta}$. What would the vector field be for this choice of state?*

Solving the ODE for θ is equivalent to finding a function

$$x(\cdot) : \mathbb{R} \to \mathbb{R}^2$$

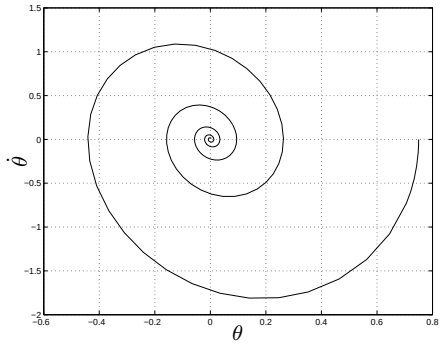

Figure 2.4. Phase plane plot of the trajectory of Figure 2.2.

such that

$$x(0) = \begin{bmatrix} x_1(0) \\ x_2(0) \end{bmatrix} = \begin{bmatrix} \theta_0 \\ \dot{\theta}_0 \end{bmatrix}$$
$$\dot{x}(t) = f(x(t)),\ \forall t \in \mathbb{R}.$$

For two dimensional systems like the pendulum it is very convenient to visualise the solutions by *phase plane* plots. These are plots of $x_1(t)$ vs $x_2(t)$ parameterised by time (Figure 2.4).

2.2.2 Logistic Map: Nonlinear Discrete Time System

The logistic map

$$x_{k+1} = ax_k(1 - x_k) = f(x_k) \tag{2.1}$$

is a nonlinear, discrete time dynamical system that has been proposed as a model for the fluctuations in the population of fruit flies in a closed container with constant food supply [35]. We assume that the population is measured at discrete times (e.g. generations) and that it is large enough to be assumed to be a continuous variable. In the terminology of the previous section, this is a one dimensional system with state $x_k \in \mathbb{R}$, whose evolution is governed by the difference equation (2.1) given above. $a \in \mathbb{R}$ is a parameter that reflects the living space and food supply.

The shape of the function f (Figure 2.5) reflects the fact that when the population is small it tends to increase due to abundance of food and living space, whereas when the population is large it tends to decrease, due to competition for food and the increased likelihood of epidemics. Assume that $0 \leq a \leq 4$ and that the initial population is such that $0 \leq x_0 \leq 1$.

Exercise 2.4 *Show that under these assumptions* $0 \leq x_k \leq 1$ *for all* $k \in \mathbb{Z}$ *with* $k \geq 0$.

The behaviour of x_k as a function of k depends on the value of a.

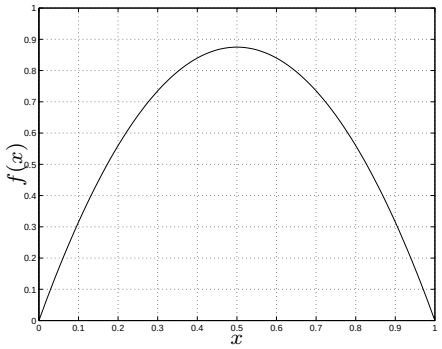

Figure 2.5. The logistic map.

1. If $0 \leq a < 1$, x_k decays to 0 for all initial conditions $x_0 \in [0, 1]$. This corresponds to a situation where there is inadequate food supply to support the population.

2. If $1 \leq a \leq 3$, x_k tends to a steady state value. In this case the population eventually stabilises.

3. If $3 < a \leq 1 + \sqrt{6} = 3.449$, x_k tends to a 2-periodic state. This corresponds to the population alternating between two values from one generation to the next.

As a increases further more and more complicated patterns are obtained: 4-periodic points, 3-periodic points, and even chaotic situations, where the trajectory of x_k is a-periodic (i.e. never meets itself).

2.2.3 Manufacturing Machine: A Discrete State System

Consider a machine in a manufacturing plant that processes parts of type p one at a time. The machine can be in one of three states: Idle (I), Working (W) or Down (D). The machine can transition between the states depending on certain events. For example, if the machine is idle and a part p arrives it will start working. While the machine is working it may break down. While the machine is down it may be repaired, etc.

Abstractly, such a machine can be modelled as a dynamical system with a *discrete state*, q, taking three values

$$q \in Q = \{I, W, D\}$$

The state "jumps" from one value to another whenever one of the *events*, σ occurs, where

$$\sigma \in \Sigma = \{p, c, f, r\}$$

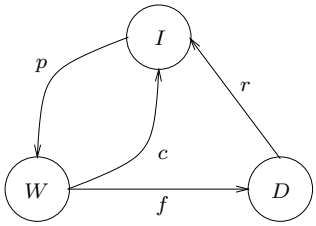

Figure 2.6. The directed graph of the manufacturing machine automaton.

(p for "part arrives", c for "complete processing", f for "failure" and r for "repair"). The state after the event occurs is given by a *transition relation*

$$\delta : Q \times \Sigma \to Q$$

Since both Q and Σ are finite sets, one can specify δ by enumeration.

$$\begin{aligned} \delta(I, p) &= W \\ \delta(W, c) &= I \\ \delta(W, f) &= D \\ \delta(D, r) &= I \end{aligned}$$

δ is undefined for the rest of the combinations of q and σ. This reflects the fact that certain events may be impossible in certain states. For example, it is impossible for the machine to start processing a part while it is down, hence $\delta(D, p)$ is undefined.

Exercise 2.5 *If the discrete state can take n values and there are m possible events, what is the maximum number of lines one may have to write down to specify δ?*

Such a dynamical system is called an *automaton*, or a *finite state machine*. Automata are special cases of *discrete event systems*. Discrete event systems are dynamical systems whose state also jumps depending on events but can take on an infinite number of values. The dynamics of a finite state machine can be represented compactly by a *directed graph* (Figure 2.6). This is a graph whose *nodes* represent the possible values of the state (in this case I, W, D). The *arcs* of the graph represent possible transitions between the state values and are labelled by the events.

Exercise 2.6 *What is the relation between the number of arcs of the graph and the number of lines one needs to write down in order to specify δ?*

Assume that the machine starts in the idle state $q_0 = I$. What are the sequences of events the machine can experience? Clearly some sequences are possible while others are not. For example, the sequence *pcp* is possible: the machine successfully processes one part and subsequently starts processing a second one. The sequence *ppc*, on the other hand is not possible: the

machine can not start processing a second part before the previous one is complete. More generally, any sequence that consists of an arbitrary number of pc's (possibly followed by a single p) is an acceptable sequence. In the discrete event literature this set of sequences is compactly denoted as

$$(pc)^*(1+p)$$

where * denotes an arbitrary number (possibly zero) of pc's, 1 denotes the empty sequence (no event takes place), and + denotes "or".

Likewise, pfr is a possible sequence of events (the machine starts processing a part, breaks down and then gets repaired) while pfp is not (the machine can not start processing a part while it is down). More generally, any sequence that consists of an arbitrary number of pfr's (possibly followed by a p or a pf) is an acceptable sequence.

Exercise 2.7 *Write this set of sequences in the discrete event notation given above.*

The set of all sequences that the automaton can experience is called the *language* of the automaton. The above discussion suggests that the language of the machine automaton is

$$(pc+pfr)^*(1+p+pf)$$

It is important to understand the properties of these languages, for example to determine how to schedule the work in a manufacturing plant.

2.3 Examples of Hybrid State Systems

Roughly speaking, hybrid systems are dynamical systems that involve the interaction of different types of dynamics. Here we are interested in hybrid dynamics that arise out of the interaction of continuous state dynamics and discrete state dynamics. A state variable is called discrete if it takes on a finite (or countable) number of values and continuous if it takes values in Euclidean space $\mathbb{R}^n$ for some $n \geq 1$. By their nature, discrete states can change value only through a discrete "jump" (c.f. the machining example in the previous section. Continuous states can change values either through a jump (c.f. the logistic map example in the previous section, or by "flowing" in continuous time according to a differential equation as seen in the previous section. Hybrid systems involve both these types of dynamics: discrete jumps and continuous flows. The analysis and design of hybrid systems is, in general, more difficult than that of purely discrete or purely continuous systems, because the discrete dynamics may affect the continuous evolution and vice versa.

Hybrid dynamics provide a convenient framework for modeling systems in a wide range of engineering applications:

- In **mechanical systems** continuous motion may be interrupted by collisions.
- In **electrical circuits** continuous phenomena such as the charging of capacitors, etc. are interrupted by switches opening and closing, or diodes going on or off.
- In **chemical process control** the continuous evolution of chemical reactions is controlled by valves and pumps.
- In **embedded computation** systems a digital computer interacts with a mostly analog environment.

In all these systems it is convenient (and usually fairly accurate) to model the "discrete" components (switches, valves, computers, etc.) as introducing instantaneous changes in the "continuous" components (charging of capacitors, chemical reactions, etc.). To motivate the analysis that follows, we start with some examples. In the examples that follow we no longer make a distinction between continuous time and discrete time: rather the evolution always follows the hybrid time trajectory.

2.3.1 The Bouncing Ball

A model for a bouncing ball can be represented as a simple hybrid system (Figure 2.7) with single discrete state and a continuous state of dimension two

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix},$$

where x_1 denotes the vertical position of the ball and x_2 its vertical velocity.

The continuous motion of the ball is governed by Newton's laws of motion. This is indicated by the differential equation that appears in the *vertex* (box), where g denotes the gravitational acceleration. This differential equation is only valid as long as $x_1 \geq 0$, i.e., as long as the ball is above the ground. This is indicated by the logical expression $x_1 \geq 0$ that appears in the vertex below the differential equation.

The ball bounces when $x_1 = 0$ and $x_2 \leq 0$, i.e. when it hits the ground on the way down. This is indicated by the logical expression that appears near the beginning of the edge (arrow). At each bounce, the ball loses a fraction of its energy. This is indicated by the equation $x_2 := -cx_2$ (with $c \in [0, 1]$) that appears near the end of the edge. This is an assignment statement, which means that after the bounce the speed of the ball will be c times the speed of the ball before the bounce, and in the opposite direction (going up).

Exercise 2.8 *Show that energy is preserved during continuous evolution. What fraction of the energy of the ball is lost at each bounce? What is the*

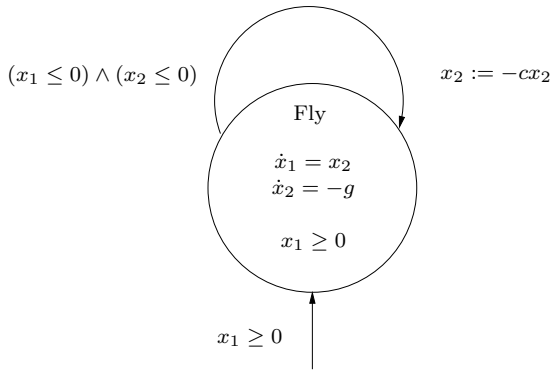

Figure 2.7. Bouncing ball

time interval that elapses between two bounces as a function of the energy of the ball?

Starting at an initial state with $x_1 \geq 0$ (as indicated by the logical condition next to the arrow pointing to the vertex), the continuous state flows according to the differential equation as long as the condition $x_1 \geq 0$ is fulfilled. When $x_1 = 0$ and $x_2 \leq 0$, a discrete transition takes place and the continuous state is reset to $x_2 := -cx_2$ (x_1 remains constant). Subsequently, the state resumes flowing according to the vector field, and so on. Such a trajectory is called an *execution* (and sometimes a *run* or a *solution*) of the hybrid system.

2.3.2 Thermostat: A Hybrid State System

Consider a room being heated by a radiator controlled by a thermostat. Assume that when the radiator is off the temperature, $x \in \mathbb{R}$, of the room decreases exponentially towards 0 degrees according to the differential equation

$$\dot{x} = -ax \tag{2.2}$$

for some $a > 0$.

Exercise 2.9 *Verify that the trajectories of* (2.2) *decrease to* 0 *exponentially.*

When the thermostat turns the heater on the temperature increases exponentially towards 30 degrees, according to the differential equation

$$\dot{x} = -a(x - 30). \tag{2.3}$$

Exercise 2.10 *Verify that the trajectories of* (2.3) *increase towards* 30 *exponentially.*

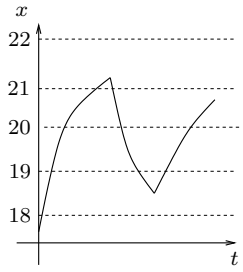

Figure 2.8. A trajectory of the thermostat system.

Assume that the thermostat is trying to keep the temperature at around 20 degrees. To avoid "chattering" (i.e. switching the radiator on an off all the time) the thermostat does not attempt to turn the heater on until the temperature falls below 19 degrees. Due to some uncertainty in the radiator dynamics, the temperature may fall further, to 18 degrees, before the room starts getting heated. Likewise, the thermostat does not attempt to turn the heater on until the temperature rises above 21 degrees. Due to some uncertainty in the radiator dynamics the temperature may rise further, to 22 degrees, before the room starts to cool down. A trajectory of the thermostat system is shown in Figure 2.8. Notice that in this case multiple trajectories may be obtained for the same initial conditions, as for certain values of the temperature there is a choice between switching the radiator on/off or not. Systems for which such a choice exists are known as *non-deterministic*.

Notice that this system has both a continuous and a discrete state. The continuous state is the temperature in the room $x \in \mathbb{R}$. The discrete state, $q \in \{ON, OFF\}$ reflects whether the radiator is on or off. The evolution of x is governed by a differential equation (as was the case with the pendulum), while the evolution of q is through jumps (as was the case with the manufacturing machine). The evolution of the two types of state is coupled. When $q = ON$, x rises according to differential equation (2.3), while when $q = OFF$, x decays according to differential equation (2.2). Likewise, q *can not jump* from ON to OFF unless $x \geq 21$. q *must jump* from ON to OFF if $x \geq 22$. Etc.

It is very convenient to compactly describe such *hybrid systems* by mixing the differential equation with the directed graph notation (Figure 2.9).

2.3.3 Gear Shift Control

The gear shift example describes a control design problem where both the continuous and the discrete controls need to be determined. Figure 2.10 shows a model of a car with a gear box having four gears.

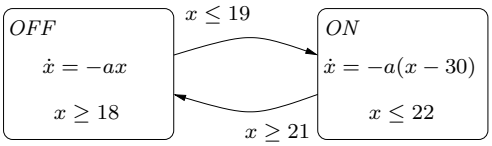

Figure 2.9. Directed graph notation for the thermostat system.

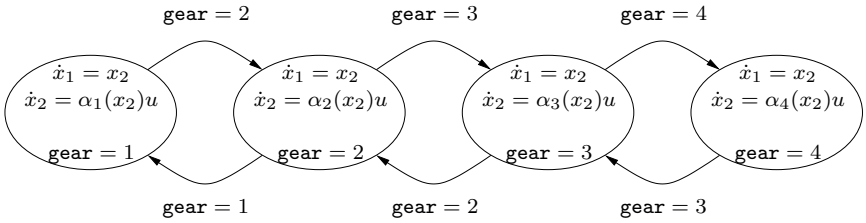

Figure 2.10. A hybrid system modeling a car with four gears.

The longitudinal position of the car along the road is denoted by x_1 and its velocity by x_2 (lateral dynamics are ignored). The model has two control signals: the gear denoted $\texttt{gear} \in \{1, \ldots, 4\}$ and the throttle position denoted $u \in [u_{\min}, u_{\max}]$. Gear shifting is necessary because little power can be generated by the engine at very low or very high engine speed. The function α_i represents the efficiency of gear i. Typical shapes of the functions α_i are shown in Figure 2.11.

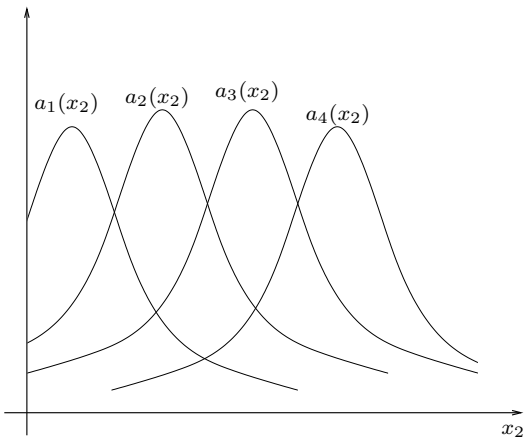

Figure 2.11. The efficiency functions of the different gears.

Exercise 2.11 *How many real valued continuous states does this model have? How many discrete states?*

Several interesting control problems can be posed for this simple car model. For example, what is the optimal control strategy to drive from $(a, 0)$ to $(b, 0)$ in minimum time? The problem is not trivial if we include the reasonable assumption that each gear shift takes a certain amount of time. The optimal controller, which will also be a hybrid system, may in general be derived using the theory of optimal control of hybrid systems.

2.3.4 Collision avoidance between airplanes (Claire)

2.3.5 Collision avoidance between cars

The need to ensure the safety of the vehicles on an automated highway system (AHS) dictates that formal methods have to be used to design and analyze the hybrid interactions.

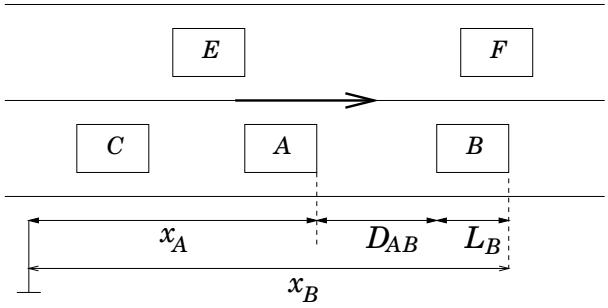

Figure 2.12. AHS model with five platoons and distances as marked.

Consider two vehicles, labeled A and B, moving on an AHS (Figure **??**) with A following B. Let L_i denote the length of platoon $i = A, B$, and x_i its position from a fixed road-side reference frame. Since neither the dynamics nor the safety requirements depend on the absolute position of the vehicles, we introduce a variable $D_{AB} = x_B - x_A - L_B$ to keep track of the spacing between A and B. We assume that (after feedback linearization) the controller of vehicle A can directly affect the acceleration of A, $u = \ddot{x}_A$ through brake and throttle actuators. We also assume that vehicle A is equipped with sensors to measure its own velocity and the spacing and relative velocity with respect to vehicle B. The acceleration of vehicle B, $\ddot{x}_B$, is assumed to be unknown to vehicle A and is treated as a disturbance.

The continuous dynamics can be described by a state vector $x = (x_1, x_2, x_3) = (\dot{x}_A, D_{AB}, \dot{D}_{AB}) \in \mathbb{R}^3$ with:

$$\dot{x} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} u + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \ddot{x}_B \tag{2.4}$$

Physical considerations impose constraints on u and $\ddot{x}_B$: vehicles are not allowed to move in reverse, and are required to keep their speed below a certain speed limit, v^{max}. To enforce these requirements we assume that u and $\ddot{x}_B$ satisfy

$$u \in \begin{cases} [0, a_A^{max}] & \text{if } x_1 \leq 0 \\ [a_A^{min}, a_A^{max}] & \text{if } 0 < x_1 < v^{max} \\ [a_A^{min}, 0] & \text{if } x_1 \geq v^{max} \end{cases}$$

$$\ddot{x}_B \in \begin{cases} [0, a_B^{max}] & \text{if } x_1 + x_3 \leq 0 \\ [a_B^{min}, a_B^{max}] & \text{if } 0 < x_1 + x_3 < v^{max} \\ [a_B^{min}, 0] & \text{if } x_1 + x_3 \geq v^{max} \end{cases}$$

To ensure that the dynamics of the system are physically meaningful, we assume that the set of initial states is such that $\dot{x}_A(0) = x_1(0) \in [0, v^{max}]$, $\dot{x}_B(0) = x_1(0) + x_3(0) \in [0, v^{max}]$, and that the constants satisfy $a_A^{min} < 0 < a_A^{max}$ and $a_B^{min} < 0 < a_B^{max}$. In this case $\dot{x}_A(t), \dot{x}_B(t) \in [0, v^{max}]$ for all $t \geq 0$.

Even though the model of this two vehicle system seems continuous at first, there are a number of sources of discrete behavior. The first is the mode switching necessary to enforce the constraints on the velocities. Three discrete states are needed for each vehicle to account for this, one for $\dot{x}_i = 0$, one for $\dot{x}_i \in (0, v^{max})$ and one for $\dot{x}_i = v^{max}$, $i = A, B$. This gives rise to a total of nine discrete states.

If A and B represent entire platoons, additional discrete phenomena are introduced by the intra-platoon collisions that A and B may experience in case of emergency. From the point of view of platoon A these collisions can be treated as a source of disturbance, and can be modeled as discrete events that instantaneously reset the velocities of certain vehicles. For simplicity, we assume that the first vehicle of platoon A (leader of A platoon) can experience at most one collision with the vehicle immediately behind it (first follower in platoon A), and parameterize the disturbance by the time at which the collision occurs (T_A) and the resulting increase in the velocity of the leader (δv_A). Likewise, we assume that the last vehicle of platoon B can experience at most one collision, and use the time at which the collision occurs (T_B) and the decrease of the velocity of the last vehicle (δv_B) to parameterize the disturbance. Since the vehicles are not allowed to move in reverse, we assume that collisions within a platoon will not result in negative velocities, or, in other words, that $\delta v_B \leq x_1(T_B) + x_3(T_B)$. Likewise, since vehicles are not allowed to move faster than the speed limit, it is natural to assume that collisions within a platoon will not result in

velocities greater than v^{max}, or, in other words, $\delta v_A \leq v^{max} - x_1(T_A)$. Finally, if the intra-platoon controllers are designed properly, we can assume that all intra-platoon collisions will be at low relative velocities, below a certain "safe" value, $v^S \approx 3m/s$. Under these assumptions, which are reasonable if the vehicles have roughly equal masses and coefficients of restitution, the discrete disturbance caused by intra-platoon collisions can be parameterized by

$$T_A \geq 0,\ \delta v_A \in [0, \min\{v^S, v^{max} - x_1(T_A)\}],$$
$$T_B \geq 0,\ \delta v_B \in [0, \min\{v^S, x_1(T_B) + x_3(T_B)\}]$$

The hybrid automaton used to capture the intra-platoon collisions in platoons A and B is shown in Figure 2.13. The discrete state is q_0 if no collisions have taken place in either platoon, q_1 is a collision takes place only in platoon B, q_2 is a collision takes place only in platoon A and q_3 if collisions take place in both platoons. Two discrete disturbance inputs ($Coll_A$ and $Coll_B$) are introduced to trigger the collisions, and four continuous disturbance inputs (T_A, T_B, δv_A, δv_B) are introduced to capture their effect. The discrete states introduced to model the velocity constraints have been suppressed to simplify the figure; with these states, the total number of discrete states is thirty-six. To simplify the notation we use $d = (\ddot{x}_B, T_A, T_B, \delta v_A, \delta v_B)$ to denote the continuous disturbance inputs.

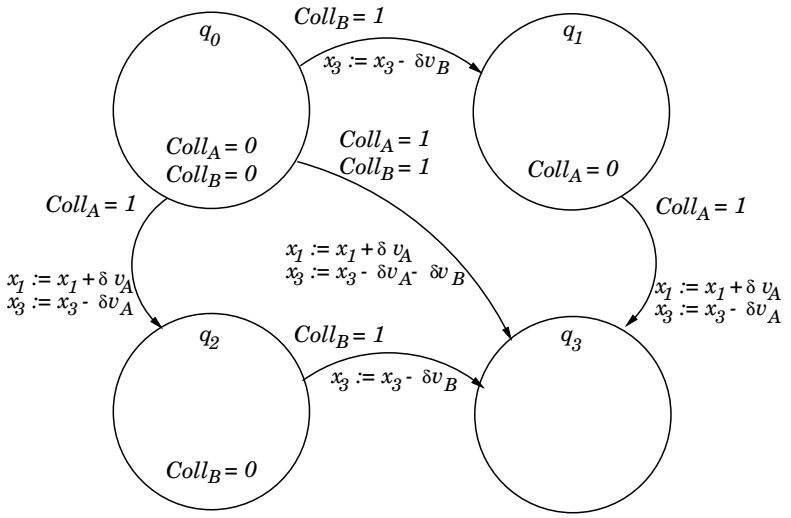

Figure 2.13. Hybrid automaton modeling intra-platoon collisions in platoons A and B. The discrete states are q_0 for no collisions, q_1 for collision inside platoon B, q_2 for collision inside platoon A, and q_3 for simultaneous intra-platoon collisions in A and B. The continuous dynamics within each discrete mode are given by (2.4).

Even though intra-platoon collisions within platoons A and B may be acceptable in case of emergency, inter-platoon collisions should be avoided at all costs. Thus, for safety, we would like to prevent collisions between platoons A and B. In other words, we would like to ensure that $x_2 \geq 0$ at all times. Notice that the limiting case $x_2(t) = 0$ is considered acceptable, since the vehicles just touch with zero relative velocity. This safety requirement can be posed as a controller synthesis problem of selecting the continuous control variable u such that for all actions of the disturbance d the two platoons are guaranteed not to collide.

2.3.6 Computer-Controlled System

Hybrid systems are natural models for computer-controlled systems (Figure 2.14), since they involve a physical process (which often can be modeled as continuous-time system) and a computer (which is fundamentally a finite state machine). The classical approach to computer-controlled systems has been using sampled-data theory, where it is assumed that measurements and control actions are taken at a fixed sampling rate. Such a scheme is easily encoded using a hybrid model. The hybrid model also captures a more general formulation where measurements may also be taken asynchronously, based for example on computer interrupts. This is sometimes closer to real-time implementations, for example, in embedded control systems.

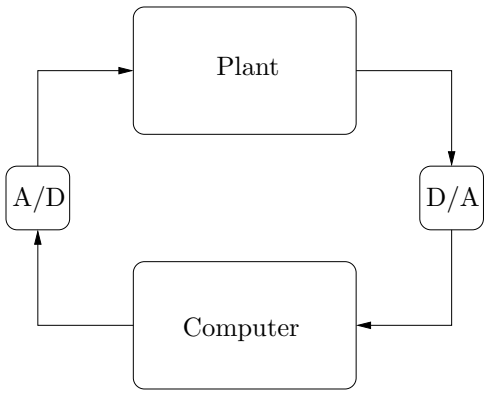

Figure 2.14. Computer-controlled system.

To model all the diverse phenomena in the above examples, one needs a modeling framework that is

- *descriptive*, to allow one to capture different types of continuous and discrete dynamics, be capable of modeling different ways in which dis-

crete evolution affects and is affected by continuous evolution, allow non-deterministic models (e.g. the thermostat example of Chapter 2) to capture uncertainty, etc.

- *composable*, to allow one to build large models by composing models of simple components (e.g. for the automated highways or air traffic applications).
- *abstractable*, to allow one to refine design problems for composite models down to design problems for individual components and, conversely, compose results about the performance of individual components to study the performance for the overall system.

Modeling frameworks that possess at least some subset of these properties have been developed in the hybrid systems literature. Different frameworks place more emphasis on different aspects, depending on the applications and problems they are designed to address. Here we will concentrate on one such framework, called *hybrid automata*. The hybrid automata we will study are fairly rich (in terms of descriptive power), but are autonomous, i.e. have no inputs and outputs. They are therefore suitable for modeling autonomous systems and analyzing their properties (stability, reachability, etc.), but are unsuitable for modeling systems compositionally and carrying out abstraction operations and solving control problems (stabilization, optimal control, safe controller synthesis, etc.).

Chapter 3
Hybrid Automata and Executions

Our goal is to develop a mathematical representation of such systems as described in the previous chapters. The representation should be compact, yet rich enough to describe both the evolution of continuous dynamics as well as the modal logic.

In this section we present a model for a *hybrid automaton*. The model is called *hybrid* because it combines nonlinear continuous dynamics with the dynamics of discrete event systems. In this chapter, we focus on autonomous hybrid systems, and in subsequent chapters we treat the case of hybrid automata with control and disturbance input variables, both continuous and discrete. Our model is based on the hybrid system model of [**?**], developed further in [36] and [**?**].

As background, we first present a model for a discrete event system, and then one for a purely continuous nonlinear system. We describe the state and input spaces, the control and environment variables, system trajectories and safety properties. We then present a model for a nonlinear hybrid automaton.

3.1 Discrete, Continuous, and Hybrid Models

3.1.1 Finite State Automata

A finite state automaton is represented as

$$(Q, \mathrm{I}nit, \delta) \tag{3.1}$$

where $Q = \{q_1, q_2, \ldots, q_m\}$ is a finite set of *discrete states*; $\delta : Q \to 2^Q$ is a *partial transition relation*, and $Init \subseteq Q$ is a set of *initial states*. Note that the behavior of the finite state automaton is *non-deterministic*: the transition function $\delta(q)$ represents a set of possible new states, rather than a single unique state. Transitions may be prevented, or blocked, from occurring at state q by setting $\delta(q) = \emptyset$.

A *system trajectory* $q[\cdot] \in Q^\omega$ is a finite or infinite sequence of states and actions which satisfies, for $i \in \mathbb{Z}$,

$$q[0] \in Init \text{ and } q[i+1] \in \delta(q[i]) \tag{3.2}$$

3.1.2 Continuous-Time Dynamics

Here, we consider continuous dynamics described by the nonlinear ordinary differential equation (ODE)

$$\dot{x}(t) = f(x(t)), \quad x(0) \in Init \tag{3.3}$$

where $x \in X$ is the finite-dimensional *state* in an n-manifold (frequently $X = \mathbb{R}^n$), f is a smooth vector field over $\mathbb{R}^n$, and $Init \subseteq X$ is a set of *initial conditions*.

A *system trajectory* over an interval $[\tau, \tau'] \subseteq \mathbb{R}$ is a map:

$$x(\cdot) : [\tau, \tau'] \to X \tag{3.4}$$

such that $x(\cdot)$ is continuous, and for all $t \in [\tau, \tau']$, $\dot{x}(t) = f(x(t))$. We assume that the function f is *globally Lipschitz* in x. Then, by the existence and uniqueness theorem of solutions for ordinary differential equations, given an interval $[\tau, \tau']$, the value of $x(t)$ for some $t \in [\tau, \tau']$ there exists a unique solution $x(\cdot)$ to (3.3).

3.1.3 Hybrid Automata

A hybrid automaton is a dynamical system that describes the evolution in time of the values of a set of discrete and continuous state variables.

Definition 3.1 (Hybrid Automaton) *A hybrid automaton H is a collection $H = (Q, X, f, Init, Dom, E, G, R)$, where*

- $Q = \{q_1, q_2, \ldots\}$ *is a set of* **discrete states***;*
- $X = \mathbb{R}^n$ *is a set of* **continuous states***;*
- $f(\cdot, \cdot) : Q \times X \to \mathbb{R}^n$ *is a* **vector field***;*
- $Init \subseteq Q \times X$ *is a set of* **initial states***;*
- $Dom(\cdot) : Q \to 2^X$ *is a* **domain***;*
- $E \subseteq Q \times Q$ *is a set of* **edges***;*

- $G(\cdot) : E \to 2^X$ *is a* **guard condition**;
- $R(\cdot, \cdot) : E \times X \to 2^X$ *is a* **reset map**.

Recall that 2^X denotes the power set (set of all subsets) of X. The notation of Definition 3.1 suggests, for example, that the function Dom assigns a set of continuous states D$om(q) \subseteq \mathbb{R}^n$ to to each discrete state $q \in Q$. We refer to $(q, x) \in Q \times X$ as the *state* of H. To eliminate technical difficulties we impose the following assumption.

Assumption 3.2 *For all $q \in Q$, $f(q, \cdot)$ is Lipschitz continuous. For all $e \in E$, $G(e) \neq \emptyset$, and for all $x \in G(e)$, $R(e, x) \neq \emptyset$.*

Lipschitz continuity is needed to ensure the existence and uniqueness of solutions for the continuous dynamics. The second part of the assumption eliminates some pathological cases and can in fact be imposed without loss of generality (see Problem 3.6).

Hybrid automata define possible evolutions for their state. Roughly speaking, starting from an initial value $(q_0, x_0) \in I\mathit{nit}$, the continuous state x flows according to the differential equation

$$\begin{aligned} \dot{x} &= f(q_0, x), \\ x(0) &= x_0, \end{aligned}$$

while the discrete state q remains constant

$$q(t) = q_0.$$

Continuous evolution can go on as long as x remains in D$om(q_0)$. If at some point the continuous state x reaches the guard $G(q_0, q_1) \subseteq \mathbb{R}^n$ of some edge $(q_0, q_1) \in E$, the discrete state may change value to q_1. At the same time the continuous state gets reset to some value in $R(q_0, q_1, x) \subseteq \mathbb{R}^n$. After this discrete transition, continuous evolution resumes and the whole process is repeated.

As we saw in Chapter 2 and in the examples of Section 2.3, it is often convenient to visualize hybrid automata as directed graphs (Q, E) with vertices Q and edges E. With each vertex $q \in Q$, we associate a set of initial states $\{x \in X \mid (q, x) \in I\mathit{nit}\}$, a vector field $f(q, \cdot) : X \to \mathbb{R}^n$ and a domain D$om(q) \subseteq X$. An edge $(q, q') \in E$ starts at $q \in Q$ and ends at $q' \in Q$. With each edge $(q, q') \in E$, we associate a guard $G(q, q') \subseteq X$ and a reset function $R(q, q', \cdot) : X \to 2^X$.

Example (Water Tank System) The two tank system, shown in Figure 3.1, consists of two tanks containing water. Both tanks are leaking at a constant rate. Water is added to the system at a constant rate through a hose, which at any point in time is dedicated to either one tank or the other. It is assumed that the hose can switch between the tanks instantaneously.

For $i = 1, 2$, let x_i denote the volume of water in Tank i and $v_i > 0$ denote the constant flow of water out of Tank i. Let w denote the constant

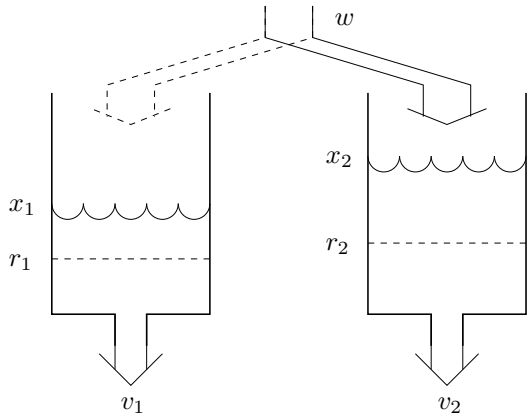

Figure 3.1. The water tank system.

flow of water into the system. The objective is to keep the water volumes above r_1 and r_2, respectively, assuming that the water volumes are above r_1 and r_2 initially. This is to be achieved by a controller that switches the inflow to Tank 1 whenever $x_1 \leq r_1$ and to Tank 2 whenever $x_2 \leq r_2$.

It is straight forward to define a hybrid automaton, to describe this process. The automaton will have:

- $Q = \{q_1, q_2\}$, two discrete states, corresponding to inflow going left and inflow going right.
- $X = \mathbb{R}^2$, two continuous states, the levels of water in the two tanks.
- A vector field
$$f(q_1, x) = \begin{bmatrix} w - v_1 \\ -v_2 \end{bmatrix}, \text{ and } f(q_2, x) = \begin{bmatrix} -v_1 \\ w - v_2 \end{bmatrix};$$
such that when the inflow is going to the tank on the right, the water level in the left tank goes down while the water level in right tank goes up, and vice versa.
- Initial states, $Init = \{q_1, q_2\} \times \{x \in \mathbb{R}^2 \mid x_1 \geq r_1 \wedge x_2 \geq r_2\}$, i.e. start with both water levels above the low level marks r_1 and r_2.
- Domains $Dom(q_1) = \{x \in \mathbb{R}^2 \mid x_2 \geq r_2\}$ and $Dom(q_2) = \{x \in \mathbb{R}^2 \mid x_1 \geq r_1\}$ reflecting the fact that we put water in the current tank as long as the level in the other tank is above the low level mark.
- Edges $E = \{(q_1, q_2), (q_2, q_1)\}$, reflecting the fact that it is possible to switch inflow from left to right and vice versa.
- Guards $G(q_1, q_2) = \{x \in \mathbb{R}^2 \mid x_2 \leq r_2\}$ and $G(q_2, q_1) = \{x \in \mathbb{R}^2 \mid x_1 \leq r_1\}$ to allow us to switch the inflow to the other tank as soon as the water there reaches the low level mark.

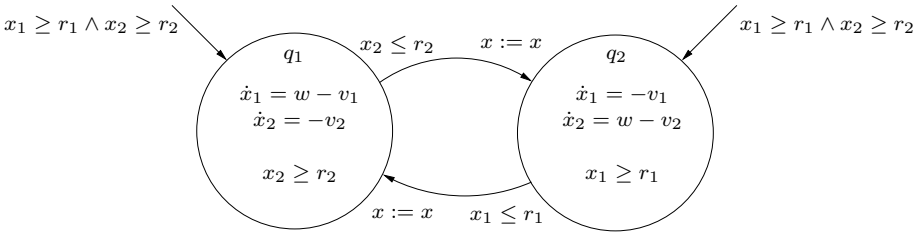

Figure 3.2. Graphical representation of the water tank hybrid automaton.

- Reset relation equal to the identity map for x, $R(q_1, q_2, x) = R(q_2, q_1, x) = \{x\}$, since the continuous state does not change as a result of switching the inflow.

The directed graph corresponding to this hybrid automaton is shown in Figure 3.2. ■

Notice that the directed graphs contain exactly the same information as Definition 3.1. They can therefore be treated as informal definitions of hybrid automata. It is common to remove the assignment $x := x$ from an edge of the graph when the continuous state does not change as a result of the discrete transition corresponding to that edge.

3.1.4 Hybrid Time Sets & Executions

To define the "solutions" of a hybrid automaton we draw an analogy to the definition for the solutions of an ordinary differential equation. Recall that the solution of the differential equation $\dot{x} = f(x)$ with initial condition x_0 is "a function $x(\cdot) : [0, T] \to \mathbb{R}^n$ such that

$$x(0) = x_0$$
$$\dot{x}(t) = f(x(t)), \forall t \in [0, T]."$$

To develop an analogous definition for hybrid automata we clearly need to generalize the space in which our function takes its values: Since our system now has both continuous and discrete state, the solution has to be a "function" from "time" to the state space $Q \times X$.

This is not enough however. Hybrid automata involve both continuous flow (determined by differential equations) and discrete jumps (determined by a directed graph). The trajectories of the differential equations evolve in continuous (real valued) time, whereas the trajectories of automata evolve effectively in discrete (integer valued) time. To characterize the evolution of the state of a hybrid automaton one therefore has to think of a more general set of times that involves both continuous intervals over which continuous evolution takes place and distinguished discrete points when discrete transitions happen. Such a set of times is called a *hybrid time set*.

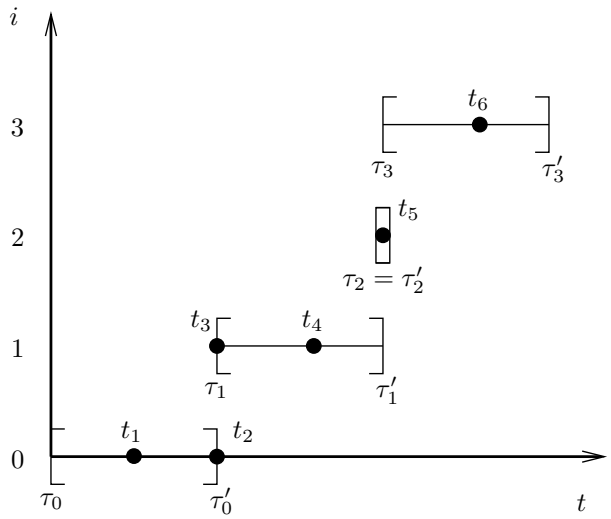

Figure 3.3. A hybrid time set $\tau = \{[\tau_i, \tau_i']\}_{i=0}^{3}$.

Definition 3.3 (Hybrid Time Set) *A hybrid time set is a sequence of intervals $\tau = \{I_0, I_1, \ldots, I_N\} = \{I_i\}_{i=0}^{N}$, finite or infinite (i.e. $N < \infty$ or $N = \infty$) such that*

- *$I_i = [\tau_i, \tau_i']$ for all $i < N$;*
- *if $N < \infty$ then either $I_N = [\tau_N, \tau_N']$ or $I_N = [\tau_N, \tau_N')$, possibly with $\tau_N' = \infty$; and*
- *$\tau_i \leq \tau_i' = \tau_{i+1}$ for all i.*

Since all the primitives in Definition 3.1 do not depend explicitly on time, we can assume without loss of generality that $\tau_0 = 0$. An example of a hybrid time set is given in Figure 3.3. Notice that the right endpoint, τ_i', of the interval I_i coincides with the left endpoint, τ_{i+1} of the interval I_{i+1} (c.f. the time instants labeled t_2 and t_3 in Figure 3.3). The interpretation is that these are the times at which discrete transitions of the hybrid system take place. τ_i' corresponds to the time instant just before a discrete transition, whereas τ_{i+1} corresponds to the time instant just after the discrete transition. Discrete transitions are assumed to be instantaneous, therefore $\tau_i' = \tau_{i+1}$. The advantage of this convention is that it allows one to model situations where multiple discrete transitions take place one after the other at the same time instant, in which case $\tau_{i-1}' = \tau_i = \tau_i' = \tau_{i+1}$ (c.f. the interval $I_2 = [\tau_2, \tau_2']$ in Figure 3.3).

Despite its somewhat complicated nature, a hybrid time set, τ, is a rather well behaved mathematical object. For example, there is a natural way in which the elements of the hybrid time set can be ordered. For $t_1 \in [\tau_i, \tau_i'] \in \tau$ and $t_2 \in [\tau_j, \tau_j'] \in \tau$ we say that t_1 *precedes* t_2 (denoted by $t_1 \prec t_2$)

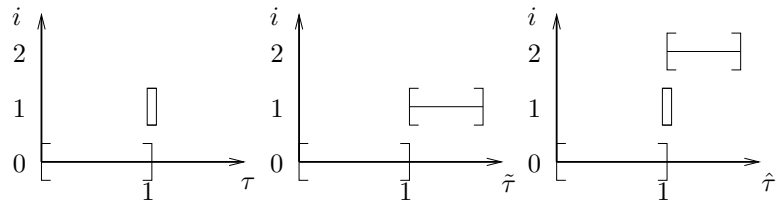

Figure 3.4. $\tau \sqsubset \hat{\tau}$ and $\tau \sqsubset \tilde{\tau}$.

if $t_1 < t_2$ (i.e. if the real number t_1 is less that the real number t_2) or if $i < j$ (i.e. if t_1 belongs to an earlier interval than t_2). In Figure 3.3, we have $t_1 \prec t_2 \prec t_3 \prec t_4 \prec t_5 \prec t_6$. In general, given any two distinct time instants, t_1 and t_2, belonging to some τ we have that either $t_1 \prec t_2$ or $t_2 \prec t_1$ (c.f. given any two distinct real numbers x and y, either $x < y$ or $y < x$). Using mathematical terminology, one would say that each hybrid time set τ is linearly ordered by the relation $\prec$.

Given two hybrid time sets τ and $\hat{\tau}$ there is also a natural way to define if one is "shorter" than the other (τ is called a *prefix* of $\hat{\tau}$ if it is "shorter"). More formally, we say that $\tau = \{I_i\}_{i=0}^{N}$ is a prefix of $\hat{\tau} = \{\hat{I}_i\}_{i=0}^{M}$ (and write $\tau \sqsubseteq \hat{\tau}$) if either they are identical, or τ is a finite sequence, $N \leq M$ (notice that M can be infinite), $I_i = \hat{I}_i$ for all $i = 0, \ldots, N-1$, and $I_N \subseteq \hat{I}_N$. We say that τ is a *strict prefix* of $\hat{\tau}$ (and write $\tau \sqsubset \hat{\tau}$) if $\tau \sqsubseteq \hat{\tau}$ and $\tau \neq \hat{\tau}$. In Figure 3.4, τ is a strict prefix of both $\hat{\tau}$ and $\tilde{\tau}$, but $\hat{\tau}$ is not a prefix of $\tilde{\tau}$ and $\tilde{\tau}$ is not a prefix of $\hat{\tau}$. Notice that given τ and $\hat{\tau}$ we may have neither $\hat{\tau} \sqsubseteq \tau$ nor $\tau \sqsubseteq \hat{\tau}$ (c.f. given two sets of real numbers $A \subseteq \mathbb{R}$ and $B \subseteq \mathbb{R}$ it is possible to have neither $A \subseteq B$ nor $B \subseteq A$). Using mathematical terminology, one would say that the set of all hybrid time sets is partially ordered by the relation $\sqsubseteq$.

Having generalized our notion of time from continuous/discrete time to the hybrid time sets implies that we also need to generalize the notion of a "function" from the time set to the state space.

Definition 3.4 (Hybrid Trajectory) *A hybrid trajectory over a set A is a pair (τ, a) consisting of a hybrid time set $\tau = \{I_i\}_0^N$ and a sequence of functions $a = \{a_i(\cdot)\}_0^N$ with $a_i(\cdot) : I_i \to A$.*

The notions of prefix and strict prefix naturally extend to hybrid trajectories. Given two hybrid trajectories (τ, a) and $(\hat{\tau}, \hat{a})$ we say that (τ, a) is a *prefix* of $(\hat{\tau}, \hat{a})$ (and write $(\tau, a) \sqsubseteq (\hat{\tau}, \hat{a})$) if $\tau \sqsubseteq \hat{\tau}$ and $a_i(t) = \hat{a}_i(t)$ for all $t \in I_i \in \tau$. We say that (τ, a) is a *strict prefix* of $(\hat{\tau}, \hat{a})$ (and write $(\tau, a) \sqsubseteq (\hat{\tau}, \hat{a})$) if $(\tau, a) \sqsubseteq (\hat{\tau}, \hat{a})$ and $\tau \sqsubset \hat{\tau}$. The relation $\sqsubseteq$ also defines a partial order on the space of hybrid trajectories.

Comparing with the definition of a solution of a differential equation $\dot{x} = f(x)$ with initial condition x_0 we see that all this work was needed just to generalize the sentence "a function $x(\cdot) : [0, T] \to \mathbb{R}^n$" to the hybrid domain. A "solution" (known as an *execution*) of an autonomous hybrid

automaton will be a hybrid trajectory, (τ, q, x) over its state space $Q \times X$. To complete the definition we now need to provide a generalization for the sentence: "such that

$$x(0) = x_0$$
$$\dot{x}(t) = f(x(t)), \forall t \in [0, T]."$$

The elements listed in Definition 3.1 impose restrictions on the types of hybrid trajectories that the hybrid automaton finds "acceptable", just like the initial condition x_0 and the vector field $f(\cdot)$ determine which functions $x(\cdot) : [0, T] \to \mathbb{R}^n$ are solutions of the differential equation and which are not.

Definition 3.5 (Execution) *An execution of a hybrid automaton H is a hybrid trajectory, (τ, q, x), which satisfies the following conditions:*

- Initial condition*:* $(q_0(\tau_0), x_0(\tau_0)) \in \mathrm{I}nit$.
- Discrete evolution*: for all i,* $(q_i(\tau_i'), q_{i+1}(\tau_{i+1})) \in E$, $x_i(\tau_i') \in G(q_i(\tau_i'), q_{i+1}(\tau_{i+1}))$, *and* $x_{i+1}(\tau_{i+1}) \in R(q_i(\tau_i'), q_{i+1}(\tau_{i+1}), x_i(\tau_i'))$.
- Continuous evolution*: for all i with $\tau_i < \tau_i'$,*
  1. $q_i(t) = q_i(\tau_i)$ *for all* $t \in I_i$*;*
  2. $x_i(\cdot) : I_i \to X$ *is the solution to the differential equation*
  $$\frac{dx_i}{dt}(t) = f(q_i(t), x_i(t))$$
  over I_i starting at $x_i(\tau_i)$; and,
  3. *for all* $t \in [\tau_i, \tau_i')$, $x_i(t) \in \mathrm{D}om(q_i(t))$.

Definition 3.5 specifies which of the hybrid trajectories are executions of H and which are not by imposing a number of restrictions. The first restriction dictates that the executions should start at an acceptable initial state in Init. For simplicity, we will use $(q_0, x_0) = (q_0(\tau_0), x_0(\tau_0)) \in \mathrm{I}nit$ to denote the initial state of an execution (τ, q, x). The second restriction determines when discrete transitions can take place and what the state after discrete transitions can be. The requirements relate the state before the discrete transition $(q_i(\tau_i'), x_i(\tau_i'))$ to the state after the discrete transition $(q_{i+1}(\tau_{i+1}), x_{i+1}(\tau_{i+1}))$: they should be such that $(q_i(\tau_i'), q_{i+1}(\tau_{i+1}))$ is an edge of the graph, $x_i(\tau_i')$ belongs to the guard of this edge and $x_{i+1}(\tau_{i+1})$ belongs the the reset map of this edge. In this context, it is convenient to think of the guard $G(e)$ as *enabling* a discrete transition $e \in E$: the execution *may* take a discrete transition $e \in E$ from a state x as long as $x \in G(e)$. The third restriction determines what happens along continuous evolution, and when continuous evolution must give way to a discrete transition. The first part dictates that along continuous evolution the discrete state remains constant. The second part requires that along continuous evolution the continuous state flows according to the differential equation

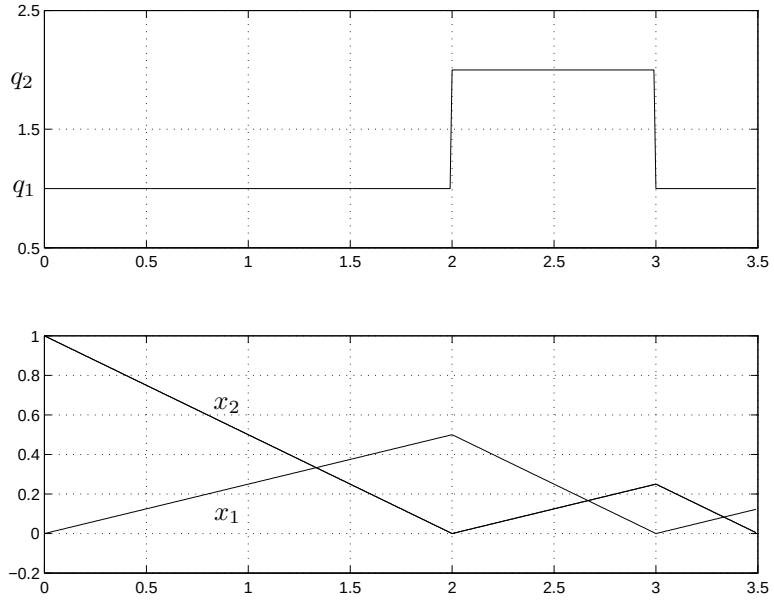

Figure 3.5. Example of an execution of the water tank hybrid automaton.

$\dot{x} = f(q, x)$. Notice that the differential equation depends on the discrete state we are currently in (which is constant along continuous evolution). The third part requires that along continuous evolution the state must remain in the domain, D$om(q)$, of the discrete state. In this context, it is convenient to think of D$om(q)$ as *forcing* discrete transitions: the execution *must* take a transition if the state is about to leave the domain.

Example (Water Tank (cont.)) Figure 3.5 shows an execution of the water tank automaton. The hybrid time set τ of the execution consists of three intervals, $\tau = \{[0, 2], [2, 3], [3, 3.5]\}$. The evolution of the discrete state is shown in the upper plot, and the evolution of the continuous state is shown in the lower plot. The values chosen for the constants are $r_1 = r_2 = 0$, $v_1 = v_2 = 1/2$ and $w = 3/4$. The initial state is $q = q_1$, $x_1 = 0$, $x_2 = 1$. ■

A convenient interpretation is that the hybrid automaton *accepts* (as opposed to generates) executions. This perspective allows one to consider, for example, hybrid automata that accept multiple executions for some initial states, a property that can prove very useful when modeling uncertain system (as illustrated by the thermostat example of Chapter 2).

Definition 3.6 (Classification of executions) *An execution (τ, q, x) is called:*

- **Finite**, *if τ is a finite sequence and the last interval in τ is closed.*

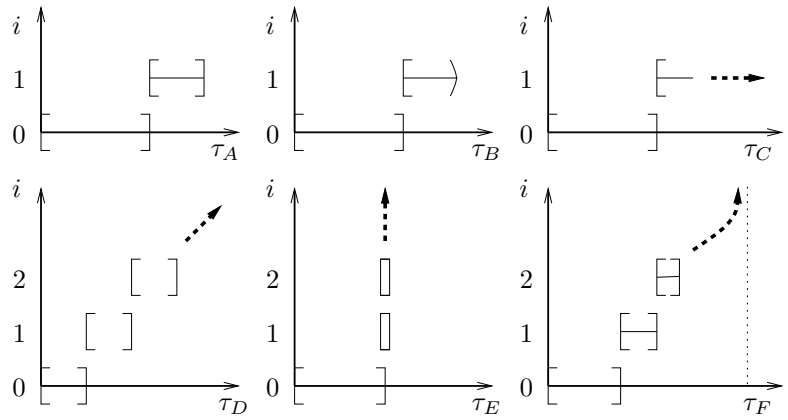

Figure 3.6. τ_A finite, τ_C and τ_D infinite, τ_E and τ_F Zeno.

- **Finite-open**, *if τ is a finite sequence ending with an interval $[\tau_N, \tau'_N)$ with $\tau'_N < \infty$.*
- **Infinite**, *if τ is an infinite sequence, or if the if it extends over an infinite time horizon,*

$$\sum_{i=0}^{N} (\tau'_i - \tau_i) = \infty.$$

- **Zeno**, *if it is infinite but $\sum_{i=0}^{\infty} (\tau'_i - \tau_i) < \infty$.*
- **Maximal** *if it is not a strict prefix of any other execution of H.*

Notice that by definition an infinite execution is also maximal. Figure 3.6 shows examples of hybrid time sets of finite, infinite and Zeno executions.

Exercise 3.1 *Show that an execution is Zeno if and only if it takes an infinite number of discrete transitions in a finite amount of time. Does an execution defined over the hybrid time set τ_B of Figure 3.6 belong to any of the classes of Definition 3.6?*

3.2 Existence and uniqueness

Powerful modeling frameworks, such as hybrid automata, allow one to model a very wide variety of physical phenomena, but also make it possible to produce models that are unreasonable, either physically or mathematically. A common danger in hybrid modeling is lack of existence of solutions. In most of the hybrid modeling frameworks one can easily construct models that admit no solutions for certain initial states. Such systems are known as *blocking* hybrid systems. This is an undesirable property when modeling

physical systems, since it suggests that the mathematical model provides an incomplete picture of the physical reality: the evolution of the physical system is likely to continue despite the fact that the evolution of the mathematical model is undefined.

Even if a hybrid system accepts executions for all initial states, it does not necessarily accept executions with infinite execution times. For example, the executions of hybrid models can take an infinite number of discrete transitions in finite time. Such executions are known as *Zeno* executions. One can argue that physical systems do not exhibit Zeno behavior. However, modeling abstraction can often lead to Zeno models of physical systems. Since abstraction is crucial for handling complex systems, understanding when it can lead to Zeno hybrid systems is important.

Another issue that arises in hybrid modeling is lack of uniqueness of solutions. Hybrid systems that accept multiple executions for a single initial state are known as *non-deterministic*. It is often desirable to retain some level of non-determinism in a hybrid model, since it can be used to model uncertainty (recall the thermostat example of Chapter 2). This, however, requires additional care when designing controllers for such systems, or when developing arguments about their performance. A common practice in continuous dynamical systems is to base proofs on arguments about *the solution* of the system. This is motivated by the fact that, under a Lipschitz continuity assumption, continuous dynamical systems have unique solutions. This proof technique is inadequate for non-deterministic systems. Instead one needs to develop arguments that hold for *all solutions* of the system.

Finally, hybrid systems are especially challenging from the point of view of simulation. The problems faced by the developers of simulation algorithms are intimately related to the modeling problems discussed so far.

- *Existence*: simulation algorithms may run into trouble if the simulated model has no solutions. Incorporating tests for existence in the simulation packages can alleviate this problem. More challenging is the case of Zeno executions. In this case, unless special care is taken, the simulation may grind to a halt, or produce spurious results.

- *Uniqueness*: Non-determinism introduces further complications from the point of view of simulation. Here the simulation algorithm may be called upon to decide between different alternatives. When a choice between continuous evolution and discrete transition is possible, a common approach is to take transitions the moment they are enabled (*as-soon-as* semantics). Probabilistic methods have also been proposed for dealing with non-determinism in the context of simulation, see Problem 3.9.

- *Discontinuity*: Lack of continuity of the solution with respect to initial conditions, an inherent characteristic of hybrid systems, can also lead to problems, both theoretical and practical. The most common problem is event detection (guard crossing).
- *Composability*: When simulating large scale systems (e.g. the Automated Highway System discussed in Chapter 2), one would like to be able to build up the simulation by composing different components (e.g. models for the motion of each vehicle). It may also be desirable to add components to the simulation on-line (e.g. to model vehicles joining the highway), eliminate components (e.g. to model vehicles leaving the highway), or redefine the interactions between components (e.g. to model vehicles changing lanes). Object oriented modeling languages have been developed to address these needs.

3.2.1 Two Fundamental Concepts

Reachability is a fundamental concept in the study of hybrid systems (and dynamical systems in general). Roughly speaking, a state, $(\hat{q}, \hat{x}) \in Q \times X$ of a hybrid automaton H is called reachable if the hybrid automaton can find its way to $(\hat{q}, \hat{x})$ 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. Reachability will also turn out to be a key concept in the study of safety properties for hybrid systems.

More formally,

Definition 3.7 (Reachable State) *A state $(\hat{q}, \hat{x}) \in Q \times X$ of a hybrid automaton H is called reachable if there exists a finite execution (τ, q, x) ending in $(\hat{q}, \hat{x})$, i.e. $\tau = \{[\tau_i, \tau_i']\}_0^N$, $N < \infty$, and $(q_N(\tau_N'), x_N(\tau_N')) = (\hat{q}, \hat{x})$.*

We will use R*each* $\subseteq Q \times X$ to denote the set of all states reachable by H. Clearly, I*nit* $\subseteq$ R*each*.

Exercise 3.2 *Why are all initial states reachable?*

Another important concept in the study of existence of executions for hybrid automata is the set of states from which continuous evolution is impossible. We will call these states *transition states*. For $(\hat{q}, \hat{x}) \in Q \times X$ and some $\epsilon > 0$, consider the solution, $x(\cdot) : [0, \epsilon) \to \mathbb{R}^n$ of the differential equation

$$\frac{dx}{dt} = f(\hat{q}, x) \text{ with } x(0) = \hat{x}. \tag{3.5}$$

Notice that, under the assumption that f is Lipschitz continuous in x, the solution to equation (3.5) exists and is unique (Theorem A.2). The states,

T*rans* $\subseteq Q \times X$, from which continuous evolution is impossible are

$$\text{T}rans = \{(\hat{q}, \hat{x}) \in Q \times X \mid \forall \epsilon > 0 \; \exists t \in [0, \epsilon) \text{ such that } x(t) \notin \text{D}om(\hat{q})\}\,.$$

In words, T*rans* is the set of states for which continuous evolution along the differential equation forces the system to exit the domain instantaneously.

Example (Water Tank (continued)) Consider again the water tank automaton, and assume that

$$0 < v_1, v_2 < w.$$

We will show how to compute the sets R*each* and T*rans* for this system.

First of all R*each* must contain all initial states. Therefore

$$\text{R}each \supseteq \{q_1, q_2\} \times \{x \in \mathbb{R}^2 \mid (x_1 \geq r_1) \wedge (x_2 \geq r_2)\} \tag{3.6}$$

Can it contain any other states? It turns out that it can not. To see why, we will show using induction that the state remains in the set I*nit*.

Consider an arbitrary initial state $(\hat{q}, \hat{x}) \in \text{I}nit$ and an arbitrary execution (τ, q, x) starting at $(\hat{q}, \hat{x})$. The fact that $(\hat{q}, \hat{x}) \in \text{I}nit$ provides the base case for the induction argument. Assume that for some i, $(q_i(\tau_i), x_i(\tau_i)) \in \text{I}nit$, and consider the case where $q_i(\tau_i) = q_1$ (the case $q_i(\tau_i) = q_2$ is similar). If $\tau_i' > \tau_i$, then continuous evolution takes place from $(q_i(\tau_i), x_i(\tau_i))$. Along this evolution, the first component of the continuous state increases (because $q_i(\tau_i) = q_1$, therefore $\dot{x}_1 = w - v_1$ and $v_1 < w$). The second component of the continuous state, on the other hand, decreases, but remains above r_2. This is because, by the definition of an execution,

$$x_i(t) \in \text{D}om(q_1) = \{x \in \mathbb{R}^2 \mid x_2 \geq r_2\}$$

for all $t \in [\tau_i, \tau_i']$. Therefore, $(q_i(t), x_i(t))$ remains in I*nit* along continuous evolution.

If $\tau_i' = \infty$, or if $[\tau_i, \tau_i']$ is the last interval in τ we are done! Otherwise, a discrete transition takes place from $(q_i(\tau_i'), x_i(\tau_i'))$. But the reset relation, R, leaves x unaffected, therefore,

$$(q_{i+1}(\tau_{i+1}), x_{i+1}(\tau_{i+1})) \in \text{I}nit$$

The last statement is true even if $\tau_i = \tau_i'$.

Summarizing, if $(q_i(\tau_i), x_i(\tau_i)) \in \text{I}nit$, then $(q_i(t), x_i(t)) \in \text{I}nit$ for all $t \in [\tau_i, \tau_i']$. Moreover, $(q_{i+1}(\tau_{i+1}), x_{i+1}(\tau_{i+1})) \in \text{I}nit$. Therefore, by induction on i, $(q_i(t), x_i(t)) \in \text{I}nit$ for all i and all t and

$$\text{R}each \subseteq \{q_1, q_2\} \times \{x \in \mathbb{R}^2 \mid (x_1 \geq r_1) \wedge (x_2 \geq r_2)\} \tag{3.7}$$

Equations (3.6) and (3.7) together imply that

$$\text{R}each = \{q_1, q_2\} \times \{x \in \mathbb{R}^2 \mid (x_1 \geq r_1) \wedge (x_2 \geq r_2)\}$$

To establish the set T*rans* for the water tank system, notice that continuous evolution is impossible if $q = q_1$ and $x_2 < r_2$ (the inflow will get

immediately switched to tank 2) or if $q = q_2$ and $x_1 < r_1$. Therefore,

$$\mathrm{T}rans \supseteq \big(\{q_1\} \times \{x \in \mathbb{R}^2 \mid x_2 < r_2\}\big) \cup \big(\{q_2\} \times \{x \in \mathbb{R}^2 \mid x_1 < r_1\}\big)$$

On the other hand, continuous evolution is possible if $q = q_1$ and $x_2 > r_2$, or if $q = q_2$ and $x_1 > r_1$. Therefore

$$\mathrm{T}rans \subseteq \big(\{q_1\} \times \{x \in \mathbb{R}^2 \mid x_2 \leq r_2\}\big) \cup \big(\{q_2\} \times \{x \in \mathbb{R}^2 \mid x_1 \leq r_1\}\big)$$

How about if $q = q_1$ and $x_2 = r_2$? If continuous evolution was to take place from this state, x_2 would immediately go below r_2. This is because $q = q_1$ implies that $\dot{x}_2 = -v_2 < 0$ (recall that $v_2 > 0$). This, however, would imply that the state would leave the domain $\mathrm{D}om(q_1)$, which is impossible along continuous evolution. Therefore, continuous evolution is also impossible from states where $q = q_1$ and $x_2 = r_2$ (and, by a symmetric argument, states where $q = q_2$ and $x_1 = r_1$). Overall,

$$\mathrm{T}rans = \big(\{q_1\} \times \{x \in \mathbb{R}^2 \mid x_2 \leq r_2\}\big) \cup \big(\{q_2\} \times \{x \in \mathbb{R}^2 \mid x_1 \leq r_1\}\big) .$$

■

3.2.2 Computation of R*each and* T*rans*

The computation of the set R*each* is a problem that has attracted considerable attention. In some cases the computation can be done analytically, using induction arguments. This *deductive* approach, taken above for the water tank example, will be reviewed more generally in Chapter 4. In other cases it may be possible to compute the set of reachable states automatically using appropriate computational *model checking* programs. This is the case, for example, for a special class of hybrid automata known as timed automata, also discussed in Chapter 4. For more general systems, one may have to resort optimal control tools, mixed integer linear programming, viability theory, etc. An approach to reachability computations for hybrid systems based on optimal control and viability theory will be presented in Chapter **??**.

Important Note: The necessary and sufficient conditions for existence and uniqueness presented in this chapter involve the set of reachable states, R*each*. This is needed, however, only to make the conditions necessary. If all we are interested in is establishing that a hybrid automaton accepts an infinite executions for all initial states, or that infinite executions are unique, it suffices to show that the conditions of the lemmas hold for all states (as opposed to all *reachable* states)or indeed any other set of states that contains the reachable states. This can make our life considerably easier.

The exact characterization of the set T*rans* may also be quite involved in general. Since the computation of T*rans* only involves arguments about

the continuous evolution of the system, it can be performed separately for each value of the discrete state. For each $q \in Q$ we set

$$\mathrm{T}rans(q) = \{x \in X \mid (q, x) \in \mathrm{T}rans\}.$$

Clearly, continuous evolution is impossible for all states outside the domain (refer to Definition 3.5). Therefore, states in the complement of the domain of q (i.e. the set of all $x \notin \mathrm{D}om(q)$, denoted by $\mathrm{D}om(q)^c$) must belong to $\mathrm{T}rans(q)$:

$$\mathrm{D}om(q)^c \subseteq \mathrm{T}rans(q).$$

Likewise, if $\hat{x}$ is in the interior of $\mathrm{D}om(\hat{q})$ (the largest open set contained in $\mathrm{D}om(\hat{q})$, denoted by $\mathrm{D}om(\hat{q})^\circ$), then there exists a small $\epsilon > 0$ such that the solution to (3.5) remains in $\mathrm{D}om(\hat{q})$. Therefore

$$\mathrm{D}om(q)^c \subseteq \mathrm{T}rans(q) \subseteq [\mathrm{D}om(q)^\circ]^c.$$

If $\mathrm{D}om(q)$ is an open set, then $\mathrm{D}om(q)^\circ = \mathrm{D}om(q)$ and $\mathrm{T}rans(q) = \mathrm{D}om(q)^c$.

Proposition 3.8 *If* $\mathrm{D}om(q)$ *is an open set and* $f(q, \cdot)$ *is Lipschitz continuous, then* $\mathrm{T}rans(q) = \mathrm{D}om(q)^c$.

If $\mathrm{D}om(q)$ is a closed set (i.e. it contains its boundary), then $\mathrm{T}rans(q)$ may also contain pieces of the boundary of the domain. In general, computing these pieces exactly requires tools from non-smooth analysis and may not be easy (see Appendix C). For all the examples considered in this chapter, however, the domain $\mathrm{D}om(q)$ and the dynamics $f(q, \cdot)$ are smooth. In this case the computation of $\mathrm{T}rans(q)$ can be done using nonlinear control tools.

Assume that $\mathrm{D}om(q)$ can be written as the zero level set of a function $\sigma : X \to \mathbb{R}$, i.e..

$$\mathrm{D}om(q) = \{x \in X \mid \sigma(x) \geq 0\}$$

Notice that if σ is continuous, then $\mathrm{D}om(q)$ is a closed set.

Assuming f and σ are sufficiently differentiable in x, we inductively define the *Lie derivatives* of σ along f, $L_f^m\sigma : X \to \mathbb{R}$, for $m \in \mathbb{N}$ by $L_f^0\sigma(x) = \sigma(x)$ and, for $m > 0$,

$$L_f^m\sigma(x) = \left(\frac{\partial}{\partial x} L_f^{m-1}\sigma(x)\right) f(q, x).$$

The *point-wise relative degree* of σ with respect to f is defined as the function $n_{(\sigma,f)} : X \to \mathbb{N}$ given by

$$n_{(\sigma,f)}(x) = \inf\left\{m \in \mathbb{N} \mid L_f^m\sigma(x) \neq 0\right\},$$

where we assume that inf of the empty set is equal to ∞.

Exercise 3.3 *Show that* $n_{(\sigma,f)}(x) = 0$ *if and only if* $\sigma(x) \neq 0$. *Assuming that* σ *is a continuous function, what does this imply about* x*?*

Proposition 3.9 *Assume that* $\mathrm{Dom}(q) = \{x \in X \mid \sigma(x) \geq 0\}$ *and* $f(q,\cdot)$ *and* $\sigma(\cdot)$ *are both analytic. Then* $\mathrm{Trans}(q) = \{x \in X \mid L_f^{n_{(\sigma,f)}(x)}\sigma(x) < 0\}$.

Proof: Consider an arbitrary $\hat{x} \in X$. Since f is analytic in x, the solution $x(t)$ of (3.5) is analytic as a function of t. Since σ is analytic in x, $\sigma(x(t))$ is also analytic as a function of t. Consider the Taylor expansion of $\sigma(x(t))$ about $t = 0$. By analyticity, the series

$$\sigma(x(t)) = \sigma(\hat{x}) + L_f\sigma(\hat{x})t + L_f^2\sigma(\hat{x})\frac{t^2}{2!} + \ldots$$

converges locally.

If $n_{(\sigma,f)}(\hat{x}) < \infty$, the first non-zero term in the expansion, $L_f^{n_{(\sigma,f)}(\hat{x})}\sigma(\hat{x})$, dominates the sum for t small enough. Therefore, if $L_f^{n_{(\sigma,f)}(\hat{x})}\sigma(\hat{x}) < 0$, for all $\epsilon > 0$ there exists $t \in [0,\epsilon)$ such that $\sigma(x(t)) < 0$. Hence, $\{x \in X \mid L_f^{n_{(\sigma,f)}(x)}\sigma(x) < 0\} \subseteq \mathrm{Trans}(q)$.

If $n_{(\sigma,f)}(\hat{x}) < \infty$, but $L_f^{n_{(\sigma,f)}(\hat{x})}\sigma(\hat{x}) > 0$, then there exists $\epsilon > 0$ such that for all $t \in [0,\epsilon)$, $\sigma(x(t)) > 0$. Hence, $\hat{x} \notin \mathrm{Trans}$. Finally, if $n_{(\sigma,f)}(\hat{x}) = \infty$, the Taylor expansion is identically equal to zero and, locally, $\sigma(x(t)) = 0$. Therefore, $\mathrm{Trans} \subseteq \{x \in X \mid L_f^{n_{(\sigma,f)}(x)}\sigma(x) < 0\}$. ∎

Exercise 3.4 *For the water tank automaton all domains and vector fields are analytic. Compute the set* Trans *using Proposition 3.9.*

The analyticity requirement can be relaxed somewhat, since the only part where it is crucial is when $n_{(\sigma,f)}(\hat{x}) = \infty$. The condition $n_{(\sigma,f)}(\hat{x}) < \infty$ indicates that the vector field f is in a sense transverse to the boundary of the domain at $\hat{x}$.

Proposition 3.10 *Assume that* $\mathrm{Dom}(q) = \{x \in X \mid \sigma(x) \geq 0\}$. *If for some* $m > 1$, $\sigma(\cdot)$ *is* m *times differentiable,* $f(q,\cdot)$ *is* $m-1$ *times differentiable and* $n_{(\sigma,f)}(x) < m$ *for all* $x \in X$, *then* $\mathrm{Trans}(q) = \{x \in X \mid L_f^{n_{(\sigma,f)}(x)}\sigma(x) < 0\}$.

Exercise 3.5 *Prove Proposition 3.10. The proof is along the lines of that of Proposition 3.9, but some more care is needed with the Taylor series expansion.*

3.2.3 Local Existence and Uniqueness

Next, we turn our attention to questions of existence of executions. We give some conditions under which infinite executions exist for all initial states, and conditions under which these executions are unique.

Definition 3.11 (Non-Blocking and Deterministic) *A hybrid automaton* H *is called non-blocking if for all initial states* $(\hat{q},\hat{x}) \in \mathrm{Init}$ *there exists*

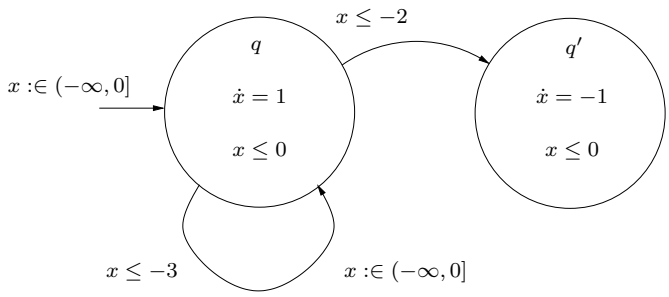

Figure 3.7. Examples of blocking and non-determinism.

an infinite execution starting at $(\hat{q}, \hat{x})$*. It is called deterministic if for all initial states* $(\hat{q}, \hat{x}) \in \mathrm{I}nit$ *there exists at most one maximal execution starting at* $(\hat{q}, \hat{x})$*.*

Roughly speaking, the non-blocking property implies that infinite executions exist for all initial states, while the deterministic property implies that the infinite executions (if they exist) are unique. Continuous dynamical systems described by differential equations have both these properties if the vector field f is assumed to be Lipschitz continuous (Theorem A.2). In hybrid systems, however, more things can go wrong.

Consider, for example, the hybrid automaton of Figure 3.7. Let $(\hat{q}, \hat{x})$ denote the initial state, and notice that $\hat{q} = q$. If $\hat{x} = -3$, executions starting at $(\hat{q}, \hat{x})$ can either flow along the vector field $\dot{x} = 1$, or jump back to q reseting x anywhere in $(-\infty, 0]$, or jump to q' leaving x unchanged. If $\hat{x} = -2$ executions starting at $(\hat{q}, \hat{x})$ can either flow along the vector field, or jump to q'. If $\hat{x} = -1$ executions stating at $(\hat{q}, \hat{x})$ can only flow along the vector field. Finally, if $\hat{x} = 0$ there are no executions starting at $(\hat{q}, \hat{x})$, other than the trivial execution defined over $[\tau_0, \tau_0']$ with $\tau_0 = \tau_0'$. Therefore, the hybrid automaton of Figure 3.7 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 3.12 *Consider a hybrid automaton satisfying Assumption 3.2. The hybrid automaton is non-blocking if for all* $(\hat{q}, \hat{x}) \in \mathrm{R}each \cap \mathrm{T}rans$*, there exists* $\hat{q}' \in Q$ *such that* $(\hat{q}, \hat{q}') \in E$ *and* $\hat{x} \in G(\hat{q}, \hat{q}')$*. If the hybrid automaton is deterministic, then it is non-blocking if and only if this condition holds.*

Proof: Consider an initial state $(q_0, x_0) \in \mathrm{I}nit$ and assume, for the sake of contradiction, that there does not exist an infinite execution starting at (q_0, x_0). Let $\chi = (\tau, q, x)$ denote a maximal execution starting at (q_0, x_0), and note that τ is a finite sequence.

Assume first that (τ, q, x) is finite open. Let $\tau = \{[\tau_i, \tau_i']\}_{i=0}^{N-1}[\tau_N, \tau_N')$ and

$$(q_N, x_N) = \lim_{t \to \tau_N'} (q_N(t), x_N(t)).$$

Note that, by the definition of execution and a standard existence argument for continuous dynamical systems, the limit exists and χ can be extended to a finite execution $\widehat{\chi} = (\widehat{\tau}, \widehat{q}, \widehat{x})$ with $\widehat{\tau} = \{[\tau_i, \tau_i']\}_{i=0}^{N}$, $\widehat{q}_N(\tau_N') = q_N$, and $\widehat{x}_N(\tau_N') = x_N$. This contradicts the assumption that χ is maximal.

Assume now (τ, q, x) is finite and let $\tau = \{[\tau_i, \tau_i']\}_{i=0}^{N}$ and $(q_N, x_N) = (q_N(\tau_N'), x_N(\tau_N'))$. Clearly, $(q_N, x_N) \in \mathrm{R}each$. If $(q_N, x_N) \notin \mathrm{T}rans$, then there exists $\epsilon > 0$ such that χ can be extended to $\widehat{\chi} = (\widehat{\tau}, \widehat{q}, \widehat{x})$ with $\widehat{\tau} = \{[\tau_i, \tau_i']\}_{i=0}^{N-1}[\tau_N, \tau_N' + \epsilon)$, by continuous evolution. If, on the other hand $(q_N, x_N) \in \mathrm{T}rans$, then by assumption there exists $(q', x') \in Q \times X$ such that $(q_N, q') \in E$, $x_N \in G(q_N, q')$ and $x' \in R(q_N, q', x_N)$. Therefore, χ can be extended to $\widehat{\chi} = (\widehat{\tau}, \widehat{q}, \widehat{x})$ with $\widehat{\tau} = \{[\tau_i, \tau_i']\}_{i=0}^{N+1}$, $\tau_{N+1} = \tau_{N+1}' = \tau_N'$, $q_{N+1}(\tau_{N+1}) = q'$, $x_{N+1}(\tau_{N+1}) = x'$ by a discrete transition. In both cases the assumption that χ is maximal is contradicted.

This argument also establishes the "if" of the second part. For the "only if", consider a deterministic hybrid automaton that violates the conditions, i.e., there exists $(q', x') \in \mathrm{R}each$ such that $(q', x') \in \mathrm{T}rans$, but there is no $\widehat{q}' \in Q$ with $(q', \widehat{q}') \in E$ and $x' \in G(q', \widehat{q}')$. Since $(q', x') \in \mathrm{R}each$, there exists $(q_0, x_0) \in \mathrm{I}nit$ and a finite execution, $\chi = (\tau, q, x)$ starting at (q_0, x_0) such that $\tau = \{[\tau_i, \tau_i']\}_{i=0}^{N}$ and $(q', x') = (q_N(\tau_N'), x_N(\tau_N'))$.

We first show that χ is maximal. Assume first that there exists $\widehat{\chi} = (\widehat{\tau}, \widehat{q}, \widehat{x})$ with $\widehat{\tau} = \{[\tau_i, \tau_i']\}_{i=0}^{N-1}\ [\tau_N, \tau_N' + \epsilon)$ for some $\epsilon > 0$. This would violate the assumption that $(q', x') \in \mathrm{T}rans$. Next assume that there exists $\widehat{\chi} = (\widehat{\tau}, \widehat{q}, \widehat{x})$ with $\widehat{\tau} = \tau[\tau_{N+1}, \tau_{N+1}']$ with $\tau_{N+1} = \tau_N'$. This requires that the execution can be extended beyond (q', x') by a discrete transition, i.e., there exists $(\widehat{q}', \widehat{x}') \in Q \times X$ such that $(q', \widehat{q}') \in E$, $x' \in G(q', \widehat{q}')$ and $\widehat{x}' \in R(q', \widehat{q}', x')$. This would contradict our original assumptions. Overall, χ is maximal.

Now assume, for the sake of contradiction that H is non-blocking. Then, there exists an infinite (and therefore maximal) χ' starting at (q_0, x_0). But $\chi \neq \chi'$ (as the former is finite and the latter infinite). This contradicts the assumption that H is deterministic. ∎

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 (recall that continuous evolution is unique by Theorem A.2). More specifically, the following lemma states that a hybrid automaton is deterministic if and only if (1) whenever a discrete transition is possible continuous evolution is impossible, and (2) discrete transitions have unique destinations.

Lemma 3.13 *Consider a hybrid automaton satisfying Assumption 3.2. The hybrid automaton is deterministic if and only if for all* $(\hat{q}, \hat{x}) \in \mathrm{R}each$*:*

1. *If* $\hat{x} \in G(\hat{q}, \hat{q}')$ *for some* $(\hat{q}, \hat{q}') \in E$*, then* $(\hat{q}, \hat{x}) \in \mathrm{T}rans$*.*
2. *If* $(\hat{q}, \hat{q}') \in E$ *and* $(\hat{q}, \hat{q}'') \in E$ *with* $\hat{q}' \neq \hat{q}''$ *then* $\hat{x} \notin G(\hat{q}, \hat{q}') \cap G(\hat{q}, \hat{q}'')$*.*
3. *If* $(\hat{q}, \hat{q}') \in E$ *and* $x \in G(\hat{q}, \hat{q}')$ *then* $R(\hat{q}, \hat{q}', \hat{x}) = \{\hat{x}'\}$*, i.e. the set contains a single element,* $\hat{x}'$*.*

Proof: For the "if" part, assume, for the sake of contradiction, that there exists an initial state $(q_0, x_0) \in \mathrm{I}nit$ and two maximal executions $\chi = (\tau, q, x)$ and $\widehat{\chi} = (\widehat{\tau}, \widehat{q}, \widehat{x})$ starting at (q_0, x_0) with $\chi \neq \widehat{\chi}$. Let $\bar{\chi} = (\bar{\tau}, \bar{q}, \bar{x})$ denote the maximal common prefix of χ and $\widehat{\chi}$. Such a prefix exists as the executions start at the same initial state. Moreover, $\bar{\chi}$ is not infinite, as $\chi \neq \widehat{\chi}$. As in the proof of Lemma 3.12, $\bar{\chi}$ can be assumed to be finite. Let $\bar{\tau} = \{[\bar{\tau}_i, \bar{\tau}_i']\}_{i=0}^{N}$ and $(q_N, x_N) = (q_N(\bar{\tau}_N'), x_N(\bar{\tau}_N')) = (\widehat{q}_N(\bar{\tau}_N'), \widehat{x}_N(\bar{\tau}_N'))$. Clearly, $(q_N, x_N) \in \mathrm{R}each$. We distinguish the following four cases:
Case 1: $\bar{\tau}_N' \notin \{\tau_i'\}$ and $\bar{\tau}_N' \notin \{\widehat{\tau}_i'\}$, i.e., $\bar{\tau}_N'$ is not a time when a discrete transition takes place in either χ or $\widehat{\chi}$. Then, by the definition of execution and a standard existence and uniqueness argument for continuous dynamical systems, there exists $\epsilon > 0$ such that the prefixes of χ and $\widehat{\chi}$ are defined over $\{[\bar{\tau}_i, \bar{\tau}_i']\}_{i=0}^{N-1}[\bar{\tau}_N, \bar{\tau}_N' + \epsilon)$ and are identical. This contradicts the fact that $\bar{\chi}$ is maximal.
Case 2: $\bar{\tau}_N' \in \{\tau_i'\}$ and $\bar{\tau}_N' \notin \{\widehat{\tau}_i'\}$, i.e., $\bar{\tau}_N'$ is a time when a discrete transition takes place in χ but not in $\widehat{\chi}$. The fact that a discrete transition takes place from (q_N, x_N) in χ indicates that there exists $q' \in Q$ such that $(q_N, q') \in E$ and $x_N \in G(q_N, q')$. The fact that no discrete transition takes place from (q_N, x_N) in $\widehat{\chi}$ indicates that there exists $\epsilon > 0$ such that $\widehat{\chi}$ is defined over $\{[\bar{\tau}_i, \bar{\tau}_i']\}_{i=0}^{N-1}[\bar{\tau}_N, \bar{\tau}_N' + \epsilon)$. A necessary condition for this is that $(q_N, x_N) \notin \mathrm{T}rans$. This contradicts condition 1 of the lemma.
Case 3: $\bar{\tau}_N' \notin \{\tau_i'\}$ and $\bar{\tau}_N' \in \{\widehat{\tau}_i'\}$, symmetric to Case 2.
Case 4: $\bar{\tau}_N' \in \{\tau_i'\}$ and $\bar{\tau}_N' \in \{\widehat{\tau}_i'\}$, i.e., $\bar{\tau}_N'$ is a time when a discrete transition takes place in both χ and $\widehat{\chi}$. The fact that a discrete transition takes place from (q_N, x_N) in both χ and $\widehat{\chi}$ indicates that there exist (q', x') and $(\widehat{q}', \widehat{x}')$ such that $(q_N, q') \in E$, $(q_N, \widehat{q}') \in E$, $x_N \in G(q_N, q')$, $x_N \in G(q_N, \widehat{q}')$, $x' \in R(q_N, q', x_N)$, and $\widehat{x}' \in R(q_N, \widehat{q}', x_N)$. Note that by condition 2 of the lemma, $q' = \widehat{q}'$, hence, by condition 3, $x' = \widehat{x}'$. Therefore, the prefixes of χ and $\widehat{\chi}$ are defined over $\{[\bar{\tau}_i, \bar{\tau}_i']\}_{i=0}^{N}[\bar{\tau}_{N+1}, \bar{\tau}_{N+1}']$, with $\bar{\tau}_{N+1} = \bar{\tau}_{N+1}' = \bar{\tau}_N'$, and are identical. This contradicts the fact that $\bar{\chi}$ is maximal and concludes the proof of the "if" part.

For the "only if" part, assume that there exists $(q', x') \in \mathrm{R}each$ such that at least one of the conditions of the lemma is violated. Since $(q', x') \in \mathrm{R}each$, there exists $(q_0, x_0) \in \mathrm{I}nit$ and a finite execution, $\chi = (\tau, q, x)$ starting at (q_0, x_0) such that $\tau = \{[\tau_i, \tau_i']\}_{i=0}^{N}$ and $(q', x') = (q_N(\tau_N'), x_N(\tau_N'))$. If condition 1 is violated, then there exist $\widehat{\chi}$ and $\tilde{\chi}$ with $\widehat{\tau} = \{[\tau_i, \tau_i']\}_{i=0}^{N-1}[\tau_N, \tau_N' + \epsilon)$, $\epsilon > 0$, and $\tilde{\tau} = \tau[\tau_{N+1}, \tau_{N+1}']$, $\tau_{N+1} = \tau_N'$,

such that $\chi \sqsubset \widehat{\chi}$ and $\chi \sqsubset \tilde{\chi}$. If condition 2 is violated, there exist $\widehat{\chi}$ and $\tilde{\chi}$ with $\widehat{\tau} = \tilde{\tau} = \tau[\tau_{N+1}, \tau'_{N+1}]$, $\tau_{N+1} = \tau'_{N+1} = \tau'_N$, and $\widehat{q}_{N+1}(\tau_{N+1}) \neq \tilde{q}_N(\tau_{N+1})$, such that $\chi \sqsubset \widehat{\chi}$, $\chi \sqsubset \tilde{\chi}$. Finally, if condition 3 is violated, then there exist $\widehat{\chi}$ and $\tilde{\chi}$ with $\widehat{\tau} = \tilde{\tau} = \tau[\tau_{N+1}, \tau'_{N+1}]$, $\tau_{N+1} = \tau'_{N+1} = \tau'_N$, and $\widehat{x}_{N+1}(\tau_{N+1}) \neq \tilde{x}_{N+1}(\tau_{N+1})$, such that $\chi \sqsubset \widehat{\chi}$, $\chi \sqsubset \tilde{\chi}$. In all three cases, let $\overline{\widehat{\chi}}$ and $\overline{\tilde{\chi}}$ denote maximal executions of which $\widehat{\chi}$ and $\tilde{\chi}$ are prefixes, respectively. Since $\widehat{\chi} \neq \tilde{\chi}$, it follows that $\overline{\widehat{\chi}} \neq \overline{\tilde{\chi}}$. Therefore, there are at least two maximal executions starting at (q_0, x_0) and thus H is non-deterministic. ■

The following theorem is a direct consequence of Lemmas 3.12 and 3.13.

Theorem 3.14 (Existence and Uniqueness) *Consider a hybrid automaton, such that $f(q, \cdot)$ is Lipschitz continuous for all $q \in Q$. The hybrid automaton accepts a unique infinite execution for all initial state if it satisfies all the conditions of Lemmas 3.12 and 3.13.*

Example (Water Tank (continued)) Consider again the water tank automaton with $0 < v_1, v_2 < w$. Recall that

$$\mathrm{R}each = \{(q, x) \in \mathbf{Q} \times \mathbb{R}^2 \mid x_1 \geq r_1 \wedge x_2 \geq r_2\},$$
$$\mathrm{T}rans = \{q_1\} \times \{x \in \mathbb{R}^2 \mid x_2 \leq r_2\} \cup \{q_2\} \times \{x \in \mathbb{R}^2 \mid x_1 \leq r_1\}.$$

Therefore,

$$\begin{aligned}\mathrm{R}each \cap \mathrm{T}rans = &\{q_1\} \times \{x \in \mathbb{R}^2 \mid x_1 \geq r_1 \wedge x_2 = r_2\} \cup \\ &\{q_2\} \times \{x \in \mathbb{R}^2 \mid x_2 \geq r_2 \wedge x_1 = r_1\}.\end{aligned}$$

Consider an arbitrary state $(\hat{q}, \hat{x}) \in \mathrm{R}each \cap \mathrm{T}rans$ (in fact the argument holds for any state $(\hat{q}, \hat{x}) \in \mathrm{T}rans$, see "important note" above). Notice that, if $\hat{q} = q_1$, then

$$\hat{x} \in \{x \in \mathbb{R}^2 \mid (x_1 \geq r_1) \wedge (x_2 = r_2)\} \subseteq G(q_1, q_2).$$

Likewise, if $q = q_2$, then $x \in G(q_1, q_2)$. Therefore, the condition of Lemma 3.12 is satisfied, and the water tank system is non-blocking.

Next, consider an arbitrary reachable state $(\hat{q}, \hat{x}) \in \mathrm{R}each$ (in fact the argument holds for any state $(\hat{q}, \hat{x}) \in Q \times X$). Assume that $\hat{q} = q_1$ (a similar argument holds if $\hat{q} = q_2$).

1. If $\hat{x} \in G(q_1, q_2) = \{x \in \mathbb{R}^2 \mid x_2 \leq r_2\}$, then $x_2 \leq r_2$. Therefore $(\hat{q}, \hat{x}) \in \mathrm{T}rans$.
2. Only one discrete transition is possible from q_1 (namely $(q_1, q_2) \in E$).
3. $R(q_1, q_2, \hat{x}) = \{\hat{x}\}$ contains one element.

Therefore, the conditions of Lemma 3.13 are also satisfied. By Theorem 3.14, the water tank automaton accepts a unique infinite execution for each initial state. ■

3.2.4 Zeno Executions

The conditions of Theorem 3.14 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. The Lipschitz assumption on f eliminates the possibility of escape to infinity in finite time along continuous evolution (c.f. finite escape example in Appendix A). However, the infinite executions may be such that the state takes an infinite number of discrete transitions in finite time. Executions with this property are known as Zeno executions.

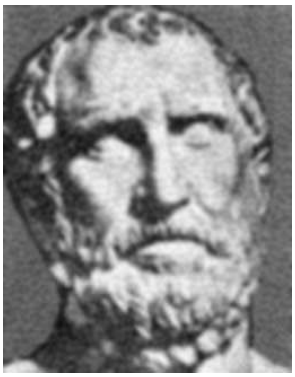

Figure 3.8. Zeno of Elea

The name "Zeno" comes from the philosopher, Zeno of Elea. Born around 490BC, Zeno (Figure 3.8) was one of the founders of the Eleatic school. He was a student of Parmenides, whose teachings rejected the ideas of plurality and transition as illusions generated by our senses. The best known contribution of Zeno to this line of thinking was a series of paradoxes designed to show 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. There are an infinite number of such segments and to cover each one of them Achilles needs a non zero amount of time. Therefore, Achilles will never overtake the turtle.

This paradox may seem simple minded, but it was not until the beginning of the 20^{th} century that it was resolved satisfactorily by mathematicians

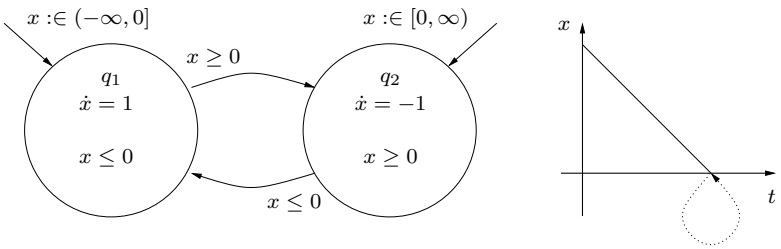

Figure 3.9. Chattering system.

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.

The Zeno phenomenon is notoriously difficult to characterize and eliminate in hybrid systems. Here we will not examine the properties of Zeno executions in detail. We will only give some examples of hybrid automata that admit Zeno behavior.

Example (Chattering System) Consider the hybrid automaton of Figure 3.9.

Exercise 3.6 *Write the system of Figure 3.9 in the notation of Definition 3.1. Show that it accepts a unique infinite execution for all initial conditions.*

It is easy to see that all infinite executions of this system are Zeno. An execution starting in x_0 at time 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.

This is a phenomenon known in continuous dynamical system as *chattering*. A bit of thought in fact reveals that this system is the same as the example used to demonstrate absence of solutions in Appendix A. In the control literature the "Zenoness" of such chattering systems is sometimes eliminated by allowing weaker solution concepts, such as sliding solutions (also known as Filippov solutions). ■

Example (Non-analytic Domain) Consider the hybrid automaton of Figure 3.10. Assume that the function $\rho : \mathbb{R} \to \mathbb{R}$ that determines the boundary of the domain is of the form

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

Exercise 3.7 *Write the system of Figure 3.10 in the notation of Definition 3.1. Show that it accepts a unique infinite execution for all initial states.*

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

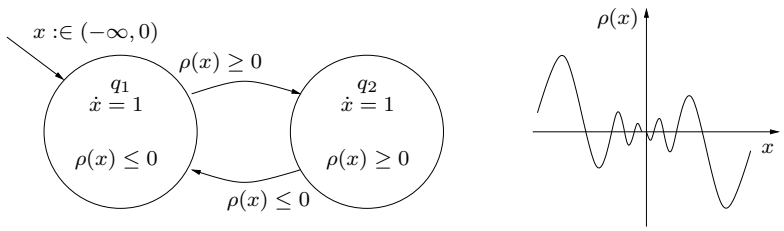

Figure 3.10. System with a smooth, non-analytic domain.

(q_1, x_0) will take an infinite number of discrete transitions before time $\tau_0 + |x_0|$ (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 if $0 < v_1, v_2 < w$. If in addition the inflow is less than the sum of the outflows ($w < v_1 + v_2$), then all infinite executions are Zeno. It is easy to show that the execution starting at time τ_0 takes an infinite number of transitions by time

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

■

3.3 Continuous dependence on initial state

In general, the executions of hybrid automata may change dramatically even for small changes in initial conditions. This fact is unavoidable in a modeling framework that is powerful enough to capture realistic systems. However, discontinuous dependence on initial conditions may cause problems, both theoretical and practical, when one tries to develop simulation algorithms for hybrid automata. Motivated by this we briefly investigate the dependence of the executions of a hybrid automaton on the initial state.

To talk about continuity we need to introduce a notion of distance between the hybrid states. The easiest way to do this is to use the standard Euclidean distance for the continuous states, together with the discrete metric for the discrete states. More precisely, for $q_1, q_2 \in Q$ consider the discrete metric $d_Q : Q \times Q \to \mathbb{R}$ defined by

$$d_Q(q_1, q_2) = \begin{cases} 1 & \text{if } q_1 = q_2 \\ 0 & \text{if } q_1 \neq q_2. \end{cases}$$

We define a metric on the hybrid state space, $d : (Q \times X) \times (Q \times X) \to \mathbb{R}$ as

$$d((q_1, x_1), (q_2, x_2)) = d_Q(q_1, q_2) + |x_1 - x_2|.$$

Exercise 3.8 *Show that* $d((q_1,x_1),(q_2,x_2)) \geq 0$, $d((q_1,x_1),(q_2,x_2)) = 0$ *if and only if* $(q_1,x_1) = (q_2,x_2)$, $d((q_1,x_1),(q_2,x_2)) = d((q_2,x_2),(q_1,x_1)) = 0$. *Show also that* d *satisfies the triangle inequality*

$$d((q_1,x_1),(q_2,x_2)) \leq d((q_1,x_1),(q_3,x_3)) + d((q_3,x_3),(q_2,x_2)).$$

Roughly speaking, a hybrid automaton is continuous if two executions starting close to one another remain close to one another.

Definition 3.15 (Continuous Hybrid Automaton) *A hybrid automaton is called continuous if for all* $(q_0,x_0) \in \mathrm{I}nit$, *for all finite executions* $\left(\{[\tau_i,\tau_i']\}_{i=0}^N, q, x\right)$ *starting at* (q_0,x_0) *and all* $\epsilon > 0$, *there exists* $\delta > 0$ *such that all maximal executions starting at some* $(\hat{q}_0,\hat{x}_0)$ *with* $d((\hat{q}_0,\hat{x}_0),(q_0,x_0)) < \delta$ *have a finite prefix* $\left(\{[\tau_i,\tau_i']\}_{i=0}^N, \hat{q}, \hat{x}\right)$ *such that* $|\hat{\tau}_N' - \tau_N'| < \epsilon$ *and*

$$d((\hat{q}_N(\hat{\tau}_N'),\hat{x}_N(\hat{\tau}_N')),(q_N(\tau_N'),x_N(\tau_N'))) < \epsilon.$$

Notice that as a consequence of the definition and the properties of the discrete metric, if a hybrid automaton is continuous then for any integer N one can choose δ small enough such that all executions starting within δ of one another go through the same sequence discrete states for their first N discrete transitions.

Exercise 3.9 *Show that this is the case.*

We give a characterization of continuous hybrid automata under an additional assumption, that helps to simplify the statements of the results.

Assumption 3.16 $\mathrm{I}nit = \mathrm{R}each = \bigcup_{q\in Q}\{q\} \times \mathrm{D}om(q)$.

This assumption is not critical for most of the discussion and relaxing it is tedious rather than conceptually difficult. As usual, for the set $\mathrm{D}om(q)$ we define

- Its interior, $\mathrm{D}om(q)^\circ$, as the largest open set contained in $\mathrm{D}om(q)$.
- Its closure, $\overline{\mathrm{D}om(q)}$, as the smallest closed set containing $\mathrm{D}om(q)$.
- Its boundary, $\partial\mathrm{D}om(q)$, as the set difference $\overline{\mathrm{D}om(q)} \setminus \mathrm{D}om(q)^\circ$.

The following theorem provides conditions under which a hybrid automaton is continuous.

Theorem 3.17 (Continuity with Initial Conditions) *A hybrid automaton satisfying Assumptions 3.2 and 3.16 is continuous if:*

1. *It is deterministic.*
2. *For all* $e \in E$, $R(e,\cdot)$ *is a continuous function.*
3. *For all* $q,q' \in Q$, *with* $(q,q') \in E$, $\mathrm{D}om(q) \cap G(q,q')$ *is an open subset of the boundary of* $\mathrm{D}om(q)$.

4. *There exists a function* $\sigma : Q \times X \to \mathbb{R}$ *differentiable in* x*, such that for all* $q \in Q$

$$\mathrm{D}om(q) = \{x \in X \,|\, \sigma(q, x) \geq 0\}.$$

5. *For all* $(q, x) \in Q \times X$ *with* $\sigma(q, x) = 0$*,* $L_f\sigma(q, x) \neq 0$*.*

For condition 2, notice that by condition 1 and Lemma 3.13, $R(e, \cdot)$ is a function and is not set valued. Roughly speaking, conditions 3–5 ensure that if from some initial state we can flow to a state from which a discrete transition is possible, then from all neighboring states we can do the same. Conditions 1 and 2 are then used to piece together such intervals of continuous evolution.

The proof of the theorem builds on a series of lemmas. Notice that Condition 4 implies that $\mathrm{D}om(q)$ is a closed set. Conditions 4 and 5 together imply that executions cannot meet the boundary of the domain tangentially along continuous evolution. In other words, continuous evolution is "transverse" to the boundary of the domain: it "pushes" states on the boundary of the domain either toward the interior of the domain or toward the complement of the domain, never along the boundary of the domain. This observation is summarized in the following lemma.

Lemma 3.18 *Consider a hybrid automaton that satisfies Assumptions 3.2 and conditions 4 and 5 of Theorem 3.17. Consider* $(\hat{q}, \hat{x}) \in \mathrm{I}nit$ *and assume there exists a finite execution,* $([\tau_0, \tau_0'], q, x)$ *starting at* $(\hat{q}, \hat{x})$ *with* $\tau_0' > \tau_0$*. Then* $x_0(t) \in \mathrm{D}om(q_0(t))^\circ$ *for all* $t \in (\tau_0, \tau_0')$*.*

Proof of Lemma 3.18: Since there is only one time interval in τ we drop the subscript 0 from x and q to simplify the notation. Since $\tau_0' > \tau_0$, $([\tau_0, \tau_0'], q, x)$ involves continuous evolution, therefore $x(t) \in \mathrm{D}om(\hat{q})$ for all $t \in [\tau_0, \tau_0')$. In particular, $\hat{x} \in \mathrm{D}om(\hat{q})$.

Assume, for the sake of contradiction, that there exists $t^* \in (\tau_0, \tau_0')$ such that $x(t^*) \in \partial\mathrm{D}om(\hat{q})$. Then, by condition 4, $\sigma(\hat{q}, x(t^*)) = 0$. Consider the Taylor series expansion of $\sigma(\hat{q}, x(t))$ about $t = t^*$

$$\sigma(\hat{q}, x(t)) = \sigma(\hat{q}, x(t^*)) + L_f\sigma(\hat{q}, x(t^*))(t - t^*) + o(t - t^*)^2$$

Since $\sigma(\hat{q}, x(t^*)) = 0$, by condition 5, $L_f\sigma(\hat{q}, x(t^*)) \neq 0$. Therefore, there exists a neighborhood, $\Omega \subseteq (\tau_0, \tau_0')$, of t^* and a $t \in \Omega$ such that[1] $\sigma(\hat{q}, x(t)) < 0$. Therefore, $x(t) \notin \mathrm{D}om(\hat{q})$ for some $t \in (\tau_0, \tau_0')$, which contradicts the assumption that $([\tau_0, \tau_0'], q, x)$ is an execution. ∎

Conditions 4 and 5 also eliminate certain pathological cases.

Lemma 3.19 *Consider a hybrid automaton that satisfies conditions 4 and 5 of Theorem 3.17. Then for all* $x \in \mathrm{D}om(q)$ *and all neighborhoods* W *of* x*,* $W \cap \mathrm{D}om(q) \setminus \{x\} \neq \emptyset$*.*

[1] Careful examination reveals that in fact $t > t^*$.

Proof of Lemma 3.19: Assume D$om(q)$ contains an isolated point, say, $\hat{x}$. Then there is a neighborhood W of $\hat{x}$ such that $W \cap \mathrm{D}om(q) = \{\hat{x}\}$. By condition 4 of Theorem 3.17, $\sigma(q, \hat{x}) = 0$ and $\sigma(q, x) < 0$ for all $x \in W \setminus \{\hat{x}\}$. Therefore, σ attains a local maximum at $(q, \hat{x})$ and, since σ is differentiable in x, $\frac{\partial \sigma}{\partial x}(q, \hat{x}) = 0$. This implies that $L_f\sigma(q, \hat{x}) = 0$, which contradicts condition 5 of Theorem 3.17. ∎

Lemma 3.20 *Consider a hybrid automaton satisfying Assumptions 3.2 and 3.16, and conditions 4 and 5 of Theorem 3.17. Consider* $(\hat{q}, \hat{x}) \in \mathrm{I}nit$ *and assume there exists a finite execution,* $([\tau_0, \tau_0'], q, x)$ *starting at* $(\hat{q}, \hat{x})$ *with* $\tau_0' > \tau_0$ *and* $x_0(\tau_0') \in \partial \mathrm{D}om(\hat{q})$. *Then there exists a neighborhood* W *of* $\hat{x}$ *and a continuous function* $T : W \cap \mathrm{D}om(\hat{q}) \to \mathbb{R}^+$ *such that for all* $\hat{x}' \in W \cap \mathrm{D}om(\hat{q})$ *there exists a finite execution* $([\tau_0, \tau_0 + T(\hat{x}')], q', x')$ *starting at* $(\hat{q}, \hat{x}')$ *such that*

1. $x_0'(\tau_0 + T(\hat{x}')) \in \partial \mathrm{D}om(\hat{q})$.
2. $x_0'(t) \in \mathrm{D}om(\hat{q})^\circ$ *for all* $t \in (\tau_0, \tau_0 + T(\hat{x}'))$.
3. *The function* $\Psi : W \cap \mathrm{D}om(\hat{q}) \to \partial \mathrm{D}om(\hat{q})$, *defined by* $\Psi(\hat{x}') = x_0'(\tau_0 + T(\hat{x}'))$, *is continuous.*

Proof of of Lemma 3.20: Since there is only one time interval in τ we drop the subscript 0 from x and q to simplify the notation. We also assume that $\tau_0 = 0$ without loss of generality. Let $\phi(t, \hat{x}) \in X$ denote the unique solution to the differential equation

$$\dot{x} = f(\hat{q}, x)$$

at time $t > 0$ starting at $\phi(0, \hat{x}) = \hat{x}$.

To show part 1, notice that, by the definition of an execution, $x(t) \in \mathrm{D}om(\hat{q})$ for all $t \in [\tau_i, \tau_i')$. Since $x(\tau_0') \in \partial \mathrm{D}om(\hat{q})$, $\sigma(\hat{q}, x(\tau_0')) = 0$. The function $\sigma(\hat{q}, \phi(t, y))$ is differentiable in t in a neighborhood of $(\tau_0', \hat{x})$. This is because, by condition 4, $\sigma(q, x)$ is differentiable in x and the flow $\phi(t, y)$ is differentiable in t. Moreover, $\sigma(\hat{q}, \phi(t, y))$ is continuous in y in a neighborhood of $(\tau_0', \hat{x})$. This is because σ and ϕ are both continuous. Finally,

$$\frac{\partial}{\partial t}\sigma(\hat{q}, \phi(t, y))\bigg|_{(t,y)=(\tau_0', \hat{x})} = L_f\sigma(\hat{q}, x(\tau_0')) \neq 0,$$

(by condition 5). By the implicit function theorem (in particular, the non-smooth version, see for example [37], Theorem 3.3.6), there exists a neighborhood Ω of τ_0' and a neighborhood $\hat{W}$ of $\hat{x}$, such that for each $\hat{x}' \in \hat{W}$ the equation $\sigma(\hat{q}, \phi(t, \hat{x}')) = 0$ has a unique solution $T(\hat{x}') \in \Omega$, where $T : \hat{W} \to \Omega$ is a continuous function such that $\phi(T(\hat{x}'), \hat{x}') \in \partial \mathrm{D}om(\hat{q})$.

To show part 2, we show that there exists a neighborhood $W \subseteq \hat{W}$ of $\hat{x}$ such that $\phi(t, \hat{x}') \in \mathrm{D}om(\hat{q})^\circ$ for all $\hat{x}' \in W \cap \mathrm{D}om(\hat{q})$ and all

$t \in (0, T(\hat{x}'))$. By Lemma 3.18, $\phi(t,x) \in \mathrm{D}om(\hat{q})^\circ$ for all $t \in (0, \tau_0')$. Therefore, by continuity of the solutions of differential equations with respect to initial conditions, there exists a neighborhood $W \subseteq \hat{W}$ of $\hat{x}$ such that $\phi(t,\hat{x}') \in \mathrm{D}om(\hat{q})^\circ$ for all $\hat{x}' \in W \cap \mathrm{D}om(\hat{q})$ and all $t \in (0,\tau_0') \setminus \Omega$. Moreover, $\sigma(\hat{q}, \phi(t,\hat{x}')) \neq 0$ for all $t \in \Omega$ with $t \neq T(\hat{x}')$, since $T(\hat{x}')$ is locally a unique solution to $\sigma(\hat{q}, \phi(t,\hat{x}')) = 0$. Therefore $\phi(t,\hat{x}') \in \mathrm{D}om(\hat{q})^\circ$ for all $t \in (0, T(\hat{x}'))$.

Finally, to show part 3, notice that, since $\phi(t,y)$ is continuous in both t and y, for all $\epsilon > 0$ there exists $\delta_1 > 0$, such that for all t with $|t - \tau_0'| < \delta_1$ and all $\hat{x}' \in W$ with $|\hat{x}' - \hat{x}| < \delta_1$, $|\phi(t,\hat{x}) - \phi(T(\hat{x}),\hat{x})| < \epsilon$ and $|\phi(T(\hat{x}'),\hat{x}) - \phi(T(\hat{x}'),\hat{x}')| < \epsilon$. Since T is continuous, there exists some $\delta_2 > 0$ such that for all $\hat{x}' \in W$ with $|\hat{x} - \hat{x}'| < \delta_2$, $|T(\hat{x}) - T(\hat{x}')| < \delta_1$. Setting $\delta = \min\{\delta_1, \delta_2\}$, it follows that for all $\hat{x}' \in W$ with $|\hat{x} - \hat{x}'| < \delta$,

$$\begin{aligned}|\Psi(\hat{x}) - \Psi(\hat{x}')| =& |\phi(T(\hat{x}),\hat{x}) - \phi(T(\hat{x}'),\hat{x}')| \\ \leq& |\phi(T(\hat{x}),\hat{x}) - \phi(T(\hat{x}'),\hat{x})| + |\phi(T(\hat{x}'),\hat{x}) - \phi(T(\hat{x}'),\hat{x}')| \\ <& 2\epsilon,\end{aligned}$$

which proves the continuity of Ψ. ∎

Proof of Theorem 3.17: Consider an arbitrary finite execution (τ, q, x) with $\tau = \{[\tau_i, \tau_i']\}_{i=0}^N$ and an $\epsilon > 0$. We show how to select $\delta > 0$ such that all executions starting at $(\hat{q},\hat{x})$ with $d((\hat{q},\hat{x}),(q_0(\tau_0),x_0(\tau_0))) < \delta$ have a finite prefix that ends up ϵ close to $(q_N(\tau_N), x_N(\tau_N))$. Without loss of generality we take $\delta < 1$ so that the executions are forced to start in the same discrete state $\hat{q} = q_0(\tau_0)$. The construction will be such that the executions are then forced to also follow the same sequence of discrete states.

We construct a sequence of sets $\{W_i, V_i\}_{i=0}^N$, where W_i is a neighborhood of $x_i(\tau_i)$ in $\mathrm{D}om(q_i(\tau_i))$ and V_i is a neighborhood of $x_i(\tau_i')$ in $\mathrm{D}om(q_i(\tau_i))$, such that the continuous evolution provides a continuous map from W_i to V_i and the reset provides continuous map from V_i to W_{i+1}. The notation is illustrated in Figure 3.11.

The construction is recursive, starting with $i = N$. Define $V_N = \{\bar{x} \in Dom(q_N(\tau_N)) \mid |\bar{x} - x_N(\tau_N')| < \epsilon\}$. V_N contains no isolated points by Lemma 3.19. We distinguish the following three cases:

Case 1: $\tau_N' > \tau_N$ and $x_N(\tau_N') \in \partial\mathrm{D}om(q_N(\tau_N))$. By Lemma 3.20, there exists a neighborhood, W, of $x_N(\tau_N)$ and a continuous function, $T_N : W \to \mathbb{R}$, such that for all $\hat{x} \in W$, $\phi(T_N(\hat{x}),\hat{x}) \in \partial\mathrm{D}om(q_N(\tau_N))$ and $\phi(t,\hat{x}) \in \mathrm{D}om^o(q_N(\tau_N))$ for all $t \in (0, T_N(\hat{x}))$. As in Lemma 3.20, define $\Psi_N : W \to \partial\mathrm{D}om(q_N(\tau_N))$ by $\Psi_N(\hat{x}) = \phi(T_N(\hat{x}),\hat{x})$. By the continuity of Ψ_N, there exists a neighborhood, $W_N \subset W \cap \mathrm{D}om(q_N(\tau_N))$, of $x_N(\tau_N)$ such that $\Psi_N(W_N) \subseteq V_N$.

Case 2: $\tau_N' > \tau_N$ and $x_N(\tau_N') \in \mathrm{D}om^o(q_N(\tau_N))$. Let W be a neighborhood of $x_N(\tau_N)$ such that for all $\hat{x} \in W$ and all $t \in (0, \tau_N' - \tau_N)$, $\phi(t,\hat{x}) \in \mathrm{D}om^o(q_N(\tau_N))$. Such a neighborhood exists, by Lemma 3.18 and the conti-

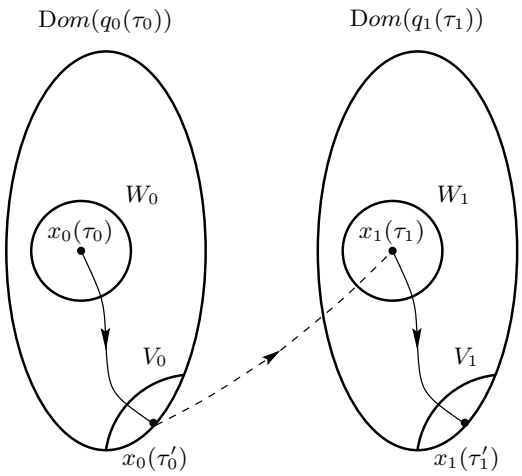

Figure 3.11. Illustration of the proof of Theorem 3.17 ($N = 1$, Case 1).

nuity of solutions to differential equations (cf. proof of Lemma 3.20). Define a constant function $T_N : W \to \mathbb{R}$ by $T_N(\hat{x}) = \tau'_N - \tau_N$, and a function $\Psi_N : W \to \mathrm{D}om(q_N(\tau_N))$ by $\Psi_N(\hat{x}) = \phi(T_N(\hat{x}), \hat{x})$. By continuous dependence of the solutions of the differential equation on the initial condition, there exists a neighborhood $W_N \subset W \cap \mathrm{D}om(q_N(\tau_N))$ of $x_N(\tau_N)$ such that $\Psi_N(W_N) \subseteq V_N$.

Case 3: $\tau'_N = \tau_N$. Define $W_N = V_N$, $T_N : W_N \to \mathbb{R}$ by $T_N(\hat{x}) \equiv 0$, and $\Psi_N : W_N \to \mathrm{D}om(q_N(\tau_N))$ to be the identity map. Clearly, $\Psi_N(W_N) = V_N$.

Next let us define V_{N-1}. Recall that $e_N = (q_{N-1}(\tau_{N-1}), q_N(\tau_{N-1})) \in E$, $x_{N-1}(\tau'_{N-1}) \in G(e_N)$ and $x_N(\tau_N) \in R(e_N, x_{N-1}(\tau'_{N-1}))$. Since the guard $G(e_N)$ is an open subset of $\partial Dom(q_{N-1}(\tau_{N-1}))$ and R is a continuous function, there exists a neighborhood V_N of $x_{N-1}(\tau'_{N-1})$ such that for all $\hat{x} \in V_{N-1} \cap \partial Dom(q_{N-1}(\tau_{N-1}))$, $\hat{x} \in G(e_N)$ and $R(e_N, \hat{x}) \subset W_N$.

Next, define T_{N-1} and Ψ_{N-1} using Lemma 3.20, as for Case 1 above. There exists a neighborhood W_{N-1} of $x_{N-1}(\tau_{N-1})$ such that $\Psi_{N-1}(W_{N-1}) \subset V_{N-1} \cap \partial \mathrm{D}om(q_{N-1}(\tau_{N-1}))$. Notice that if (τ, q, x) takes an instantaneous transition at τ_{N-1} (i.e. $\tau'_{N-1} = \tau_{N-1}$) some neighboring executions may take an instantaneous transition while others may have to flow for a while before they follow the transition of (τ, q, x).

By induction, we can construct a sequence of sets $\{W_i, V_i\}_{i=0}^N$ and continuous functions $T_i : W_i \to \mathbb{R}$ and $\Psi_i : W_i \to V_i$ for $i = 0, \ldots, N$. For $k = 0, \ldots, N$, define the function $\Phi_k : W_0 \to W_k$ recursively as

$$\Phi_0(\hat{x}) = \hat{x}$$
$$\Phi_k(\hat{x}) = R(\Psi_{k-1}(\Phi_{k-1}(\hat{x})))$$

For $k = 0, \ldots, N$, define the function $\gamma_k : W_0 \to \mathbb{R}$ as

$$\gamma_k(\hat{x}) = \sum_{\ell=0}^{k} T_\ell(\Phi_\ell(\hat{x})).$$

Then, $\Phi_k(\hat{x}) = \hat{x}_k(\hat{\tau}_k)$ and $\gamma_k(\hat{x}) = \hat{\tau}'_k - \hat{\tau}_0$ for the execution starting at $\hat{q} = q_0(\tau_0)$ and $\hat{x} \in W^0$. The functions Φ_k and γ_k are continuous by construction. By the continuity of γ_N, there exists $\delta_1 > 0$ such that for all $\hat{x}$ with $|\hat{x} - x_0(\tau_0)| < \delta_1$, we have $|\gamma_N(\hat{x}) - \gamma_N(x_0(\tau_0))| < \epsilon$. By the continuity of Ψ_N there exists $\delta_2 > 0$ such that for all $\hat{x}' \in W_N$ with $|\hat{x}' - x_N(\tau_N)| < \delta_2$, $|\Psi_N(\hat{x}/) - x_N(\tau'_N)| < \epsilon$. By the continuity of Φ_N, there exists $\delta_3 > 0$ such that for all $\hat{x} \in W_0$ with $|\hat{x} - x_0(\tau_0)| < \delta_3$, $|\Phi_N(\hat{x}) - x_N(\tau_N)| < \delta_2$. Since $\Psi_N(\Phi_N(\hat{x})) = \hat{x}_N(\hat{\tau}'_N)$, we have $|\hat{x}_N(\hat{\tau}'_N) - x_N(\tau'_N)| < \epsilon$. The proof is completed by setting $\delta = \min\{\delta_1, \delta_3\}$. ∎

3.4 Bibliography and Further Reading

The formal definition of hybrid automata is based on a fairly standard class of autonomous hybrid systems. The notation used here comes from [38, 39]. This class of systems has been studied extensively in the literature in a number of variations, for a number of purposes, and by a number of authors. Special cases of the class of systems considered here include switched systems [40], complementarity systems [41], mixed logic dynamic systems [42], and piecewise linear systems [43] (the autonomous versions of these, to be more precise). The hybrid automata considered here are a special case of the hybrid automata of [44] and the impulse differential inclusions of [45] (discussed in Appendix C of these notes), both of which allow differential inclusions to model the continuous dynamics. They are a special case of the General Hybrid Dynamical Systems of [46], which allow the continuous state to take values in manifolds (different ones for each discrete state). They are also a special case of hybrid input/output automata of [13], which, among other things, allow infinite-dimensional continuous state.

The simulation of hybrid systems presents special challenges, that need particular attention. General purpose simulation packages such as Matlab and Simulink can deal adequately with most complications. Specialized packages have also been developed that allow accurate simulation of hybrid systems (at least to the extent that this is possible in theory). For references see [47, 48, 49, 50, **?**]. See also [51] for a compositional language for hybrid system simulation.

The fundamental properties of existence and uniqueness of solutions, continuity with respect to initial conditions, etc. naturally attracted the attention of researchers in hybrid systems from fairly early on. The majority of the work in this area concentrated on developing conditions for

well-posedness (existence and uniqueness of solutions) for special classes of hybrid systems: piecewise linear systems [52, 53], complementarity systems [41, 54], mixed logic dynamical systems [42], etc. The discussion here is based on [38, 55, 39].

Continuity of the solutions with respect to initial conditions and parameters has also been studied, but somewhat less extensively. Motivated by questions of simulation, [56] established a class of hybrid systems that have the property of continuous dependence of solutions for almost every initial condition. More recently, an approach to the study of continuous dependence on initial conditions based on the Skorohod topology was proposed [57, **?**, **?**]. The Skorohod topology, used in stochastic processes for the space of cadlag functions [58], is mathematically appealing, but tends to be cumbersome to work with in practice. [38] was the basis for the discussion in this chapter; it presents a practical (but still limited) approach to the question of continuity.

Zeno executions are treated in [59, 60, 61] from a computer science perspective and [39, 62, 63, 64, 65] from a control theory perspective. [39, 66] attempt to define extensions of Zeno execution beyond the Zeno time, motivated by the classical definition of "sliding" flows and Filippov solutions for discontinuous vector fields [67, 68]. See also Problems 3.10–3.12.

3.5 Problems

Problem 3.1 *Consider a hybrid automaton satisfying Assumption 3.2. Show that for all finite-open executions* (τ, q, x) *there exists a finite execution* $(\hat{\tau}, \hat{q}, \hat{x})$ *such that* $(\tau, q, x) \sqsubset (\hat{\tau}, \hat{q}, \hat{x})$.

Problem 3.2 *Consider a set of variables* A. *A set,* $\mathcal{A}$, *of hybrid trajectories over* A *is called* prefix closed *if*

$$[(\tau, a) \in \mathcal{A}] \wedge [(\hat{\tau}, \hat{a}) \sqsubseteq (\tau, a)] \Rightarrow [(\hat{\tau}, \hat{a}) \in \mathcal{A}]$$

i.e. for all (τ, a) *in* $\mathcal{A}$ *all prefixes of* (τ, a) *are also in* $\mathcal{A}$ *Show that:*

1. *The class of prefix closed sets of hybrid trajectories is closed under arbitrary unions and intersections.*
2. *The complement of a prefix closed set of hybrid trajectories is not necessarily prefix closed.*
3. *Consider a hybrid automaton and an initial state* $(\hat{q}, \hat{x}) \in \mathrm{I}nit$. *Show that the set of executions starting at* $(\hat{q}, \hat{x})$ *is prefix closed. Deduce that the set of executions of the hybrid automaton is also prefix closed.*

Problem 3.3 *Consider the bouncing ball system (Figure 2.7).*

1. *Derive a hybrid automaton model* $BB = (Q, X, f, \mathrm{I}nit, D, E, G, R)$.

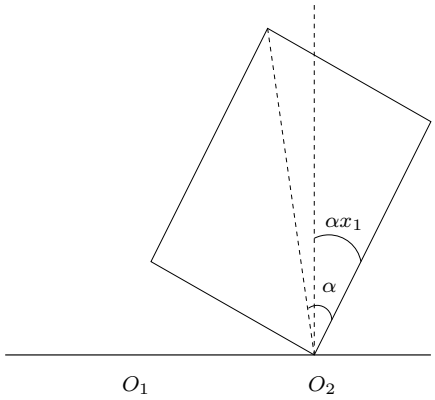

Figure 3.12. Rocking block system.

2. *Compute the set of states for which continuous evolution is impossible (special care is needed to determine what happens when $x_1 = 0$).*
3. *Is BB non-blocking? Is it deterministic?*
4. *By computing the time interval between two successive bounces, show that the bouncing ball automaton is Zeno.*

Problem 3.4 *Consider the thermostat system, shown in Figure 2.9.*

1. *Derive a hybrid automaton model,* $\mathrm{Th} = (Q, X, f, \mathrm{Init}, D, E, G, R)$.
2. *Compute the set of states from which continuous evolution is impossible.*
3. *Is* Th *non-blocking?*
4. *Is* Th *deterministic? If not, which of the conditions of Lemma 3.13 are violated?*
5. *Is the thermostat automaton Zeno? Why?*

Problem 3.5 *Show that the water tank automaton (Figure 3.2) and the chattering automaton (Figure 3.9) satisfy the conditions of Theorem 3.17, and therefore are continuous.*

Problem 3.6 *Ursula asked David to develop a hybrid model of a physical system. David produced a hybrid automaton model using the graphical representation. Ursula studied the model and decided that David's model was useless, because it violated Assumption 3.2: it contained edges ($e \in E$) with empty guards ($G(e) = \emptyset$) or empty resets ($x \in G(e)$, but $R(e, x) = \emptyset$). Show David how he can save face by constructing a hybrid automaton that accepts exactly the same set of executions as the first model he developed, but is such that for all $e \in E$, $G(e) \neq \emptyset$ and for all $x \in G(e)$, $R(e, x) \neq \emptyset$.*

Problem 3.7 *The rocking block dynamical system has been proposed as a model for the rocking motion of rigid bodies (Figure 3.12) during earthquakes [69]. Figure 3.13 shows the autonomous version of the rocking block system without the external excitation of the earthquake. Write the rocking block system in the hybrid automaton notation. Assume that* $r \in [0, 1]$. *Show that the system is non-blocking and deterministic. Compute the energy that the block loses during each impact with the ground. Simulate the system for* $\alpha = \pi/4$ *and* $r = 0.8$ *and show that it is Zeno.*

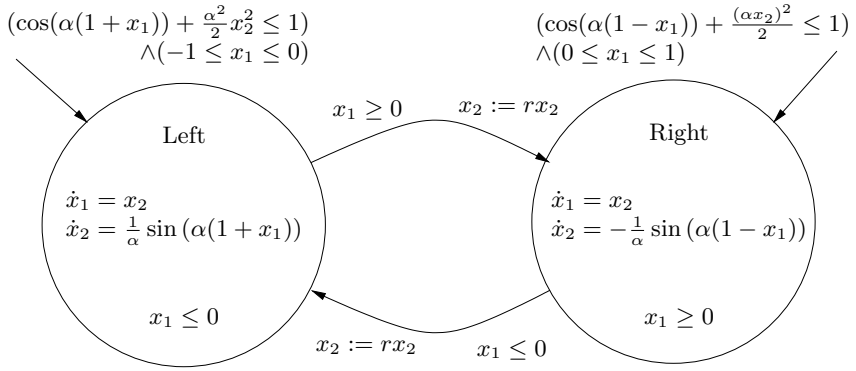

Figure 3.13. Directed graph representation of rocking block automaton.

Problem 3.8 *Consider the version of the water tank system shown in Figure 3.14.*

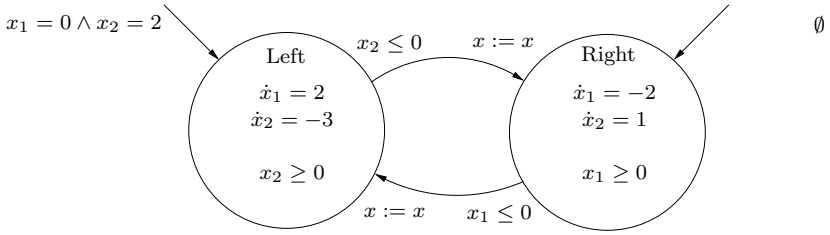

Figure 3.14. Water tank hybrid automaton.

1. *Implement this system in simulation.*
2. *The execution of the system for the given initial condition is Zeno. Compute the Zeno time, i.e. the finite limit of the infinite sequence of transition times.*
3. *Investigate the behavior of your simulation around the Zeno time. Do the precision parameters of the simulation and the simulation algo-*

rithm have any effect on the results?

Problem 3.9 *Consider now the thermostat system of Figure 2.9. Implement the system in simulation. Use two different methods for resolving the non-deterministic choice between discrete transition and continuous evolution.*

1. *"As soon as" semantics (a transition takes place as soon as it is enabled).*
2. *An exponential probability distribution for the transition times. For example, if along an execution (τ, q, x) at some point $t \in [\tau_i, \tau_i'] \in \tau$ we have $q_i(t) = \text{ON}$ and $x_i(t) = 21$ then the next transition time is given by a probability distribution of the form*

$$P[\tau_i' > T] = \begin{cases} e^{-\lambda(T-t)} & \text{if } x_i(T) \leq 22 \\ 0 & \text{otherwise} \end{cases}$$

for some constant $\lambda > 0$ (similarly for $q_i(t) = \text{OFF}$ and $x_i(t) = 19$). Investigate the effect of λ on the distribution of transition times.

Notice that in case 2 we are dealing with a stochastic hybrid system, since the transition times are probabilistic. In fact, this is an example from a general class of stochastic hybrid systems known as Piecewise Deterministic Markov Processes, discussed in Chapter 8.

Problem 3.10 *The hybrid automaton of Figure 3.15 is a variant of the bouncing ball automaton, where we assume that the bounces are not instantaneous but require time $\epsilon > 0$. An additional discrete state (shaded) has been added to allow us to model this.*

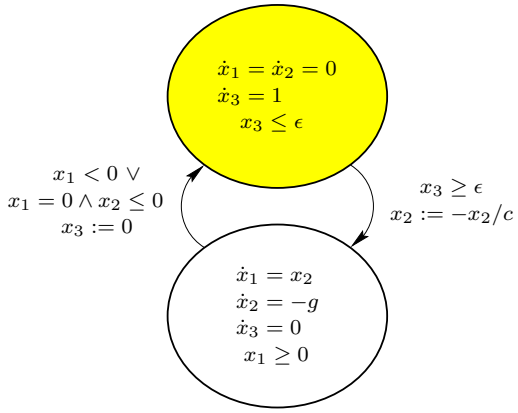

Figure 3.15. Temporal regularization of the bouncing ball automaton.

1. *Show that the automaton accepts a unique infinite execution for all initial conditions. Is this execution Zeno?*
2. *Simulate the automaton for different values of ϵ. What happens as $\epsilon \to 0$?*

Problem 3.11 *The hybrid automaton of Figure 3.16 is another variant of the bouncing ball automaton, where the ground is modeled as a stiff spring with spring constant $1/\epsilon > 0$ and no damping. Repeat Problem 3.10 for this automaton.*

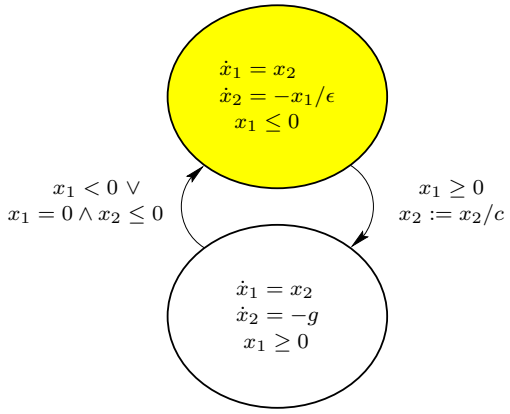

Figure 3.16. Dynamic regularization of the bouncing ball.

Problem 3.12 *Figures 3.17 and 3.18 show two variants of the water tank hybrid automaton. In Figure 3.17 , it is assumed that the switching of the inflow requires a small amount of time, $\epsilon > 0$; two additional discrete states (shaded) have been added to allow us to model this. In Figure 3.18, it is assumed that switching of the inflow requires the water in the tank to fall by at least a small amount (again $\epsilon > 0$). Repeat Problem 3.10 for these two systems. Is the limit as $\epsilon \to 0$ the same in the two cases?*

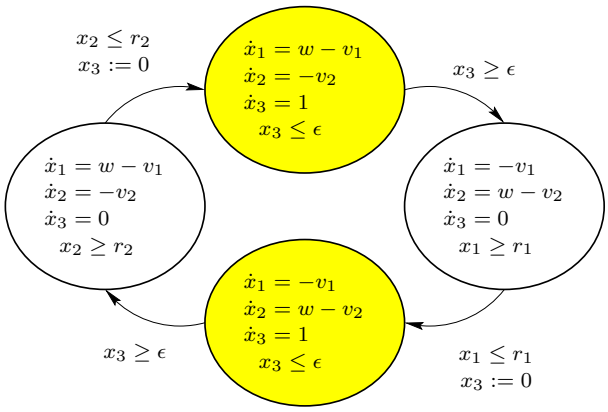

Figure 3.17. Temporal regularization of the water tank automaton.

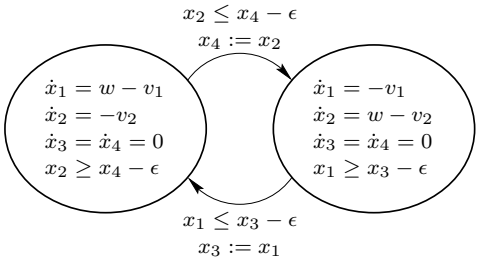

Figure 3.18. Spatial regularization of the water tank automaton.

Chapter 4
Analysis and Synthesis Problems

The reason why we are interested in modeling hybrid systems is that we would like to be able to analyze the resulting models, and infer some properties of the real system from them. If control inputs are available, we would also like to be able to design controllers such that the closed loop hybrid system satisfies certain desirable properties. In other words, given a hybrid automaton model of a physical system and some desirable property (*specification*) that we would like this system to possess, we would like to be able to address the following two questions:

1. **Verification**: does the hybrid automaton satisfy the desirable property (*meet the specification*).

2. **Synthesis**: if there are some design choices to be made (e.g. the system has control inputs and a controller needs to be designed) can the design be done in such a way that the resulting system meets the specification.

For real systems, verification and synthesis are usually followed by a process of **validation**: the theoretical hybrid design is implemented on a prototype of the real system and tests are performed to determine whether it meets the specification. It is not uncommon for a design to fail the validation test, due to factors that were omitted in the hybrid automaton model. The design then needs to be tuned further and the process of synthesis-verification-validation needs to be repeated.

4.1 Specifications

What kinds of specifications may one want to impose on a hybrid system? A common specification is stability: one would like to determine whether a hybrid system is stable or asymptotically stable. If control inputs are available, the synthesis problem becomes one of stabilization: Can one choose a controller such that the closed loop system is stable or asymptotically stable? Just as for purely continuous systems, Lyapunov methods are very useful in this context. Both the stability definitions and Lyapunov theorems need to be appropriately modified to account for the presence of discrete states. This topic will not be addressed further here.

We will concentrate primarily on properties that can be encoded as sets of hybrid trajectories. Recall that a hybrid trajectory over the state space $Q \times X$ (Definition 3.4) is a triple, (τ, q, x) consisting of a hybrid time set τ and two sequences of functions, q and x mapping each interval of τ to the discrete states Q and the continuous states $\mathbb{R}^n$ respectively. A specification can be encoded as a set of desirable hybrid trajectories, $\mathcal{H}$. A hybrid automaton is then said to meet the specification if all the executions of the hybrid automaton belong to this set $\mathcal{H}$. (Recall that the executions of a hybrid automaton are also hybrid trajectories, in particular the ones that meet the conditions of Definition 3.5).

What kind of properties can be encoded like this? The most common are properties that have to do with reachability. For example, the property

"The state (q, x) always remains in a set of states $F \subseteq Q \times X$"

is one such property, and so is the dual property

"The state (q, x) eventually reaches a set of states $F \subseteq Q \times X$"

The first property is known as a **safety property**. Here the set F is assumed to contain the "good" or "safe" states. Safety properties encode the requirement that "bad" things should not happen to our system. For example, when analyzing the behavior of two cars following one another on an automated highway, F can be the set of states for which the two cars have not collided (i.e. the spacing between them is greater than zero); we would like to ensure that the cars always remain in the set F (i.e. do not collide). Using notation from **Temporal Logic** this safety property can be written as

$$\Box[(q, x) \in F]$$

$\Box$ stands for "always"; the way to read the above formula is "always (q, x) is in F".

The second property ("The state (q, x) eventually reaches F") is a simple case of what is known as a **liveness property**. It reflects the fact that something good should eventually happen to our system. For example, cars

on an automated highway not only want to avoid collisions with other cars, but also want to reach their destination. In the temporal logic notation this property can be written as

$$\Diamond[(q, x) \in F]$$

$\Diamond$ stands for "eventually"; the way to read the above formula is "eventually (q, x) is in F".

Using concepts like these one can encode arbitrarily complex properties. For example the property

$$\Box\Diamond[(q, x) \in F]$$

stands for "always, eventually (q, x) is in F", or in other words, the state visits the set F "infinitely often". Another example is

$$\Diamond\Box[(q, x) \in F]$$

which stands for "eventually always (q, x) is in F", or in other words, (q, x) reaches F at some point and stays there for ever after. And so on.

How can one check that a hybrid automaton meets such a specification? Roughly speaking there are three different approaches to this problem:

1. **Model Checking**: For certain classes of hybrid systems the process can be carried out completely automatically. The system and the specification are encoded in an appropriate programming language, and given to a computer to analyze. After a few minutes/hours/days/weeks the computer either runs out of memory, or comes back with an answer: "The system satisfies the specification", or "The system does not satisfy the specification". In the latter case the computer also generates an execution of the hybrid automaton that fails to meet the specification; such an execution is known as a *witness*. The witness is useful for redesigning the system. The basics of this approach will be given in Section 4.3.

2. **Deductive**: Induction arguments, progress variables, etc. are used to develop a proof that the system meets the specification. Most of the work in structuring the proof has to be done my hand. Computational theorem proving tools may then be used to check the individual steps of the proof. The basics of this approach will be the topic of the rest of Section 4.2.

3. **Optimal Control and Viability Theory**: Reachability problems can also be encoded in an optimal control or viability theory setting. The key issues of the optimal control and viability approaches to reachability for continuous systems will be reviewed in Chapter 5. For hybrid systems, these approaches require some rather advanced machinery from optimal control theory and non-smooth analysis. The

foundations of the optimal control and viability approaches to reachability problems for hybrid systems will be presented in Chapter **??**. The viability theory approach will also be discussed in Appendix C, for a class of hybrid systems known as impulse differential inclusions. Most of the work with these approaches has to be done analytically (by hand!) Because optimal control problems are notoriously difficult to solve analytically one often has to resort to numerical tools (PDE solvers, etc.) to approximate the solution. The use of such tools for the analysis and synthesis of hybrid systems will be discussed in Chapter 7.

4.2 Deductive Methods

In this section we will introduce some basic principles of deductive analysis.The discussion is motivated by safety specifications, but can be extended to other types of specifications; examples are given in the problems at the end of the chapter.

First, we note the following immediate fact.

Proposition 4.1 *A hybrid automaton H satisfies a specification $\Box[(q,x) \in F]$ if and only if* R*each* $\subseteq F$.

Exercise 4.1 *Prove Proposition 4.1.*

Deductive arguments aim to establish bounds on R*each* through invariant sets. The definition of "invariant set" for hybrid automata is a direct generalization of the definition for continuous dynamical systems: a set of states is called invariant if all executions of the hybrid automaton starting in that set remain in that set for ever. More formally,

Definition 4.2 (Invariant Set) *A set of states, $M \subseteq Q \times X$, of a hybrid automaton, H, is called invariant if for all $(\hat{q},\hat{x}) \in M$, all executions (τ,q,x) starting at $(\hat{q},\hat{x})$, all $I_i \in \tau$ and all $t \in I_i$ we have that $(q_i(t),x_i(t)) \in M$.*

In the above statement "execution (τ,q,x) starting at $(\hat{q},\hat{x})$" refers to a hybrid trajectory with $(q_0(\tau_0),x_0(\tau_0)) = (\hat{q},\hat{x})$ that satisfies the discrete and continuous evolution conditions of Definition 3.5, but not necessarily the initial condition (i.e. we allow $(\hat{q},\hat{x}) \notin$ I*nit* in Definition 4.2). The reader is asked to forgive this slight abuse of the terminology.

The following fact is a direct consequence of the definition.

Proposition 4.3 *Consider a hybrid automaton H.*

1. *The union and intersection of two invariant sets of H are also invariant sets of H.*
2. *If M is an invariant set and* I*nit* $\subseteq M$*, then* R*each* $\subseteq M$.

Exercise 4.2 *Prove Proposition 4.3.*

The two parts of the proposition can be used together to provide progressively tighter bounds on R*each*, by finding invariant sets that contain I*nit* and then taking their intersection.

Given a specification of the form $\Box[(q,x) \in F]$, the idea is to find an invariant set that contains I*nit* (and hence R*each*) and is contained in F, i.e.

$$\mathrm{I}nit \subseteq M \subseteq F.$$

How does one prove that a set is invariant? Usually by induction. Assume we suspect that a certain set of states, $M \subseteq Q \times \mathbb{R}^n$, may be invariant. First of all we may want to check that the initial states are contained in M

- $\mathrm{I}nit \subseteq M$,

otherwise M may turn out to be useless for proving safety properties. Then we check that if continuous evolution starts from a state in M then it remains in M throughout. In other words, we check that

- for all $T \geq 0$,

if

- $(\hat{q}, \hat{x}) \in M$, and
- $x : [0,T] \to \mathbb{R}^n$ is the solution to $\frac{dx}{dt} = f(\hat{q}, x)$ starting at $x(0) = \hat{x}$, and
- $x(t) \in \mathrm{D}om(\hat{q})$ for all $t \in [0,T)$,

then

- $(\hat{q}, x(T)) \in M$.

Exercise 4.3 *Why is it sufficient to show that $(\hat{q}, x(T)) \in M$? What about the intermediate times $t \in (0,T)$? (The answer is not difficult, but requires a bit of thought.)*

Finally, we check that if a discrete transition is possible from somewhere in M, then the state after the discrete transition is also in M. In other words, if

- $(\hat{q}, \hat{x}) \in M$, and
- $(\hat{q}, \hat{q}') \in E$, and
- $\hat{x} \in G(\hat{q}, \hat{q}')$

then

- $R(\hat{q}, \hat{q}', \hat{x}) \subseteq M$

We have actually seen this procedure in practice already: these were the steps we followed to determine the set of states reachable by the water tank system.

4.3 Model checking

Finite state systems are relatively easy to work with because one can investigate their properties by systematically exploring their states. For example, one can decide if a finite state system will eventually visit a particular set of states by following the system trajectories one by one. This is tedious to do by hand, but is easy to implement on a computer. Moreover, the process is guaranteed to terminate: since the number of states is finite, sooner or later we will find ourselves visiting the same states over and over again. At this point either the desired set has already been reached, or, if not, it will never be reached.

With hybrid systems it is in general impossible to do this. Because the number of states is infinite, it is impossible to enumerate them and try to explore them one by one. However, there are hybrid systems for which one can find a finite state system which is, in some sense, equivalent to the hybrid system. This is usually done by partitioning the state space into a finite number of sets with the property that any two states in a give set exhibit similar behavior. Then, to decide whether the hybrid system has certain properties, one has to work with the sets of the partition (whose number is finite), as opposed to the infinite states of the original hybrid system. Moreover, the generation and analysis of the finite partition can be carried out automatically by a computer.

The process of automatically analyzing the properties of systems by exploring their state space is known as *model checking*. In this section we discuss some fundamental ideas behind model checking, and introduce a class of hybrid systems, known as *timed automata*, that are amenable to this approach. As with deductive methods the discussion will be motivated by safety (reachability) analysis. Because of the complexity of the material we will not develop the results in their full beauty and generality; directions for further reading are given in the concluding section of this chapter.

4.3.1 Transition Systems

We first introduce a very general class of dynamical systems, known as transition systems, on which the above questions can be formulated.

Definition 4.4 (Transition System) *A transition system, $T = (S, \delta, S_0, S_F)$ consists of*

- *A set of states S (finite or infinite);*

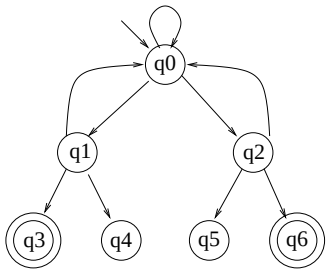

Figure 4.1. Finite state transition system.

- *A transition relation* $\delta : S \to 2^S$*;*
- *A set of initial states* $S_0 \subseteq S$*;*
- *A set of final states* $S_F \subseteq S$*.*

The set of final states is included because we are interested in reachability type specifications. We will use it to encode desired final states, sets of states in which we want to stay, etc.

Definition 4.5 (Trajectory of Transition System) *A trajectory of a transition system is finite or infinite sequence of states* $\{s_i\}_{i=0}^{N}$ *such that*

1. $s_0 \in S_0$*; and,*
2. $s_{i+1} \in \delta(s_i)$ *for all* $i = 0, 1, \ldots, N-1$*.*

Example (Finite State Transition System) A transition system with finite states is shown in Figure 4.1. The formal definition of the system is

1. $S = \{q_0, \ldots, q_6\}$;
2. $\delta(q_0) = \{q_0, q_1, q_2\}$, $\delta(q_1) = \{q_0, q_3, q_4\}$, $\delta(q_2) = \{q_0, q_5, q_6\}$, $\delta(q_3) = \delta(q_4) = \delta(q_5) = \delta(q_6) = \emptyset$;
3. $S_0 = \{q_0\}$ (denoted by an arrow pointing to q_0);
4. $S_F = \{q_3, q_6\}$ (denoted by a double circle).

■

Example (Transition System of a Hybrid Automaton) Hybrid automata can be transformed into transition systems by abstracting away time. Consider a hybrid automaton $H = (Q, X, \mathrm{I}nit, f, \mathrm{D}om, E, G, R)$ together with a distinguished "final" set of states $F \subseteq Q \times X$. We will define a transition system for the hybrid automaton. Start with

- $S = Q \times X$, i.e. $s = (q, x)$
- $S_0 = \mathrm{I}nit$

- $S_F = F$.

The transition relation δ can be defined in many parts: a discrete transition relation $\delta_e : S \to 2^S$ for each edge $e \in E$ of the graph and a continuous transition relation $\delta_C : S \to 2^S$ to model the passage of time. For each $e = (q, q') \in E$ define

$$\delta_e(\hat{q}, \hat{x}) = \begin{cases} \{q'\} \times R(e, \hat{x}) & \text{if } [\hat{q} = q] \wedge [\hat{x} \in G(e)] \\ \emptyset & \text{if } [\hat{q} \neq q] \vee [\hat{x} \notin G(e)] \end{cases}$$

For the continuous transition relation let

$$\delta_C(\hat{q}, \hat{x}) = \{(\hat{q}, \hat{x}') \in Q \times X \mid \exists T \geq 0,\ [x(T) = \hat{x}'] \wedge [\forall t \in [0, T), x(t) \in \mathrm{D}om(\hat{q})]\}$$

where $x(\cdot)$ denotes the solution of the differential equation

$$\dot{x} = f(\hat{q}, x) \text{ starting at } x(0) = \hat{x}$$

The overall transition relation is then

$$\delta(s) = \delta_C(s) \cup \bigcup_{e \in E} \delta_e(s)$$

In words, a transition from $s \in S$ to $s' \in S$ is possible in the transition system if either a discrete transition $e \in E$ of the hybrid system will bring s to s', or s can flow continuously to s' after some time. Notice that in the last statement time has been abstracted away. We do not care how long it takes to get from s to s', we only care whether it is possible to get there eventually. The transition system captures the sequence of "events" that the hybrid system may experience, but not the timing of these events. ■

Transition systems are designed to allow one to answer reachability (and other) questions algorithmically. For example, say we would like to answer the question

"Does there exist a trajectory of T such that $s_i \in S_F$ for some i?"

If this is the case we say that S_F *is reachable*. More formally,

Definition 4.6 (Reachability) *The set S_F is said to be reachable by the transition system T if there exists a finite trajectory $\{s_i\}_{i=0}^N$ with $N < \infty$ such that $s_N \in S_F$.*

Exercise 4.4 *As discussed above, one can associate a transition system, T, to a hybrid automaton and a distinguished set of its states F. Show that S_F is reachable by T if and only if $F \cap \mathrm{R}each \neq \emptyset$. (Hint: Consider the sequence of states $x_0(\tau_0)$, $x_0(\tau_0')$, ..., $x_i(\tau_i)$, $x_i(\tau_i')$, etc.)*

Questions of reachability for transition systems can be approached using the predecessor operator

$$\mathrm{P}re : 2^S \to 2^S.$$

Algorithm 1 (Backward Reachability)

initialization: $W_0 = S_F$, $i = 0$
repeat
  if $W_i \cap S_0 \neq \emptyset$
    return "S_F reachable"
  end if
  $W_{i+1} = \mathrm{P}re(W_i) \cup W_i$
  $i = i + 1$
until $W_i = W_{i-1}$
return "S_F not reachable"

Table 4.1. Backward reachability algorithm

For each set of states $S' \subseteq S$, $\mathrm{P}re(S')$ is defined as

$$\mathrm{P}re(S') = \{s \in S \mid \exists s' \in S' \text{ with } s' \in \delta(s)\}.$$

In other words, the operator $\mathrm{P}re$ takes a set of states, S', and returns the states that can reach S' in one transition. The algorithm given in Table 4.1 can then be used to determine if S_F is reachable by T.

Using an induction argument it is easy to show that if the algorithm *terminates* (i.e. at some point it returns "S_F reachable" or "S_F not reachable") then it produces the right answer.

Exercise 4.5 *Show this.*

This algorithm is written in what is known as *pseudo-code*. It is conceptually useful, but is still a long way from being implementable on a computer. To effectively implement the algorithm one needs to figure out a way to explain to the computer how to

1. store sets of states,
2. compute the $\mathrm{P}re$ of a set of states,
3. take the union and intersection of sets of states,
4. check whether a set of states is empty, and
5. check whether two sets of states are equal.

Exercise 4.6 *Show that 5. can be replaced by "take the complement of sets".*

If the number of states S is finite all of these are relatively easy to do by enumerating the states. None of these steps are completely straight forward, however, if the state has real valued components (as is the case with hybrid systems).

Even if one was able to perform all of these operations using a computer program, it is still unclear whether the program would always produce an

answer to the reachability question. The above algorithm may come up with the answer "S_F reachable", the answer "S_F not reachable", but it may also come up with no answer at all. This will be the case if new states get added to W_i each time we go around the repeat-until loop (hence $W_{i-1} \neq W_i$) but none of these states belongs to S_0 (hence $W_i \cap S_0 = \emptyset$).

Example (Non-Terminating System) Consider the transition system $T = (S, \delta, S_0, S_F)$ with $S = \mathbb{R}$,

$$\delta(x) = 2x$$

$S_0 = \{-1\}$, $S_F = \{1\}$. The Backwards Reachability algorithm produces the following sequence of sets:

$$W_0 = \{1\}, W_1 = \{1, \frac{1}{2}\}, \ldots, W_i = \{1, \frac{1}{2}, \ldots, \left(\frac{1}{2}\right)^i\}, \ldots$$

Notice that W_{i+1} contains one more element that W_i, therefore we will always have $W_i \neq W_{i+1}$. Moreover, because $W_i \subseteq \mathbb{R}_+$, -1 will never be in W_i, therefore $W_i \cap S_0 = \emptyset$. Hence the algorithm will not terminate. ■

With finite state systems termination is not a problem: the set W_i will sooner or later stop growing.

Example (Finite State Transition System (cont.)) When applied to the finite state system of Figure 4.1 the Backwards Reachability Algorithm produces the following sequence of sets:

$$W_0 = \{q_3, q_6\}, W_1 = \{q_1, q_2, q_3, q_6\}, W_2 = \{q_0, q_1, q_2, q_3, q_6\}$$

Notice that $W_2 \cap S_0 = \{q_0\} \neq \emptyset$. Therefore, after two steps the algorithm terminates with the answer "S_F reachable". ■

Exercise 4.7 *Assume the set S is finite and contains M states. Give an upper bound on the number of times the algorithm will have to perform the "repeat-until" loop.*

4.3.2 Bisimulation

Since finite state systems are so much easier to work with, we are going to try to turn our infinite state systems into finite state ones, by grouping together states that have "similar" behavior. Such a grouping of states is called a *partition* of the state space. A partition is a collection of sets of states, $\{S_i\}_{i \in I}$, with $S_i \subseteq S$, such that

1. Any two sets, S_i and S_j, in the partition are disjoint, i.e. $S_i \cap S_j = \emptyset$ for all $i, j \in I$ with $i \neq j$. (A family of sets with this property is called *mutually disjoint*).

2. The union of all sets in the partition is the entire state space, i.e.

$$\bigcup_{i \in I} S_i = S$$

(A family of sets with this property is said to *cover* the state space).

The index set, I, of the partition may be either finite or infinite. If I is a finite set (e.g. $I = \{1, 2, \ldots, M\}$ for $M < \infty$) then we say that the partition $\{S_i\}_{i \in I}$ is a *finite partition*.

Example (Finite State Transition System (cont.)) The collection of sets

$$\{q_0\}, \{q_1, q_2\}, \{q_3, q_6\}, \{q_4, q_5\}$$

is a partition of the state space S of the finite system of Figure 4.1. The collection

$$\{q_0\}, \{q_1, q_3, q_4\}, \{q_2, q_5, q_6\}$$

is also a partition. However, the collection

$$\{q_1, q_3, q_4\}, \{q_2, q_5, q_6\}$$

is not a partition (because it does not "cover" q_0), and neither is

$$\{q_0, q_1, q_3, q_4\}, \{q_0, q_2, q_5, q_6\}$$

(because q_0 appears twice). ■

Given a transition system, $T = (S, \delta, S_0, S_F)$ and a partition of the state space $\{S_i\}_{i \in I}$ we can define a transition system whose states are the elements of the partition $S_i \subseteq S$, rather than individual states $s \in S$. This transition system $\hat{T} = (\hat{S}, \hat{\delta}, \hat{S}_0, \hat{S}_F)$ is defined as

- $\hat{S} = \{S_i\}_{i \in I}$, i.e. the states are the sets of the partition;
- $\hat{\delta}$ allows a transition from one set in the partition (say S_i) to another (say S_j) if and only if δ allows a transition from some state in S_i (say $s \in S_i$) to some state in S_j (say $s' \in S_j$). In mathematical notation,

$$\hat{\delta}(S_i) = \{S_j \mid \exists s \in S_i, \exists s' \in S_j \text{ such that } s' \in \delta(s)\}$$

- A set in the partition (say S_i) is in the initial set of $\hat{T}$ (i.e. $S_i \in \hat{S}_0$) if and only if some element of S_i (say $s \in S_i$) is an initial state of the original transition system (i.e. $s \in S_0$).
- A set in the partition (say S_i) is a final set of $\hat{T}$ (i.e. $S_i \in \hat{S}_F$) if and only if some element of S_i (say $s \in S_i$) is a final state of the original transition system (i.e. $s \in S_F$).

Exercise 4.8 *Show that the above definitions are equivalent to*

$$\hat{\delta}(S_i) = \{S_j \mid \delta(S_i) \cap S_j \neq \emptyset\}$$
$$S_i \in \hat{S}_0 \Leftrightarrow S_i \cap S_0 \neq \emptyset$$
$$S_i \in \hat{S}_F \Leftrightarrow S_i \cap S_F \neq \emptyset.$$

You need to define $\delta(S_i)$ appropriately, but otherwise this is a tautology.

The transition system generated by the partition is known as the *quotient transition system*. Notice that if the partition is finite, then the quotient transition system $\hat{T}$ is a finite state system and therefore can be easily analyzed.

Using this method we can in principle construct finite state systems out of any transition system. The problem is that for most partitions the properties of the quotient transition system do not allow us to draw any useful conclusions about the properties of the original system. However, there is a special type of partition for which the quotient system $\hat{T}$ is in a sense equivalent to the original transition system, T. This type of partition is known as a *bisimulation*.

Definition 4.7 (Bisimulation) *A bisimulation of a transition system $T = (S, \delta, S_0, S_F)$ is a partition $\{S_i\}_{i \in I}$ of the state space S of T such that*

1. *S_0 is a union of elements of the partition,*
2. *S_F is a union of elements of the partition,*
3. *if one state (say s) in some set of the partition (say S_i) can transition to another set in the partition (say S_j), then all other states $\hat{s} \in S_i$ must be able to transition to some state in S_j. More formally, for all $i, j \in I$ and for all states $s, \hat{s} \in S_i$, if $\delta(s) \cap S_j \neq \emptyset$, then $\delta(\hat{s}) \cap S_j \neq \emptyset$.*

Exercise 4.9 *Show that a partition $\{S_i\}_{i \in I}$ is a bisimulation if and only if conditions 1 and 2 above hold and 3 is replaced by "for all i, $Pre(S_i)$ is a union of elements of $\{S_i\}_{i \in I}$".*

Example (Finite Transition System (cont.)) The partition

$$\{q_0\}, \{q_1, q_2\}, \{q_3, q_6\}, \{q_4, q_5\}$$

is a bisimulation of the finite system of Figure 4.1. Let us test this:

1. $S_0 = \{q_0\}$ which is an element of the partition;
2. $S_F = \{q_3, q_6\}$ which is also an element of the partition;
3. Let us study the third condition for the set $\{q_1, q_2\}$. From q_1 one can jump to the following sets in the partition

$$\{q_0\}, \{q_3, q_6\}, \{q_4, q_5\}$$

From q_2 one can jump to exactly the same sets in the partition. Therefore the third condition is satisfied for set $\{q_1, q_2\}$. It is also easy to check this condition for the remaining sets (for the set $\{q_0\}$ the third condition is trivially satisfied).

The partition

$$\{q_0\}, \{q_1, q_3, q_4\}, \{q_2, q_5, q_6\}$$

on the other hand, is not a bisimulation. For example, S_F is not a union of elements of the partition. Also, from q_1 one can transition to q_0, whereas from q_3 and q_4 (the other elements of the set $\{q_1, q_3, q_4\}$) one cannot transition to q_0. ■

Bisimulations are important because of the following property.

Theorem 4.8 (Reachability Equivalence) *Let $\{S_i\}_{i\in I}$ be a bisimulation of a transition system, T, and let $\hat{T}$ be the corresponding quotient transition system. S_F is reachable by T if and only if $\hat{S}_F$ is reachable by $\hat{T}$.*

Proof: Assume first that S_F is reachable by T. Then there exists a finite sequence $\{s_i\}_{i=0}^N$ such that $s_0 \in S_0$, $s_N \in S_F$ and $s_{i+1} \in \delta(s_i)$ for all $i = 0, 1, \ldots, N-1$. Clearly there exists a sequence $\{S'_i\}_{i=0}^N$ of elements of the partition such that $s_i \in S'_i$ for all $i = 0, 1, \ldots, N$. Because $s_N \in S_F$ and $s_N \in S'_N$ we have that $S'_N \in \hat{S}_F$. Likewise, $S'_0 \in \hat{S}_0$. Finally, because for all $i = 0, 1, \ldots, N-1$, $s_{i+1} \in \delta(s_i)$ we have that $S'_{i+1} \in \hat{\delta}(S'_i)$. Therefore, $\hat{S}_F$ is reachable by $\hat{T}$. Notice that so far the argument holds for any partition, not only for bisimulations.

Conversely, assume that $\{S_i\}_{i\in I}$ is a bisimulation and that $\hat{S}_F$ is reachable by the corresponding quotient transition system $\hat{T}$. Then there exists a finite sequence $\{S'_i\}_{i=0}^N$ of elements of the partition such that $S'_0 \in \hat{S}_0$, $S'_N \in \hat{S}_F$ and $S'_{i+1} \in \hat{\delta}(S'_i)$ for all $i = 0, 1, \ldots, N-1$. Pick an arbitrary state $s_0 \in S'_0$. Notice that because $s_0 \in S'_0 \in \hat{S}_0$ and the partition is a bisimulation we must have $s_0 \in S_0$. Moreover, since $S'_1 \in \hat{\delta}(S'_0)$ and the partition is a bisimulation there exist $s_1 \in S'_1$ such that $s_1 \in \delta(s_0)$. Proceed inductively, defining a finite sequence of states $\{s_i\}_{i=0}^N$ such that $s_i \in S'_i$ and $s_{i+1} \in \delta(s_i)$. Notice that since $s_N \in S'_N \in \hat{S}_F$ and the partition is a bisimulation we must have $s_N \in S_F$. Therefore S_F is reachable by T. ■

It should be noted that this is a simplified version of a much deeper theorem. It turns out that bisimulations preserve not only reachability properties, but all properties that can be written as formulas in a temporal logic known as the Computation Tree Logic (CTL).

Theorem 4.8 is a very important and useful result. For finite state systems its implications are mostly in terms of computational efficiency. If we can find a bisimulation of the finite state system (like we did in the finite state example discussed above) we can study reachability in the quotient system instead of the original system. The advantage is that the quotient system

Algorithm 2 (Bisimulation)

initialization: $\mathcal{P} = \{S_0, S_F, S \setminus (S_0 \cup S_F)\}$
while there exists $S_i, S_j \in \mathcal{P}$ such that $S_i \cap \mathrm{P}re(S_j) \neq \emptyset$
and $S_i \cap \mathrm{P}re(S_j) \neq S_i$ **do**
$S_i' = S_i \cap \mathrm{P}re(S_j)$
$S_i'' = S_i \setminus \mathrm{P}re(S_j)$
$\mathcal{P} = (\mathcal{P} \setminus S_i) \cup \{S_i', S_i''\}$
end while
return $\mathcal{P}$

Table 4.2. Bisimulation algorithm

will in general be much smaller that the original system. In the above example, the quotient system had 4 states whereas the original system had 7.

The implications are much more profound for infinite state systems. Even when the original transition system has an infinite number of states, sometimes we may be able to find a bisimulation that consists of a finite number of sets. Then we will be able to answer reachability questions for the infinite state system by studying an equivalent finite state system. Since finite state systems are so much easier to work with this could be a very big advantage. A class of hybrid systems for which we can always find finite bisimulations will be introduced in the next section.

The algorithm in Table 4.2 can be used to find a bisimulation of a transition system $T = (S, \delta, S_0, S_F)$.

The symbol $\setminus$ in the algorithm stands for set difference: $S_i \setminus \mathrm{P}re(S_j)$ is the set that contains all elements of S_i that are not elements of $\mathrm{P}re(S_j)$, in other words

$$S_i \setminus \mathrm{P}re(S_j) = \{s \in S \mid [s \in S_i] \wedge [s \notin \mathrm{P}re(S_j)]\}$$

The algorithm maintains a partition of the state space, denoted by $\mathcal{P}$, which gets refined progressively so that it looks more and more like a bisimulation. The definition of the bisimulation suggests that if $\mathcal{P}$ is to be a bisimulation then it must at least allow us to "distinguish" the initial and final states. We therefore start with a partition that contains three sets: S_0, S_F and everything else $S \setminus (S_0 \cup S_F)$, i.e.

$$\mathcal{P} = \{S_0, S_F, S \setminus (S_0 \cup S_F)\}$$

At each step of the algorithm, we examine the sets of the candidate partition. Assume we can find two sets $S_i, S_j \in \mathcal{P}$ such that $\mathrm{P}re(S_j)$ contains some elements of S_i (i.e. $S_i \cap \mathrm{P}re(S_j) \neq \emptyset$) but not all of them (i.e. $S_i \not\subseteq \mathrm{P}re(S_j)$, or, equivalently, $S_i \cap \mathrm{P}re(S_j) \neq S_i$). Then some states $s \in S_i$ may find themselves in S_j after one transition (namely those with $s \in S_i \cap \mathrm{P}re(S_j)$), while others may not (namely those with $s \in S_i \setminus \mathrm{P}re(S_j)$). This is not allowed if $\mathcal{P}$ is to be a bisimulation. We

therefore replace S_i by two sets: the states in S_i that can transition to S_j ($S_i' = S_i \cap \mathrm{P}re(S_j)$) and the states in S_i that cannot transition to S_j ($S_i'' = S_i \setminus \mathrm{P}re(S_j)$). Notice that after the replacement $\mathcal{P}$ has one more set than it did before. The process is repeated until for all sets $S_i, S_j \in \mathcal{P}$ either $S_i \cap \mathrm{P}re(S_j) \neq \emptyset$ or $S_i \cap \mathrm{P}re(S_j) \neq S_i$. The resulting collection of sets, $\mathcal{P}$ is a bisimulation. In fact:

Theorem 4.9 (Coarsest Bisimulation) *If the Bisimulation Algorithm terminates it will produce the coarsest bisimulation of the transition system (i.e. the bisimulation containing the smallest number of sets).*

Proof: We only show that if the algorithm terminates is produces a bisimulation. Assume that the bisimulation algorithm terminates and returns a partition $\mathcal{P}$; notice that by default $\mathcal{P}$ is a finite partition. Since S_0 and S_F are elements of the initial partition and the algorithm only splits elements, S_0 and S_F are unions of elements of $\mathcal{P}$. Moreover, the termination condition implies that for any $S_i, S_j \in \mathcal{P}$ either $S_i \cap \mathrm{P}re(S_j) = \emptyset$ or $S_i \subseteq \mathrm{P}re(S_j)$. Therefore $\mathrm{P}re(S_j)$ is a union of elements of $\mathcal{P}$. ∎

For finite state systems this algorithm is easy to implement (by enumerating the states) and will always terminate.

Example (Finite State Transition System (cont.)) Let us apply the bisimulation algorithm to the finite system of Figure 4.1. Initially

$$\mathcal{P} = \{S_0, S_F, S \setminus (S_0 \cup S_F)\} = \{\{q_0\}, \{q_3, q_6\}, \{q_1, q_2, q_4, q_5\}\}$$

Notice that $\mathrm{P}re(\{q_3, q_6\}) = \{q_1, q_2\}$. This is not an element of the partition $\mathcal{P}$. It intersects $\{q_1, q_2, q_4, q_5\}$ but is not equal to it. We therefore split $\{q_1, q_2, q_4, q_5\}$ into two sets

$$\{q_1, q_2, q_4, q_5\} \cap \mathrm{P}re(\{q_3, q_6\}) = \{q_1, q_2\}$$

and

$$\{q_1, q_2, q_4, q_5\} \setminus \mathrm{P}re(\{q_3, q_6\}) = \{q_4, q_5\}$$

After one iteration of the algorithm the partition becomes

$$\mathcal{P} = \{\{q_0\}, \{q_3, q_6\}, \{q_1, q_2\}, \{q_4, q_5\}\}$$

As we have already seen it is easy to check that this is a bisimulation. Clearly S_0 and S_F are elements of the partition. Moreover,

$$\mathrm{P}re(\{q_0\}) = \{q_0, q_1, q_2\}$$

which is a union of elements of the partition, and so on. ∎

The problem with using the bisimulation algorithm on finite state systems is that it may be more work to find a bisimulation than to investigate the reachability of the original system. Sometimes bisimulations can be

computed by "inspection", by taking advantage of symmetries of the transition structure. In the above example, we can immediately see that the left sub-tree is a mirror image of the right sub-tree. This should make us suspect that there is a bisimulation lurking somewhere. There is an entire community in computer science that develops methods for detecting and exploiting such symmetries.

When we try to apply the bisimulation algorithm to infinite state systems we face the same problems we did with the Backward Reachability algorithm: how to store sets of states in the computer, how to compute Pre, etc. Moreover, even in cases where we can do all these, the algorithm may never terminate. The reason is that not all infinite state transition systems have finite bisimulations. In the next section we will introduce a class of (infinite state) hybrid systems for which we can not only implement the above algorithm in a computer, but also ensure that it will terminate in a finite number of steps.

4.3.3 Timed Automata

Timed automata are a class of hybrid systems that involve particularly simple continuous dynamics: all differential equations are of the form $\dot{x} = 1$ and all the domains, guards, etc. involve comparison of the real valued states with constants ($x = 1$, $x < 2$, $x \geq 0$, etc.). Clearly timed automata are somewhat limited when it comes to modeling physical systems. They are very good however for encoding timing constraints ("event A must take place at least 2 seconds after event B and not more than 5 seconds before event C", etc.). For some applications, such as multimedia, Internet and audio protocol verification, etc. this type of description is sufficient for both the dynamics of the system and the properties that we want the system to satisfy. We conclude this chapter with a brief discussion of the properties of timed automata. Because complicated mathematical notation is necessary to formally introduce the topic we will give a rather informal exposition.

Consider $x \in \mathbb{R}^n$. A subset of $\mathbb{R}^n$ set is called *rectangular* if it can be written as a finite boolean combination of constraints of the form $x_i \leq a$, $x_i < b$, $x_i = c$, $x_i \geq d$, and $x_i > e$, where a, b, c, d, e are rational numbers. Roughly speaking, rectangular sets are "rectangles" in $\mathbb{R}^n$ whose sides are aligned with the axes, or unions of such rectangles. For example, in $\mathbb{R}^2$ the set

$$(x_1 \geq 0) \wedge (x_1 \leq 2) \wedge (x_2 \geq 1) \wedge (x_2 \leq 2)$$

is rectangular, and so is the set

$$[(x_1 \geq 0) \wedge (x_2 = 0)] \vee [(x_1 = 0) \wedge (x_2 \geq 0)]$$

The empty set is also a rectangular set (e.g. $\emptyset = (x_1 \geq 1) \wedge (x_1 \leq 0)$). However the set

$$\{x \in \mathbb{R}^2 \mid x_1 = 2x_2\}$$

is not rectangular.

Exercise 4.10 *Draw these sets in $\mathbb{R}^2$. You should immediately be able to see why rectangular sets are called rectangular.*

Notice that rectangular sets are easy to encode in a computer. Instead of enumerating the set itself (which is generally impossible since the set may contain an infinite number of states) we can store and manipulate the list of constraints used to generate the set (which is finite).

Roughly speaking, a timed automaton is a hybrid automaton which

- involves differential equations of the form $\dot{x}_i = 1$. Continuous variables governed by this differential equation are known as *clocks*.
- the sets involved in the definition of the initial states, guards and domain are rectangular sets
- the reset is either a rectangular set, or may leave certain states unchanged.

Example (Timed Automaton) An example of a timed automaton is given in Figure 4.2. We have

- $Q = \{q_1, q_2\}$;
- $X = \mathbb{R}^2$;
- $f(q_1, x) = f(q_1, x) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$;
- $\mathrm{I}nit = \{(q_1, (0,0))\}$;
- $\mathrm{D}om(q_1) = \{x \in \mathbb{R}^2 \mid x_2 \leq 3\}$, $\mathrm{D}om(q_2) = \{x \in \mathbb{R}^2 \mid x_1 \leq 5\}$;
- $E = \{(q_1, q_2), (q_2, q_1)\}$;
- $G(q_1, q_2) = \{x \in \mathbb{R}^2 \mid x_2 > 2\}$, $G(q_2, q_1) = \{x \in \mathbb{R}^2 \mid x_1 > 4\}$;
- $R(q_1, q_2, x) = \{(3, 0)\}$, $R(q_2, q_1, x) = \{(0, x_2)\}$

■

Exercise 4.11 *Is this timed automaton non-blocking? Is it deterministic?*

Notice that in the timed automaton of the example all the constants that appear in the definition are non-negative integers. It turns out that we can in general assume that this is the case: given any timed automaton whose definition involves rational and/or negative constants we can define

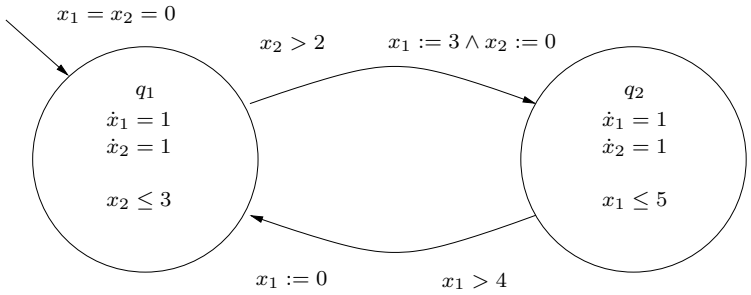

Figure 4.2. Example of a timed automaton.

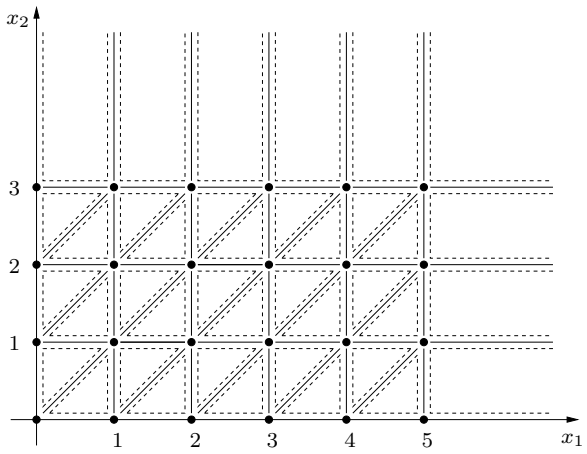

Figure 4.3. Region graph for the timed automaton of Figure 4.2.

an equivalent timed automaton whose definition involves only non-negative integers. This is done by "scaling" (multiplying by an appropriate integer) and "shifting" (adding an appropriate integer) some of the states.

We can turn timed automata into transition systems by "abstracting away" time, just like we did for general hybrid systems above. It turns out that the transition system corresponding to a timed automaton always has a finite bisimulation. One standard bisimulation that works for all timed automata is the *region graph*. The region graph for the timed automaton of Figure 4.2 is shown in Figure 4.5.

We will briefly describe the way the region graph is constructed. Assume that all the constants that appear in the definition of the timed automaton are non-negative integers (this can be done without loss of generality as noted above). As usual, let us label the continuous variables (clocks) as $x_1, \ldots, x_n$. Let C_i be the largest constant with which x_i is compared in any of the sets used in the definition of the timed automaton (initial sets,

guards, etc.). In the example $C_1 = 5$ and $C_2 = 3$. If all we know about the timed automaton is these bounds C_i, x_i could be compared with any integer M with $0 \leq M \leq C_i$ in some guard, reset or initial condition set. Therefore, the discrete transitions of the timed automaton may be able to "distinguish" states with $x_i < M$ from states with $x_i = M$ and from states with $x_i > M$, for all $0 \leq M \leq C_i$. Distinguish means, for example, that a discrete transition may be possible from a state with $x_i < M$ but not from a state with $x_i > M$ (because the guard contains the condition $x_i < M$). Because these sets may be distinguished by the discrete transitions we add them to our candidate bisimulation. In the example this gives rise to the sets

$$\begin{aligned}
\text{for } x_1 : & \; x_1 \in (0,1), x_1 \in (1,2), x_1 \in (2,3), x_1 \in (3,4), x_1 \in (4,5), x_1 \in (5,\infty) \\
& \; x_1 = 0, x_1 = 1, x_1 = 2, x_1 = 3, x_1 = 4, x_1 = 5, \\
\text{for } x_2 : & \; x_2 \in (0,1), x_2 \in (1,2), x_2 \in (2,3), x_2 \in (3,\infty) \\
& \; x_2 = 0, x_2 = 1, x_2 = 2, x_2 = 3.
\end{aligned}$$

The product of all these sets, i.e. the sets

$$\begin{aligned}
&\{x \in \mathbb{R}^2 \mid x_1 \in (0,1) \wedge x_2 \in (0,1)\} \\
&\{x \in \mathbb{R}^2 \mid x_1 \in (0,1) \wedge x_2 = 1\} \\
&\{x \in \mathbb{R}^2 \mid x_1 = 1 \wedge x_2 \in (0,1)\} \\
&\{x \in \mathbb{R}^2 \mid x_1 = 1 \wedge x_2 = 1\} \\
&\{x \in \mathbb{R}^2 \mid x_1 \in (1,2) \wedge x_2 \in (3\infty)\}, \textit{etc.}
\end{aligned}$$

define all the sets in $\mathbb{R}^2$ that the discrete dynamics (initial states, guards, domain and reset relations) can distinguish. Notice that in the continuous state space $\mathbb{R}^2$ these product sets are open squares (squares without their boundary), open horizontal and vertical line segments (line segments without their end points), points on the integer grid and open, unbounded rectangles (when some $x_i > C_i$).

Since $\dot{x}_1 = \dot{x}_2 = 1$, time flow makes the continuous state move diagonally up along 45° lines. By allowing time to flow the timed automaton may therefore distinguish points below the diagonal of each square, points above the diagonal and points on the diagonal. For example, points above the diagonal of the square

$$\{x \in \mathbb{R}^2 \mid x_1 \in (0,1) \wedge x_2 \in (0,1)\}$$

will leave the square through the line

$$\{x \in \mathbb{R}^2 \mid x_1 \in (0,1) \wedge x_2 = 1\}$$

points below the diagonal will leave the square through the line

$$\{x \in \mathbb{R}^2 \mid x_1 = 1 \wedge x_2 \in (0,1)\}$$

while points on the diagonal will leave the square through the point

$$\{x \in \mathbb{R}^2 \mid x_1 = 1 \wedge x_2 = 1\}$$

This leads us to split each open square in three: two open triangles and an open diagonal line segment.

It can in fact be shown that this is enough to generate a bisimulation.

Theorem 4.10 *The region graph is a finite bisimulation of the timed automaton.*

It should be stressed that the region graph is not necessarily the coarsest bisimulation. One may be able to find bisimulations with fewer sets; the elements of these bisimulations will be unions of sets of the region graph. It is generally considered bad form to try to construct the entire region graph to investigate properties of a timed automaton. Usually, one either uses the bisimulation algorithm to construct a coarse bisimulation, or uses the reachability algorithm directly to investigate reachability. The point of Theorem 4.10 is that it guarantees that there is at least one finite bisimulation for each timed automaton, which in turn guarantees that the Bisimulation and Reachability algorithms can be implemented and will terminate.

A counting exercise reveals that the total number of regions in the region graph is of the order of

$$n!\, 2^n \prod_{i=1}^{n} (2C_i + 2)$$

(! denotes factorial.) Even for relatively small n this is a very large number! (! denotes exclamation point) What is worse, the number grows very quickly (exponentially) as n increases. (In addition to $n!$ and 2^n, there is another hidden exponential in this formula. Because on a computer numbers are stored in binary, C_i is exponential in the number of bits used to store it).

This is bad news. It implies that a relatively simple timed automaton can give rise to a region graph with a very large number of sets, which will be a nightmare to analyze. It turns out that this is a general property of timed automata and has nothing to do with the way the region graph was constructed. Because timed automata have finite bisimulations, we know that they can be automatically analyzed by a computer. However, in the worst case, the number of operations that the computer will have to perform to analyze the system will be exponential in the size of the problem (e.g. the length of the input file we need to code our timed automaton for the computer). This is irrespective of how well we write the program. In computational complexity terminology, model checking for timed automata turns out to be *PSPACE Complete*.

A second bit of bad news about the model checking approach is that it does not work for more complicated systems. For example, one can show that simple variants of timed automata do not have finite bisimulations.

This is the case for example if we allow $\dot{x}_i = c_i(q)$ for some constant $c_i \neq 1$ (*skewed clocks*, leading to *multi-rate automata*), or allow comparisons between clocks (terms of the form $x_i \leq x_j$ in the guards or reset relations), or reseting one clock to another (terms of the form $x_i := x_j$ in the reset relations). In computer theory terminology, the reachability question for such systems is *undecidable*, i.e. there does not exist an algorithm that will answer the question in finite time.

Fortunately, there are still many interesting systems to which the model checking approach can produce very valuable results. In addition to timed automata, reachability problems are known to be decidable for the so called *initialized* variant of certain classes of hybrid systems. These are systems for which the continuous state is reset (possibly non-deterministically in a rectangular set) every time the continuous dynamics change as a result of a discrete transition, and the continuous dynamics are governed by

- Skewed clocks, $\dot{x}_i = c_i(q)$ with $c_i(q)$ a rational number;
- Constant differential inclusions $\dot{x}_i \in [l_i(q), u_i(q)]$, with $l_i(q) \leq u_i(q)$ rational or possibly $\pm\infty$;
- Linear differential equations, $\dot{x} = A(q)x$, where $A(q)$ is a matrix with rational entries that is either nilpotent, or has distinct, real eigenvalues, or distinct imaginary eigenvalues.

Moreover, the model checking approach can even be used on classes of systems for which reachability problems are known to be undecidable in general, but the operations necessary to implement the reachability or bisimulation algorithms can be encoded computationally. This is the case, for example, for non-initialized systems of the types listed above, linear systems with complex eigenvalues, systems where clocks are compared with one another, etc. In this case there is no guarantee that the reachability/bisimulation algorithm will ever terminate, but if it does then it is guaranteed to give a right answer. Finally, it can be shown that for more general classes of hybrid systems (e.g. with nonlinear continuous dynamics, where it is even impossible to encode the operations needed for the reachability/bisimulation algorithms) the reachability problem over compact time intervals can be approximated arbitrarily closely by systems for which reachability questions are decidable or semi-decidable (e.g. systems with constant differential inclusions). The price to pay for this approximation is that the number of discrete states needed for the approximation is in general exponential in the accuracy of the approximation.

4.4 Bibliography and Further Reading

Of the analysis questions listed in this chapter, the one that has arguably attracted the most attention if the question of stability of equilibria and

invariant sets. Most of the work in this area has concentrated on extensions of Lyapunov's Direct Method to the hybrid domain [70, 71, 72]. Approaches using Lyapunov like functions (barrier functions) have also been developed for the study of other properties, safety properties **??** and classes of liveness properties **??**. The work of [73] provided effective computational tools, based on Linear Matrix Inequalities, for applying these results to a class of piecewise linear systems. [38, 74] discuss extensions of LaSalle's Method and Lyapunov's Indirect Method to the hybrid domain. Related to the latter is also the work of [75, 76], where a direct study of the stability of piecewise linear systems is developed. For an excellent overview of the literature in this area the reader is referred to [77].

The corresponding synthesis problem of stabilization has been somewhat less extensively studied. Much of the work in this area deals with switched systems (usually linear and/or planar). The proposed stabilization schemes typically involve selecting appropriate times for switching between a set of given systems [40, 78, 79, 80, 76]. In some cases this approach has been extended to robust stabilization schemes for systems that involve certain types of uncertainty [81, 75]. A good coverage of the stability and stabilization problems for hybrid systems can be found in [82].

Temporal logic is widely used in computer theory to encode properties given as sets of trajectories (safety properties, etc.) as well as dynamics for discrete systems. A very thorough treatment can be found in [83, 84].

Deductive methods are commonly used with discrete systems; see [83, 84, 85] for an overview. One way of formally extending deductive arguments to the hybrid domain is presented in [86, 13, 87]; the approach of [86, 13] has been applied to a number of examples, primarily to establish the safety of transportation systems [88, 89, 90, 91, 92, 93].

Deductive arguments can be automated (at least partly) using theorem provers. One tool that provides computational support for deductive arguments for hybrid systems is STeP [94]. Other theorem provers that have been used to code deductive arguments for hybrid systems are PVS **??** and HOL **??**.

Timed automata were the first class of hybrid systems that were shown to be amenable to model checking methods in the classic paper [11]. Since then a number of other classes of hybrid systems with this property have been established: initialized multi-rate automata [95], initialized hybrid automata with continuous dynamics governed by constant differential inclusions [96] and classes of initialized hybrid automata with continuous dynamics governed by linear differential equations [97]. The limits of undecidability have been established in [96]. It has also been shown that a very wide class of hybrid systems can be approximated arbitrarily closely by such "decidable" hybrid systems [98] (albeit at the cost of exponential computational complexity). For an excellent overview of the developments in this area see [44].

In the case of hybrid control systems, related methods have been developed for automatically synthesizing controllers to satisfy specifications (e.g. given in temporal logic) whenever possible [14, 99, 15, 100].

Based on the theoretical results, computational tools been developed to automatically perform verification or synthesis tasks [16, 101, 102, 103, 104].

4.5 Problems

Problem 4.1 *Show that a hybrid system satisfies the specification* $\Box[(q,x) \in F]$ *if and only if there exists an invariant set* M *such that:*

$$\mathrm{I}nit \subseteq M \subseteq F.$$

Problem 4.2 *Consider the thermostat system, shown in Figure 2.9.*

1. *Assume that* $\mathrm{I}nit = \{\mathrm{OFF}, \mathrm{ON}\} \times \{20\}$. *Show that the system satisfies the specification* $\Box\,[(q,x) \in \{\mathrm{OFF}, \mathrm{ON}\} \times [18, 22]]$. *Does the system satisfy the specification* $\Box\,[(q,x) \in \{\mathrm{OFF}, \mathrm{ON}\} \times [19, 21]]$*?*
2. *Assume now that* $\mathrm{I}nit = \{\mathrm{OFF}, \mathrm{ON}\} \times \mathbb{R}$. *Show that the system satisfies the specifications*

$$\Diamond\Box\,[(q,x) \in \{\mathrm{OFF}, \mathrm{ON}\} \times [18, 22]]$$

and

$$\Box\Diamond[(q,x) \in \{\mathrm{OFF}, \mathrm{ON}\} \times [19, 21].$$

Problem 4.3 *As discussed in Section 4.3, one can associate a transition system,* T*, to a hybrid automaton and a distinguished set of its states* F*. Show that* S_F *is reachable by* T *if and only if* $F \cap \mathrm{R}each \neq \emptyset$.

Problem 4.4 *For a transition system,* T*, define an appropriate operator* $\mathrm{P}ost : 2^S \to 2^S$ *that for each set of states* $\hat{S} \subseteq S$ *returns the set of states that can be reached from* $\hat{S}$ *in one transition. Hence develop a forward reachability algorithm. Show that if your algorithm terminates then it produces the right answer.*

Problem 4.5 *Consider the water tank system, shown in graphical notation in Figure 3.2. Assume that* $\max\{v_1, v_2\} < w < v_1 + v_2$. *Show that the set of states*

$$\{(q,x) \mid (x_1 \geq r_1) \wedge (x_2 \geq r_2)\}$$

is invariant.

The condition $\max\{v_1, v_2\} < w < v_1 + v_2$ *implies that the rate at which water is added to the system is less than the rate at which water is removed. Physical intuition suggests that in this case at least one of the water tanks*

will have to eventually drain. Why does the analysis of the hybrid automaton fail to predict that?

Problem 4.6 *Consider a hybrid automaton.*

1. *Show that the class of invariant subsets of $Q \times X$ is closed under arbitrary unions and intersections.*
2. *Show that if $M \subseteq Q \times X$ is invariant and $\mathrm{Init} \subseteq M$, then $\mathrm{Reach} \subseteq M$.*

Problem 4.7 *Consider the timed automaton shown in Figure 4.4, and its region graph shown in Figure 4.5. Let $e_1 = (q_1, q_2)$ and $e_2 = (q_2, q_1)$. For the four regions, $P_1, \ldots, P_4$ shown in Figure 4.5 compute the predecessor operators Pre_{e_1} (states that can end up in P after transition e_1), Pre_{e_2} (states that can end up in P after transition e_2) and*

$$\mathrm{Pre}_C(P) = \{(q,x) \in Q \times \mathbb{R}^2 \mid \exists (q',x') \in P, t \geq 0, \text{ such that } (q = q') \wedge \left(x' = x + t \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right)\}$$

Verify that all the predecessors are unions of partitions in the region graph (or the empty set). (Note: The domains are assumed to be the whole of X and so are omitted from the figure.)

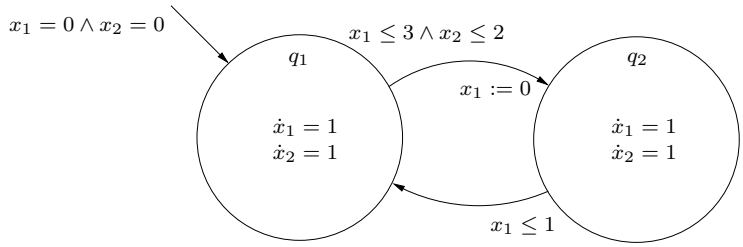

Figure 4.4. A timed automaton.

Problem 4.8 *A hybrid automaton is called domain preserving if the state remains in the domain along all executions, i.e. if for all executions (τ, q, x) for all $I_i \in \tau$ and for all $t \in I_i$, $x_i(t) \in \mathrm{Dom}(q_i(t))$. Assume that for all $\hat{q} \in Q$, $\mathrm{Dom}(\hat{q})$ is a closed set.*

1. *Show that the hybrid automaton is domain preserving if*
   - *for all $(\hat{q}, \hat{x}) \in \mathrm{Init}$, we have that $\hat{x} \in \mathrm{Dom}(\hat{q})$; and,*
   - *for all $\hat{q} \in Q$, for all $\hat{x} \in \mathrm{Dom}(\hat{q})$ and for all $(\hat{q}, \hat{q}') \in E$ such that $\hat{x} \in G(\hat{q}, \hat{q}')$, $R(\hat{q}, \hat{q}', \hat{x}) \subseteq \mathrm{Dom}(\hat{q}')$.*

   what if the system is blocking?
2. *Show that the bouncing ball system of Figure 4.6 is domain preserving.*

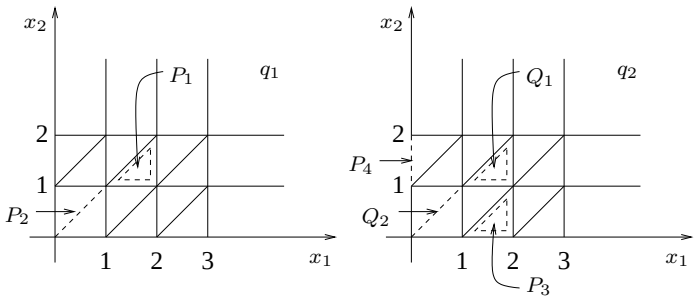

Figure 4.5. Region graph for the automaton of Figure 4.2

Problem 4.9 *Consider the bouncing ball system of Figure 4.6. Notice that in this case we assume that the ball is released at height h with zero vertical velocity.*

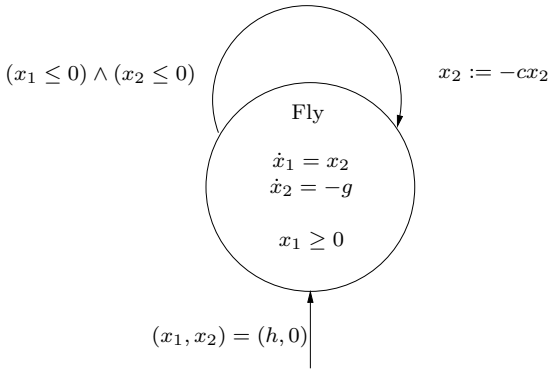

Figure 4.6. Bouncing ball

1. *Show that if $c \in [0, 1]$ then the system satisfies the specification*

$$\Box \left[(q, x) \in \left\{ (\text{Fly}, x) \mid x_1 + \frac{x_2^2}{2g} \leq h \right\} \right].$$

2. *Show that if $c \in [0, 1)$ then for any $i \in \mathbb{N}$ the system satisfies the specification*

$$\Diamond \left[(q, x) \in \left\{ (\text{Fly}, x) \mid x_1 + \frac{x_2^2}{2g} \leq \frac{1}{2^i} \right\} \right].$$

Problem 4.10 *Consider the rocking block system of Figure 3.13. Assume that $r \in [0, 1)$. Show that the system satisfies the specifications:*

$$\Box \left[\cos(\alpha(1 + x_1)) + \frac{\alpha^2}{2} x_2^2 \leq 1 \right]$$

and

$$\Box\Diamond[x_1 = 0].$$

Chapter 5

Controller Synthesis for Discrete and Continuous Systems

This chapter is the first of two chapters on designing control laws for hybrid systems. What is a control law? It is a method by which we steer the automaton using input variables, so that the closed loop exhibits certain desirable properties. A control problem involves:

1. A plant
2. A specification
3. A controller

Controller synthesis involves coming up with a methodology for designing controllers that meet the specification. The discussion in this chapter is based on [**?**] and [36, 105]. Here, we will deal mostly with *safety* specifications; although the methodology presented in this chapter can be extended to more general classes of properties.

Our first step will be to augment the finite state automaton and continuous state differential equation models of Chapter 3 with input variables. Typically some input variables can be controlled (their values can be assigned at will by a controller), while others cannot be controlled (their values are determined by the environment). Uncontrollable variables (or *disturbances*) typically represent:

- noise in the sensors, error from numerical computations;
- external forces, such as wind;
- unmodeled dynamics;

- uncertainty about the actions of other agents (such as in the air traffic control and highway system examples).

Our focus will be on the design of controllers to ensure that the overall system satisfies the safety property, while imposing minimal restrictions on the controls it allows. There are at least two reasons why such a controller is desirable:

1. As discussed above, safety properties can sometimes be satisfied using trivial controllers (that cause deadlocks or zeno executions for example). Imposing as few restrictions as possible allows us to find a meaningful controller whenever possible.
2. In many cases multiple, prioritized specifications are given for a particular problem. Imposing fewer restrictions on the controls when designing controllers for higher priority specifications allows us greater flexibility when trying to satisfy lower priority specifications.

5.1 Controller synthesis for discrete systems

We will start with the simplest case, the design of controllers for finite state automata.

5.1.1 Controlled Finite State Automaton

Consider the discrete state system of Chapter **??**, now augmented with discrete input and disturbance variables.

Definition: A *controlled finite state automaton* M is a mathematical model of a system represented as

$$(Q, \Sigma, \mathrm{I}nit, R) \tag{5.1}$$

where

- Q is a finite set of *discrete state variables*
- $\Sigma = \Sigma_1 \cup \Sigma_2$ is a finite set of *discrete input variables*, where Σ_1 contains the controller's inputs and Σ_2 contains the environment's inputs, which cannot be controlled
- $\mathrm{I}nit \subseteq Q$ is a set of *initial states*
- $R : Q \times \Sigma \to 2^Q$ is a *transition relation* which maps the state and input space to subsets of the state space and thus describes the transition logic of the finite automaton

Each transition between states depends on a joint action (σ_1, σ_2), where $\sigma_1 \in \Sigma_1$ and $\sigma_2 \in \Sigma_2$. The behavior of the finite state automaton is *nondeterministic*: the transition relation $R(q, \sigma_1, \sigma_2)$ represents a set of possible

new states, rather than a single unique state. Transitions are prevented, or blocked, from occurring at state q by setting $R(q, \sigma_1, \sigma_2) = \emptyset$.

A *system trajectory* $(q[\cdot], \sigma_1[\cdot], \sigma_2[\cdot]) \in Q^\omega \times \Sigma_1^\omega \times \Sigma_2^\omega$ is a finite or infinite sequence of states and actions which satisfies, for $i \in \mathbb{Z}$,

$$q[0] \in \mathrm{I}nit \text{ and } q[i+1] \in R(q[i], \sigma_1[i], \sigma_2[i]) \tag{5.2}$$

The controller specification is a trajectory acceptance condition Ω. The trajectory acceptance condition describes a desired goal that the system should achieve, which is expressed as a specification on the state trajectory. For safety specifications, in which the state trajectories must remain within a safe subset $F \subseteq Q$, the trajectory acceptance condition is written as $\Omega = \Box F$, meaning that $\forall i, q[i] \in F$.

The controller wins the game (UNDEFINED??) if the trajectory satisfies $\Box F$, otherwise the environment wins.

5.1.2 Controller Synthesis

The problem of synthesizing control laws $\sigma_1[\cdot] \in \Sigma_1^\omega$ in the presence of uncertain actions $\sigma_2[\cdot] \in \Sigma_2^\omega$ was first posed by Church in 1962 [106], who was studying solutions to digital circuits, and was solved by Büchi and Landweber [107] and Rabin [108] in the late 1960's and early 1970's using a version of the von Neumann-Morgenstern discrete game [**?**]. More recently, Ramadge and Wonham [109] added new insight into the structure of the control law. A temporal logic for modeling such games is introduced in [**?**].

We assume that the goal of the environment could be directly orthogonal to that of the controller's. This is a key assumption in our derivation of controllers for safety critical systems: the control law must protect against worst case uncertainty in the actions of the environment. With most realistic systems, the designer has a model of the environment and its actions: the better the model, the more flexibility the designer has in choosing a control law.

Returning to the controller specification of establishing the largest class of initial conditions for which there exists a controller that keeps all the executions of the automation inside $F \subset Q$. We will begin with the demonstration of how we might proceed through an example. Consider a finite automaton with $\mathbf{Q} = \{q_1, \ldots, q_{10}\}$, $\mathbf{U} = \mathbf{D} = \{1, 2\}$, $F = \{q_1, \ldots, q_8\}$, and the transition structure of Figure 5.1 (the inputs are listed in the order (u, d) and * denotes a "wild-card"). Try to establish the largest set of initial conditions, W^* such that there exists a feedback controller that keeps the execution inside F. Clearly:

$$W^* \subseteq F = \{q_1, \ldots, q_8\}$$

Next, look at states that can end up in F in one transition (q_4, q_5, and q_8). Notice that from q_4 if $u = 1$ whatever d chooses to do we remain in F,

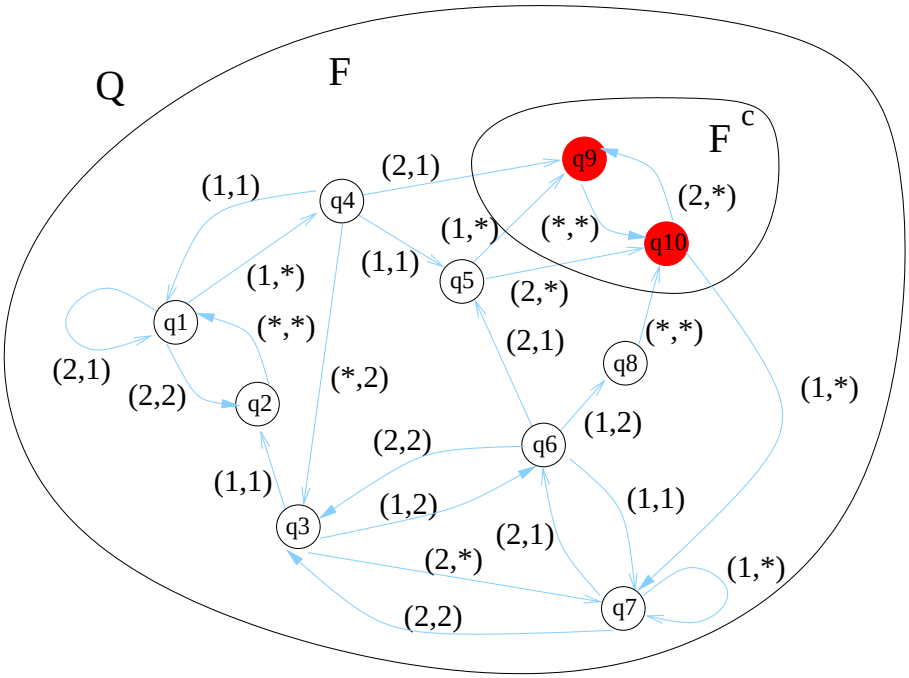

Figure 5.1. Example of discrete controller synthesis

while from q_5 and q_8 whatever u chooses to do we leave F. Therefore:

$$W^* \subseteq \{q_1, q_2, q_3, q_4, q_6, q_7\}$$

Next look at states that can leave the above set in one transition (q_4 and q_6). Notice that from q_4 if $d = 1$ whatever u chooses we leave the set, while from q_6 if $u = 1$ d can choose $d = 2$ to force us to leave the set while if $u = 2$ d can choose $d = 1$ to force us to leave the set. Therefore:

$$W^* \subseteq \{q_1, q_2, q_3, q_7\}$$

From the remaining states, if u chooses according to the following feedback map:

$$g(q) = \begin{cases} \{1\} & \text{if } q = q_7 \\ \{2\} & \text{if } q \in \{q_1, q_3\} \\ \{1, 2\} & \text{if } q = q_2 \end{cases}$$

we are guaranteed to remain in $\{q_1, q_2, q_3, q_7\}$ for ever. Therefore,

$$W^* = \{q_1, q_2, q_3, q_7\}$$

and g is the least restrictive controller that renders W^* invariant (proof of both facts is by enumeration).

More generally this scheme can be implemented by the following algorithm [14]:

Algorithm 3 (Controlled Invariant Set)
Initialization:
$W^0 = F$, $W^1 = \emptyset$, $i = 0$
p **while** $W^i \neq W^{i+1}$ **do**
begin
$W^{i-1} = W^i \cap \{q \in \mathbf{Q} \ : \ \exists u \in \mathbf{U} \ \forall d \in \mathbf{D} \ \delta(x,(u,d)) \subseteq W^i\}$
$i = i - 1$
end

The index decreases as a reminder that the algorithm involves a predecessor operation. This is a real algorithm (that is to say, it can be implemented by enumerating the set of states and inputs and that it terminates after a finite number of steps) since:

$$W^{i-1} \subseteq W^i \text{ and } |W^0| = |F| \leq |\mathbf{Q}| < \infty$$

It is also instructive to formulate this procss as a game by introducing a value function:

$$J : \mathbf{Q} \times \mathbb{Z}_- \to \{0,1\} \tag{5.3}$$

Consider the difference equation:

$$\begin{aligned} J(q,0) &= \begin{cases} 1 & q \in F \\ 0 & q \in F^c \end{cases} \\ J(q,i-1) - J(q,i) &= \min\{0, \max_{u\in\mathbf{U}} \min_{d\in\mathbf{D}} [\min_{q'\in\delta(q,(u,d))} J(q',i) - J(q,i)]\} \end{aligned} \tag{5.4}$$

We will refer to equation (5.24) as a *discrete Hamilton-Jacobi* equation.

Proposition 5.1 (Winning States for u) *A fixed point $J^* : \mathbf{Q} \to \{0,1\}$ of (5.24) is reached in a finite number of iterations. The set of states produced by the algorithm is $W^* = \{x \in \mathbf{X} | J^*(x) = 1\}$.*

Proof: We show more generally that $W^i = \{x \in \mathbf{X} | J(x,i) = 1\}$. The proof is by induction (see [110]). ∎

This resembles a two person zero sum game between the control and the disturbance. Consider what happens when a fixed point of (5.24) is reached. Ignoring the outermost min for the time being leads to:

$$J^*(q) = \max_{u\in\mathbf{U}} \min_{d\in\mathbf{D}} \min_{q'\in\delta(q,(u,d))} J^*(q') \tag{5.5}$$

Notice the similarity between this and (excuse the overloading of the notation):

$$J^*(q_0,x_0) = \max_g \min_d \left(\min_{\chi=(\tau,q,x,(u,d))\in\mathcal{H}_g} J(\chi) \right) \tag{5.6}$$

The equations look very similar. The only difference is that instead of having to worry about all feedback controllers, all disturbance trajectories and all possible executions we reduce the problem to a pointwise argument. Clearly equation (5.5) is much simpler to work with that equation

(6.1). This is a standard trick in dynamic programming. Equation (5.5) is a special case of what is known in dynamic programming as Bellman's equation, while the difference equation (5.24) is a form of value iteration for computing a fixed point to Bellman's equation.

What about the outer min? This is an inevitable consequence of turning the sequence argument of equation (6.1) to a pointwise argument. It is there to prevent states that have been labeled as unsafe at some point from being relabeled as safe later on. If it were not there, then in the example of Figure 5.1 state q_{10} would be labeled as safe after one step of the algorithm (i.e. $J(q_{10}, -1) = 1$). The extra min operation implements the intersection with W^i in the algorithm of the previous section, ensures the monotonicity of W^i with respect to i and guarantees termination.

The fixed point of equation (5.24) also provides a simple characterization of a least restrictive controller that renders W^* invariant:

$$g(q) = \begin{cases} \left\{u \in \mathbf{U} \ : \ \min_{d \in \mathbf{D}} \min_{q' \in \delta(q,(u,d))} J^*(q') = 1\right\} & \text{if } q \in W^* \\ \mathbf{U} & \text{if } q \in (W^*)^c \end{cases}$$

Notice that $g(q) \neq \emptyset$ for all q by construction. Any $u \in \mathbf{U}$ needs to be allowed outside W^* to make the controller least restrictive.

Summarizing:

Proposition 5.2 (Characterization of W^* and g) *W^* is the maximal controlled invariant subset of F and g is the unique, least restrictive feedback controller that renders W^* invariant.*

5.1.3 Discrete Hamilton-Jacobi Equation

Consider the finite automaton (3.1) with trajectory acceptance condition $\Omega = \Box F$, for $F \subseteq Q$ representing a safe set of states. We first describe the iteration process for calculating the set of states from which the controller can always keep the system inside F. We then show how this iteration process can be written as the difference equation of a *value* function, which we denote as the "discrete Hamilton-Jacobi equation".

State Space Partition

We define the *winning states* W^* for the controller as the subset of F from which the system (3.1) has a sequence of control actions $\sigma_1[\cdot]$ which can force the system to remain in F despite the actions of the environment $\sigma_2[\cdot]$. The set W^* can be calculated as the fixed point of the following iteration (where a negative index $i \in \mathbb{Z}_-$ is used to indicate that each step is a predecessor operation):

$$\begin{aligned} W^0 &= F \\ W^{i-1} &= W^i \cap \{q \in Q \mid \exists \sigma_1 \in \Sigma_1 \ \forall \sigma_2 \in \Sigma_2 \ \delta(q, \sigma_1, \sigma_2) \subseteq W^i\} \end{aligned} \tag{5.7}$$

The iteration terminates when $W^i = W^{i-1} \triangleq W^*$. At each step of the iteration, $W^{i-1} \subseteq W^i$, thus due to the assumption of the finiteness of $|Q|$, the iteration terminates in a finite number of steps. The set W^i contains those states for which the controller has a sequence of actions $\sigma_1[i]\sigma_1[i+1]\dots\sigma_1[0]$ which will ensure that the system remains in F for at least i steps, for all possible actions $\sigma_2[\cdot] \in \Sigma_2$.

The Value Function

Define the *value function* for this system as

$$J(q,i) : Q \times \mathbb{Z}_- \to \{0,1\} \tag{5.8}$$

such that

$$J(q,i) = \begin{cases} 1 & q \in W^i \\ 0 & q \in (W^i)^c \end{cases} \tag{5.9}$$

Therefore, $W^i = \{q \in Q \mid J(q,i) = 1\}$. Since the controller tries to keep the system inside F while the environment tries to force the system out of F,

$$\max_{\sigma_1 \in \Sigma_1} \min_{\sigma_2 \in \Sigma_2} \min_{q' \in \delta(q,\sigma_1,\sigma_2)} J(q',i) = \begin{cases} 1 & \text{if } \exists \sigma_1 \in \Sigma_1 \forall \sigma_2 \in \Sigma_2, \delta(q,\sigma_1,\sigma_2) \subseteq W^i \\ 0 & \text{otherwise} \end{cases} \tag{5.10}$$

The "$\min_{q' \in \delta(q,\sigma_1,\sigma_2)}$" in the above compensates for the nondeterminism in δ; the order of operations $\max_{\sigma_1} \min_{\sigma_2}$ means that the controller *plays first*, trying to maximize the minimum value of $J(\cdot)$. The environment has the advantage in this case, since it has "prior" knowledge of the controller's action when making its own choice. Therefore, in general,

$$\max_{\sigma_1 \in \Sigma_1} \min_{\sigma_2 \in \Sigma_2} \min_{q' \in \delta(q,\sigma_1,\sigma_2)} J(\cdot) \leq \min_{\sigma_2 \in \Sigma_2} \max_{\sigma_1 \in \Sigma_1} \min_{q' \in \delta(q,\sigma_1,\sigma_2)} J(\cdot) \tag{5.11}$$

with equality occurring when the action (σ_1, σ_2) is a *saddle solution*, or a *no regret* solution for each player. Here, we do not need to assume the existence of a saddle solution, rather we always give advantage to the environment, the player doing its worst to drive the system out of F, in order to ensure a conservative solution.

The iteration process (5.7) may be summarized by the difference equation:

$$J(q,i-1) - J(q,i) = \min\{0, \max_{\sigma_1 \in \Sigma_1} \min_{\sigma_2 \in \Sigma_2} [\min_{q' \in \delta(q,\sigma_1,\sigma_2)} J(q',i) - J(q,i)]\} \tag{5.12}$$

which describes the relationship between the change in $J(\cdot)$ due to one step of the iteration and the change in $J(\cdot)$ due to one state transition. We call equation (5.24) the "discrete Hamilton-Jacobi equation" for this reason. The first "min" in the equation ensures that states outside W^i that can be forced by the controller to transition into W^i are prevented from

appearing in W^{i-1}. This means that once a state has associated to it a value of zero, the value stays at zero for all subsequent iterations: enforcing the requirement that "once a state becomes unsafe, it remains unsafe".

Proposition 5.3 (Winning States W^*) *A fixed point $J^*(q)$ of (5.24) is reached in a finite number of steps. The set of winning states for the controller is $W^* = \{q \in Q \mid J^*(q) = 1\}$.*

Proof: First note that, by induction on equation (5.24), once $J(q,i) = 0$ for some i, then $J(q,j) = 0$ for $j < i$. That the fixed point $J^*(q)$ is reached in a finite number of steps follows from this and the fact that $|Q|$ is finite.

Suppose that the fixed point is reached at $i = k$. Let q be a winning state. Thus the controller has a sequence of actions which ensures that the system, starting at q, remains in F for at least k steps. Thus $q \in W^k$. Thus $q \in \{q \in Q \mid J^*(q) = 1\}$. Therefore, $W^* \subseteq \{q \in Q \mid J^*(q) = 1\}$.

Now suppose that $q \in \{q \in Q \mid J^*(q) = 1\}$, and the environment has a sequence of actions which drives the system out of F. Thus, for some $i \in \{0, -1, \ldots, k\}$,

$$\max_{\sigma_1 \in \Sigma_1} \min_{\sigma_2 \in \Sigma_2} \min_{q' \in \delta(q,\sigma_1,\sigma_2)} J(q', i+1) = 0$$

which implies, from equation (5.24) that $J(q,i) = 0$. This in turn implies that $J(q,j) = 0$ for $j < i$. Thus $J^*(q) = 0$, which is a contradiction. Therefore, $\{q \in Q \mid J^*(q) = 1\} \subseteq W^*$. ∎

A feedback controller for σ_1 that renders W^* invariant can now be constructed. For all $q \in W^*$ the controller allows only the $\sigma_1 \in \Sigma_1$ for which:

$$\min_{\sigma_2 \in \Sigma_2} \min_{q' \in \delta(q,\sigma_1,\sigma_2)} J^*(q') = 1$$

Existence of such σ_1 for all $q \in W^*$ is guaranteed by construction.

Proposition 5.4 (Characterization of W^*) *W^* is the largest controlled invariant subset of F.*

5.2 Controller Synthesis for Continuous State Systems using Optimal Control and Games

Controller synthesis begins from using the theory of optimal control. This is a large and involved topic in itself. We have attempted to give a short summary of this in Appendix B and we will repeat some of the main results to help the exposition of this chapter. We begin with optimal control, where the formulation begins with minimizing

$$J(x,u) = \phi(x(t_f)) + \int_{t_0}^{t_f} L(x(t), u(t))dt \tag{5.13}$$

subject to:

$$\dot{x}(t) = f(x(t), u(t)), \quad x(t_0) = x_0 \tag{5.14}$$

$$\psi(x(t_f)) = 0 \tag{5.15}$$

where $L : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$, $\phi : \mathbb{R}^n \to \mathbb{R}$, $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ Lipschitz continuous in x and continuous in u and $\psi : \mathbb{R}^n \to \mathbb{R}$. In Appendix B, we review how how this problem can be related to the solution of a partial differential equation, known as the Hamilton-Jacobi-Bellman equation, by introducing the *value function* (or "cost to go" function), $J^* : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$, given by:

$$J^*(x,t) = \min_{u(\cdot)} \left\{ \phi(x(t_f)) + \int_t^{t_f} L(x(\tau), u(\tau)) d\tau \right\}$$

We argued that if a continuously differentiable solution to the Hamilton-Jacobi-Bellman partial differential equation [111]:

$$\frac{\partial J^*}{\partial t}(x,t) = -\min_{u \in \mathbf{U}} \left\{ \frac{\partial J^*}{\partial x}(x,t) f(x,u) + L(x,u) \right\} \tag{5.16}$$

with boundary condition $J^*(x,0) = \phi(x)$ on $\psi(x) = 0$ can be found, then the optimal controls are given by:

$$u^*(x,t) = \arg\min_{u \in \mathbf{U}} \left\{ \frac{\partial J^*}{\partial x} f(x,u) + L(x,u) \right\}$$

Equation (B.19) can be written more compactly if we introduce the Hamiltonian, $J^* : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$:

$$H(x,p,u) = L(x,u) + p^T f(x,u)$$

and define the optimal Hamiltonian as:

$$H^*(x,p) = \min_{u \in \mathbf{U}} H(x,p,u)$$

Then equation (B.19) becomes:

$$\frac{\partial J^*}{\partial t} = -H^* \left(x, \frac{\partial J^*}{\partial x}(x,t) \right)$$

Notice that we have effectively turned the problem of optimizing a cost over the space of curves (an infinite dimensional vector space), to optimizing a cost pointwise over a finite dimensional vector space. Of course to achieve this we are still required to solve a partial differential equation.

Remarks:

1. The solution is close to the solution obtained through the calculus of variations. Simply replace the co-state p by $\frac{\partial J^*}{\partial x}(x,t)$.
2. On the positive side, dynamic programming (unlike the calculus of variations) inherently leads to feedback solutions.

3. On the negative side, dynamic programming requires one to assume differentiability of the value functions.

4. Differentiability is difficult to guarantee. Even if all the data is "smooth", the solution to the PDE may still develop "corners" (known as *shocks*). The situation is exasperated by the fact that the min operation is continuous but not smooth.

5. Optimal controls are often *bang-bang*. Consider the system:

$$\begin{array}{rcl} \dot{x} & = & f(x) + g(x)u \quad \text{(affine in } u\text{)} \\ L & : & \mathbb{R}^n \to \mathbb{R} \quad \text{(independent of } u\text{)} \\ u & \in & [U_1, U_2] \quad \text{(compact control set)} \end{array}$$

   Then:

$$H\left(x, \frac{\partial J^*}{\partial x}(x,t), u\right) = L(x) + \frac{\partial J^*}{\partial x}(x,t)f(x) + \frac{\partial J^*}{\partial x}(x,t)g(x)u$$

   therefore:

$$u^*(x,t) = \begin{cases} U_1 & \text{if } \frac{\partial J^*}{\partial x}(x,t)g(x) > 0 \\ [U_1, U_2] & \text{if } \frac{\partial J^*}{\partial x}(x,t)g(x) = 0 \\ U_2 & \text{if } \frac{\partial J^*}{\partial x}(x,t)g(x) < 0 \end{cases}$$

   Notice that u^* switches between its extreme values whenever $\frac{\partial J^*}{\partial x}(x,t)g(x)$ changes sign. Therefore, even if J^* is continuously differentiable, H^* is continuous, but not continuously differentiable.

6. If $\mathbf{U}$ is not compact, then the optimal controls may be undefined pointwise (consider for example the previous situation with $\mathbf{U} = (-\infty, \infty)$ or $\mathbf{U} = (U_1, U_2)$).

7. The situation may be even worse if $\mathbf{U}$ is not convex. The optimal controls may only be defined in the space of relaxed controls, which are not piecewise continuous as a function of time [112].

8. Finally, both dynamic programming ad the calculus of variations depend on local arguments (small perturbations about the optimal solution). For certain systems these arguments may fail. For example, there may be a curve connecting the initial point to a desired final point, but no neighboring curves have this property. This leads to *abnormal extremals*, which are not captured by the above arguments and have to be studied separately.

5.2.1 The Theory of Games

Game theory is related to optimal control, with the difference that there are two player (the control variables are divided into two classes) with possibly

conflicting objectives. The simplest case is the *two player, zero sum game*. Consider again a curve:

$$(x, u, d) : \mathbb{R} \to \mathbb{R}^n \times \mathbb{R}^{m_u} \times \mathbb{R}^{m_d}$$

Assume that d is trying to minimize and u is trying to maximize the function:

$$J(x, u, d) = \phi(x(t_f)) + \int_{t_0}^{t_f} L(x(t), u(t), d(t))dt$$

subject to:

$$\dot{x}(t) = f(x(t), u(t), d(t)), \quad x(t_0) = x_0 \tag{5.17}$$

$$\psi(x(t_f)) = 0 \tag{5.18}$$

Definition 5.5 *A pair (u^*, d^*) is called a* **saddle equilibrium** *if for all u, d:*

$$J(x, u, d^*) \leq J(x, u^*, d^*) \leq J(x, u^*, d)$$

Dynamic programming arguments can also be used to characterize saddle equilibria [113]. As before introduce a Hamiltonian:

$$H(x, p, u, d) = L(x, u, d) + p^T f(x, u, d)$$

Proposition 5.6 *If there exists a continuously differentiable function $J^* : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ such that:*

$$\begin{aligned} \frac{\partial J^*}{\partial t}(x,t) &= -\max_{u \in \mathbf{U}} \min_{d \in \mathbf{D}} H\left(x, \frac{\partial J^*}{\partial x}(x,t), u, d\right) \\ &= -\min_{d \in \mathbf{D}} \max_{u \in \mathbf{U}} H\left(x, \frac{\partial J^*}{\partial x}(x,t), u, d\right) \\ &= H\left(x, \frac{\partial J^*}{\partial x}(x,t), u^*, d^*\right) \end{aligned}$$

then (u^*, d^*) is a saddle equilibrium.

The proof is similar to the one given in the optimal control case, and relies on the fact that by freezing u to u^* we turn the problem to an optimal control problem for d (and vice versa).

Remarks:

1. The above partial differential equation is known as the *Isaacs equation*, and the min max = max min requirement is known as the *Isaacs condition*.
2. Saddle equilibria do not always exist.
3. A variational formulation is also possible.
4. The same remarks about convexity and compactness of the control and disturbance sets apply.

5. The generalization of the saddle equilibrium concept to multi player games is known as the *Nash equilibrium*. Assume there are N players, each trying to minimize their own cost function:

$$J_i(x, u_1, \ldots, u_N), \quad i = 1, \ldots, N$$

Definition 5.7 *$(u_1^*, \ldots, u_N^*)$ is a* **non-cooperative Nash equilibrium** *if for all i and for all u_i:*

$$J_i(x, u_1^*, \ldots, u_i, \ldots u_N^*) \geq J_i(x, u_1^*, \ldots, u_i^*, \ldots u_N^*)$$

The non-cooperative qualification needs to be introduced, since in this setting it may be possible for "gangs" to form: players may be able to improve their returns by collaborating with other players. Note that the saddle equilibrium concept is implicitly non-cooperative, because the game is zero sum.

The solution concept we have in mind for controller synthesis is not as symmetric as the saddle equilibrium, since we give the benefit of the doubt to the disturbance. The interpretation is that the control picks its strategy and lets the disturbance know about it. The disturbance then does its best to damage the controllers plans using this information. Consider a two player game with cost functionals J_1 and J_2 that players 1 and 2 are trying to minimize respectively. Assume that player 1 is the *leader*, i.e. decides on a strategy and lets player 2 know before player 2 has to make a decision. Given a strategy, g_1, for player 1 define the rational responses of player 2 as the set of strategies:

$$R_2(g_1) = \{g \ : \ J_2(g_1, g) \leq J_2(g_1, g') \text{ for all } g_2\}$$

Definition 5.8 *g_1^* is a* **Stackelberg equilibrium strategy** *for the leader if:*

$$\max_{g \in R_2(g_1^*)} J_1(g_1^*, g) = \min_{g_1} \max_{g \in R_2(g_1)} J_1(g_1, g)$$

In general, Stackelberg equilibrium strategies are difficult to compute in feedback form, since the concept is prone to "side payments", "incentives" and "threats". These are all techniques that can be employed by the leader to make it more appealing for player 2 to adopt a certain strategy. Fortunately, the games considered for controller synthesis will be zero sum ($J_1 = -J_2$) which ensures means like that can not be employed and the solution can be computed in feedback form using dynamic programming.

5.3 Reachability for nonlinear, continuous time systems

Consider a continuous time control system,

$$\begin{cases} \dot{x} = f(x,u) \\ x \in \mathbb{R}^n \\ u \in U \subseteq \mathbb{R}^m \\ f(\cdot,\cdot) : \mathbb{R}^n \times U \to \mathbb{R}^n, \end{cases} \tag{5.19}$$

and an arbitrary time horizon, $T \geq 0$. Let $\mathcal{U}_{[t,t']}$ denote the set of Lebesgue measurable functions from the interval $[t,t']$ to U. To eliminate technical difficulties we impose the following standing assumption.

Assumption 5.9 *$U \subseteq \mathbb{R}^m$ is compact. f is bounded and Lipschitz continuous.*

Under Assumption 5.9 the control system

$$\begin{cases} \dot{x} = f(x,u) \\ x(t) = x \in \mathbb{R}^n \end{cases} \tag{5.20}$$

admits a unique solution $x(\cdot) : [t,T] \to \mathbb{R}^n$ for all $t \in [0,T]$, $x \in \mathbb{R}^n$ and $u(\cdot) \in \mathcal{U}_{[t,T]}$. For $\tau \in [t,T]$ we will use

$$\phi(\tau, t, x, u(\cdot)) = x(\tau)$$

to denote this solution. Let $C_f > 0$ be a bound such that for all $x, \hat{x} \in \mathbb{R}^n$ and for all $u \in U$,

$$|f(x,u)| \leq C_f \text{ and } |f(x,u) - f(\hat{x},u)| \leq C_f |x - \hat{x}|.$$

Clearly, under Assumption 5.9 such bounds exist.

Given the control system of equation (A.2), the horizon $T \geq 0$ and a set of states $K \subseteq \mathbb{R}^n$, a number of questions can be naturally formulated regarding the relation between the set K and the state trajectories of (A.2) over the horizon T. Problems of interest include the following.

Viability Does there exist a choice of $u(\cdot) \in \mathcal{U}_{[0,T]}$ for which the trajectory $x(\cdot)$ satisfies $x(t) \in K$ for all $t \in [0,T]$? Clearly this is impossible if initially $x(0) \in K^c$. It may also be impossible, however, for some initial states $x(0) \in K$.

Invariance Do the trajectories $x(\cdot)$ for all $u(\cdot) \in \mathcal{U}_{[0,T]}$ satisfy $x(t) \in K$ for all $t \in [0,T]$? Once again, this is impossible if initially $x(0) \in K^c$, but may also be impossible for some initial states $x(0) \in K$.

Reachability Does there exist a $u(\cdot) \in \mathcal{U}_{[0,T]}$ and a $t \in [0,T]$ such that the trajectory satisfies $x(t) \in K$? This is trivially possible if $x(0) \in K$. It may also be possible, however, for some $x(0) \in K^c$.

As usual, K^c stands for the complement of the set K in $\mathbb{R}^n$. One would typically like to characterize the set of initial states for which the answer to the viability/invariance/reachability questions is "yes". Or, more generally, one would like to characterize the sets

$$\begin{aligned}
\text{Viab}(t, K) &= \{x \in \mathbb{R}^n \mid \exists u(\cdot) \in \mathcal{U}_{[t,T]} \; \forall \tau \in [t, T] \; \phi(\tau, t, x, u(\cdot)) \in K\} \\
\text{Inv}(t, K) &= \{x \in \mathbb{R}^n \mid \forall u(\cdot) \in \mathcal{U}_{[t,T]} \; \forall \tau \in [t, T] \; \phi(\tau, t, x, u(\cdot)) \in K\} \\
\text{R}each(t, K) &= \{x \in \mathbb{R}^n \mid \exists u(\cdot) \in \mathcal{U}_{[t,T]} \; \exists \tau \in [t, T] \; \phi(\tau, t, x, u(\cdot)) \in K\},
\end{aligned}$$

Exercise 5.1 *Show that* $\text{R}each(t, K) = (Inv(t, K^c))^c$. *Therefore, the invariance and reachability problems are duals of one another and need not be treated separately.*

5.3.1 Reachability through the Optimal Control Perspective

Our treatment follows [110] and [36].

Recall that continuous systems are a special class of hybrid systems with:

- $Q = \{q\}$, $\mathbf{Q} = \{q_0\}$ (trivial discrete state);
- $X = \{x\}$, $\mathbf{X} = \mathbb{R}^n$;
- $V = \{u, d\}$, $\mathbf{U} \subseteq \mathbb{R}^{m_u}$ and $\mathbf{D} \subseteq \mathbb{R}^{m_d}$ (no discrete inputs);
- I$nit \subseteq \{q_0\} \times \mathbf{X}$ (drop the trivial dependence on the discrete state from now on);
- $f : \mathbf{X} \times \mathbf{U} \times \mathbf{D} \to \mathbb{R}^n$;
- $I(q) = \mathbf{X}$;
- $E = \emptyset$ (no discrete dynamics);
- $\phi(q_0, x) = \mathbf{V}$ (no state dependent input constraints).

Dropping the trivial discrete dynamics, the system can be more compactly characterized by an ordinary differential equation:

$$\begin{array}{rcl}
\dot{x}(t) & = & f(x(t), u(t), d(t)) \\
x(0) & = & x_0 \\
u(t) & \in & \mathbf{U} \\
d(t) & \in & \mathbf{D}
\end{array} \tag{5.21}$$

To prevent technical problems, we introduce the following assumption:

Assumption 5.10 *f is Lipschitz continuous in x and continuous in u and d. u and d are piecewise continuous as functions of time. $\mathbf{U}$ and $\mathbf{D}$ are convex and compact subsets of $\mathbb{R}^{m_u}$ and $\mathbb{R}^{m_d}$ respectively.*

We use $\mathcal{U}_I$ to denote the set of piecewise continuous functions from an in interval $I \subseteq \mathbb{R}$ to $\mathbf{U}$ and $\mathcal{U}$ for the union of $\mathcal{U}_I$ over all I (and similarly for $\mathcal{D}$). The assumptions on f, u and d are needed to ensure that the system is

well posed. Under this assumption the system is (in a sense) deterministic and non-blocking, since, by a standard existence and uniqueness argument for ordinary differential equations, for every $x_0 \in \mathbb{R}^n$, $T \geq 0$, $u \in \mathcal{U}_{[0,T]}$ and $d \in \mathcal{D}_{[0,T]}$, there exists a continuous, piecewise differentiable function $x : [0, T] \to \mathbf{X}$ with $x(0) = x_0$ and $\dot{x}(t) = f(x(t), u(t), d(t))$ for all t where (u, d) is continuous ("almost everywhere"). For this reason we will use:

$$\chi = (x_0, u, d)$$

to denote the unique execution starting at $x_0 \in \mathbb{R}^n$ under control $u \in \mathcal{U}$ and disturbance $d \in \mathcal{D}$.

In addition, the piecewise continuity assumption prevents the controller from "cheating" by forcing the execution to be Zeno. Strictly speaking, the execution of the system will have to be defined over a sequence of intervals, where u and d are continuous over each interval. Piecewise continuity of u and d implies that there exists an execution such that every compact interval of time overlaps with a finite number of such intervals, or, in other words, there can not be an infinite number of "transitions" (in this case discontinuities in u and d) in a finite amount of time.

The assumptions on $\mathbf{U}$ and $\mathbf{D}$ will be needed to prevent technical problems when posing the optimal control and gaming problems. Consider a set of states $F \subseteq \mathbf{Q}$. Try to establish the maximal control invariant subset of F, i.e. the largest set of initial states for which there exists a controller that manages to keep all executions inside F. Somewhat informally this set can be characterized as:

$$W^* = \{x_0 \in \mathbb{R}^n \ : \ \exists u \in \mathcal{U} \ \forall d \in \mathcal{D}, \ \Box F(x_0, u, d) = \text{True}\}$$

The only additional caveat is that u is implemented by a memoryless controller. To eliminate technical complications we assume that:

Assumption 5.11 *There exists a continuously differentiable function* $l : \mathbb{R}^n \to \mathbb{R}$ *such that:*

$$\begin{array}{lll} l(x) > 0 & \textit{if} & x \in F^o \\ l(x) = 0 & \textit{if} & x \in \partial F \\ l(x) < 0 & \textit{if} & x \in F^c \\ l(x) = 0 & \Rightarrow & \frac{\partial l}{\partial x}(x) \neq 0 \end{array}$$

The assumption implies that F is a closed set with non-empty interior, whose boundary is a $n-1$ dimensional manifold.

Dynamic Programming Solution

To apply the optimal control tools introduced in the previous section let $t_f = 0$, consider an arbitrary $t \leq 0$ and introduce the value function:

$$\hat{J}(x, t) = \max_{u \in \mathcal{U}_{[t,0]}} \min_{d \in \mathcal{D}_{[t,0]}} l(x(0))$$

Notice that this optimal control problem involves no Lagrangian ($L \equiv 0$), just a terminal cost. The game obtained in this setting falls in the class of two person zero sum games studied above. The (u, d) obtained by the above optimal control problem is also a Stackelberg equilibrium for the game between control and disturbance, with the control playing the role of the leader. Recall that in general the computation of Stackelberg equilibria in feedback form may be complicated by the possibility of "incentives" and "threats", employed by the leader to coerce the other player into a beneficial course of action. In this case the situation is simplified by the fact that the game is zero sum (any gain achieved by u is equal to a loss suffered by d), so the possibility of incentives and threats is eliminated.

$\hat{J}$ can be computed using the dynamic programming tools discussed in Appendix B. Introduce a Hamiltonian:

$$\begin{array}{rcl} H : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^{m_u} \times \mathbb{R}^{m_d} & \longrightarrow & \mathbb{R} \\ (x, p, u, d) & \longmapsto & p^T f(x, u, d) \end{array}$$

Consider the optimal Hamiltonian:

$$H^*(x, p) = \max_{u \in \mathbf{U}} \min_{d \in \mathbf{D}} H(x, p, u, d)$$

Notice again that the minimization over u and d is pointwise, as opposed to over functions of time. Then, if $\hat{J}$ is continuously differentiable it satisfies:

$$\begin{array}{rcl} \frac{\partial \hat{J}}{\partial t}(x, t) & = & -H^* \left(x, \frac{\partial \hat{J}}{\partial x}(x, t) \right) \\ \hat{J}(x, 0) & = & l(x) \end{array} \tag{5.22}$$

Notice that the evolution of the partial differential equation is "backwards" in time.

Consider the set:

$$\widehat{W}_t = \{x_0 \in \mathbf{X} \ : \ \hat{J}(x_0, t) \geq 0\}$$

This is the set of all states for which starting at $x(t) = x_0$, there exists a controller for u such that for all disturbance trajectories $d \in \mathcal{D}_{[t,0]}$, $l(x(0)) \geq 0$ or, in other words, $x(0) \in F$. This is not quite what we need yet. We would like the set of all states for which there exists a u such that for all d and for all $t' \in [t, 0]$, $x(t') \in F$. This excludes points in $\widehat{W}_t$ which leave F at some point in $[t, 0]$ but re-enter it before time 0. This requirement can be encoded either after the computation of $\hat{J}$, or by modifying the Hamilton-Jacobi equation:

$$\begin{array}{rcl} \frac{\partial J}{\partial t}(x, t) & = & -H^* \left(x, \frac{\partial J}{\partial x}(x, t) \right) \\ J(x, 0) & = & l(x) \end{array} \tag{5.23}$$

Compare this with the discrete Hamilton-Jacobi equation:

$$\begin{aligned} J(q,0) &= \begin{cases} 1 & q \in F \\ 0 & q \in F^c \end{cases} \\ J(q,i-1) - J(q,i) &= \min\{0, \max_{u\in\mathbf{U}} \min_{d\in\mathbf{D}} [\min_{q'\in\delta(q,(u,d))} J(q',i) - J(q,i)]\} \end{aligned} \tag{5.24}$$

$\delta(q,(u,d))$ essentially implements the spatial partial derivative of J along the dynamics of the system. The innermost minimization is not needed in the continuous case, as the continuous system is "deterministic". As before, we seek a stationary solution to this equation. Assume that as $t \to -\infty$, $J(x,t)$ converges to a continuously differentiable function $J^* : \mathbf{X} \to \mathbb{R}$.

Proposition 5.12 *The set $W^* = \{x \in \mathbf{X} \ : \ J^*(x) \geq 0\}$ is the largest controlled invariant set contained in F.*

The solution to the partial differential equation also leads to a least restrictive controller that renders W^* invariant. Consider:

$$g(x) = \begin{cases} \left\{ u \in \mathbf{U} \ : \ \min_{d\in\mathbf{D}} \left(\frac{\partial J^*(x)}{\partial x}\right)^T f(x,u,d) \geq 0 \right\} & \text{if } x \in \partial W^* \\ \mathbf{U} & \text{if } x \in (W^*)^\circ \cup (W^*)^c \end{cases} \tag{5.25}$$

Proposition 5.13 *g is the unique, least restrictive memoryless controller that renders W^* invariant.*

Notice that no constraint is imposed on u in the interior of W^* (we can delay action until we reach the boundary) and outside W^* (its too late to do anything about it, so might as well give up!).

Geometric interpretation

For an arbitrary time $t \leq 0$ define:

$$W_t = \{x \in \mathbf{X} \ : \ J(x,t) \geq 0\}$$

Consider and $x \in \partial W_t$ and assume that:

$$\begin{aligned} & H^*\left(x, \frac{\partial J}{\partial x}(x,t)\right) < 0 \\ \Leftrightarrow \quad & \max_{u\in\mathbf{U}} \min_{d\in\mathbf{D}} H\left(x, \frac{\partial J}{\partial x}(x,t), u, d\right) < 0 \\ \Leftrightarrow \quad & \max_{u\in\mathbf{U}} \min_{d\in\mathbf{D}} \frac{\partial J}{\partial x}(x,t) f(x,u,d) < 0 \\ \Leftrightarrow \quad & \forall u \in \mathbf{U} \ \ \exists d \in \mathbf{D} \text{ such that } \frac{\partial J}{\partial x}(x,t) f(x,u,d) < 0 \end{aligned}$$

But $\frac{\partial J}{\partial x}(x,t)$ is the normal to the boundary of W_t at x, pointing inside W_t. Moreover, $\frac{\partial J}{\partial x}(x,t) f(x,u,d)$ is the inner product between this normal and

the vector $f(x,u,d)$. Let θ be the angle between $\frac{\partial J}{\partial x}(x,t)$ and $f(x,u,d)$. Then:

$$\begin{aligned}
\frac{\partial J}{\partial x}(x,t) f(x,u,d) > 0 \quad &\text{if} \quad \theta < \pi/2 \\
\frac{\partial J}{\partial x}(x,t) f(x,u,d) = 0 \quad &\text{if} \quad \theta = \pi/2 \\
\frac{\partial J}{\partial x}(x,t) f(x,u,d) < 0 \quad &\text{if} \quad \theta > \pi/2
\end{aligned}$$

Therefore, the above statement is equivalent to:

> for all $u \in \mathbf{U}$ there exists $d \in \mathbf{D}$ such that the normal to ∂W_t at x pointing towards the interior of W_t makes an angle greater than $\pi/2$ with $f(x,u,d)$,

or, equivalently:

> for all $u \in \mathbf{U}$ there exists $d \in \mathbf{D}$ such that $f(x,u,d)$ points outside W_t.

These are points where whatever u does d can force them to leave the set W_t instantaneously. Notice that the order of the quantifiers in the above expression implies that d may depend on u, in addition to x and t. The part of the boundary of W_t where $H^* < 0$ is known as the "usable part" in the pursuit-evasion game literature.

Returning back to the Hamilton Jacobi equation, we see that for these points:

$$\begin{aligned}
\frac{\partial J}{\partial t}(x,t) &= -\min\left\{0, H^*\left(x, \frac{\partial J}{\partial x}(x,t)\right)\right\} \\
&= -H^*\left(x, \frac{\partial J}{\partial x}(x,t)\right) \\
&> 0
\end{aligned}$$

Therefore, as t decreases, J also decreases. For these points on the boundary of W_t, J becomes negative instantaneously, and they "fall out of" W_t.

What if $H^* \geq 0$? A similar argument shows that in this case:

> there exists $u \in \mathbf{U}$ such that for all $d \in \mathbf{D}$ the normal to ∂W_t at x pointing towards the interior of W_t makes an angle at most $\pi/2$ with $f(x,u,d)$,

or, equivalently:

> there exists $u \in \mathbf{U}$ such that for all $d \in \mathbf{D}$, $f(x,u,d)$ either points inside W_t or is tangent to ∂W_t.

These are points for which there exists a choice or u that for all d forces the state to remain in W_t. Notice that the order of the quantifiers implies

that u may only depend x and t, and not d. For these points:

$$\begin{aligned}\frac{\partial J}{\partial t}(x,t) &= -\min\left\{0, H^*\left(x, \frac{\partial J}{\partial x}(x,t)\right)\right\} \\ &= 0\end{aligned}$$

Therefore, as t decreases, J remains constant. These are points that want to move towards the interior of W_t. The role of the outermost minimum is to ensure that the value of J does not increase for these points, so that W_t does not grow. This is to prevent states that have been labeled as unsafe (can reach F^c) from being relabeled as safe later on.

Reachability through the Viability Perspective

In this section, we will develop a method that takes advantage of optimal control tools to characterize the reachable, viable and invariant sets at level sets of value functions. For this purpose we introduce a function $l(\cdot) : \mathbb{R}^n \to \mathbb{R}$ and define two optimal control problems with value functions $V_1 : \mathbb{R}^n \times [0,T] \to \mathbb{R}$ and $V_2 : \mathbb{R}^n \times [0,T] \to \mathbb{R}$ given by

$$V_1(x,t) = \sup_{u(\cdot)\in\mathcal{U}_{[t,T]}} \min_{\tau\in[t,T]} l(\phi(\tau,t,x,u(\cdot))) \tag{5.26}$$

$$V_2(x,t) = \inf_{u(\cdot)\in\mathcal{U}_{[t,T]}} \min_{\tau\in[t,T]} l(\phi(\tau,t,x,u(\cdot))). \tag{5.27}$$

In the first problem the objective of the input u is to maximize the minimum value attained by the function l along the state trajectory over the horizon $[t,T]$. In the second problem the objective of u is to minimize this minimum. Notice that the minimum with respect to time is well defined. For obvious reasons we will subsequently refer to the first optimal control problem as the SupMin problem and to the second problem as the InfMin problem. To prevent technical difficulties we introduce a further standing assumption.

Assumption 5.14 *l is bounded and Lipschitz continuous.*

Under this assumption, let $C_l > 0$ be a bound such that for all $x, \hat{x} \in \mathbb{R}^n$

$$|l(x)| \le C_l \text{ and } |l(x) - l(\hat{x})| \le C_l|x - \hat{x}|.$$

We first establish a connection between viability and the SupMin optimal control problem. Assume that the set K is closed and is related to the zero level set of the function $l : \mathbb{R}^n \to \mathbb{R}$ by

$$K = \{x \in \mathbb{R}^n \mid l(x) \ge 0\}.$$

A natural choice for the function l is the signed distance to the set K given by

$$l(x) = \begin{cases} -d(x,K) & \text{if } x \in K^c \\ d(x,K^c) & \text{if } x \in K, \end{cases} \tag{5.28}$$

where $d(x, K)$ stands for the usual distance to the set K

$$d(x, K) = \inf_{\hat{x} \in K} |x - \hat{x}|.$$

To ensure that l satisfies Assumption 5.14 one can impose a saturation to the distance function at some value C_l.

Exercise 5.2 *Assume $K = [-1, 1] \subseteq \mathbb{R}$. Plot the function $l(x)$ of equation 5.28. Plot also the function $l(x)$ saturated at $C_l = 0.5$. Is the function $l(x)$ Lipschitz continuous? What is the Lipschitz constant?*

The following propositions establish a link between the sets Viab and Inv and the level sets of the value functions V_1 and V_2 respectively.

Proposition 5.15 *Viab$(t, K) = \{x \in \mathbb{R}^n \mid V_1(x, t) \geq 0\}$.*

Proof: Consider $x \in \text{Viab}(t, K)$. By definition there exists a $u(\cdot) \in \mathcal{U}_{[t,T]}$ such that $\phi(\tau, t, x, u(\cdot)) \in K$ for all $\tau \in [t, T]$. Note that

$$V_1(x, t) \geq \min_{\tau \in [t,T]} l(\phi(\tau, t, x, u(\cdot))) \geq 0$$

therefore $\text{Viab}_F(t, K) \subseteq \{x \in \mathbb{R}^n \mid V_1(x, t) \geq 0\}$.

Conversely, assume that $V_1(x, t) \geq 0$ and consider a sequence $\epsilon_n \geq 0$ converging to zero. Then there exist $u_n(\cdot) \in \mathcal{U}_{[t,T]}$ such that

$$\min_{\tau \in [t,T]} l(\phi(\tau, t, x, u_n(\cdot))) \geq -\epsilon_n.$$

Because the set of solution starting in a compact set is compact ([114], Theorem 3.5.2) there exists a subsequence (denoted again by $u_n(\cdot)$ for convenience) that converges to a solution $\phi(\tau, t, x, u(\cdot))$ uniformly on compact intervals. We claim that $\phi(\tau, t, x, u(\cdot)) \in K$ for all $\tau \in [t, T]$. Assume, for the sake of contradiction, that there exists some $\hat{t} \geq [t, T]$ such that $\phi(\hat{t}, t, x, u(\cdot)) \notin K$, i.e. $l(\phi(\hat{t}, t, x, u(\cdot))) < 0$. Since $l(\cdot)$ is Lipschitz and the convergence is uniform over the compact interval $[t, T]$, there exists an N such that

$$l(\phi(\hat{t}, t, x, u_n(\cdot))) \leq l(\phi(\hat{t}, t, x, u(\cdot)))/2 < 0$$

for all $n \geq N$. Therefore

$$0 > l(\phi(\hat{t}, t, x, u(\cdot)))/2 \geq \min_{\tau \in [t,T]} l(\phi(\hat{t}, t, x, u_n(\cdot))) \geq -\epsilon_n \text{ for all } n \geq N$$

which is a contradiction since the ϵ_n converge to zero. ∎

Exercise 5.3 *Show that if K is an open set and $K = \{x \in \mathbb{R}^n \mid l(x) > 0\}$, then Viab$(t, K) = \{x \in \mathbb{R}^n \mid V_1(x, t) > 0\}$.*

Proposition 5.16 *Inv$(t, K) = \{x \in \mathbb{R}^n \mid V_2(x, t) \geq 0\}$.*

Proof: By definition, $V_2(x, t) \geq 0$ is equivalent to

$$\inf_{u(\cdot) \in \mathcal{U}_{[t,T]}} \min_{\tau \in [t,T]} l(\phi(\tau, t, x, u(\cdot))) \geq 0,$$

or, in other words, for all $u(\cdot) \in \mathcal{U}_{[t,T]}$, $\min_{\tau \in [t,T]} l(\phi(\tau, t, x, u(\cdot))) \geq 0$. The last statement is in turn equivalent to "for all $u(\cdot) \in \mathcal{U}_{[t,T]}$ and for all $\tau \in [t,T]$, $l(\phi(\tau, t, x, u(\cdot))) \geq 0$", i.e. $\phi(\tau, t, x, u(\cdot)) \in K$. ∎

Exercise 5.4 *Show that if K is an open set and $K = \{x \in \mathbb{R}^n \mid l(x) > 0\}$, then $Inv(t, K) = \{x \in \mathbb{R}^n \mid V_2(x,t) > 0\}$. This requires some work, along the lines of the proof of Proposition 5.15.*

Propositions 5.15 and 5.16 suggest that any method to characterize and compute the value functions V_1 and V_2 automatically provides a method for solving reachability problems for continuous systems. In the optimal control literature the standard way for characterizing value functions is as solutions to appropriate partial differential equations. Tools for numerically solving partial differential equations can then be used to compute solutions to the optimal control problems. Here we provide such a characterization of the value functions V_1 and V_2. More specifically, we show that V_1 is a viscosity solutions of the terminal value problem

$$\frac{\partial V_1}{\partial t}(x,t) + \min\left\{0, \sup_{u \in U} \frac{\partial V_1}{\partial x}(x,t) f(x,u)\right\} = 0 \tag{5.29}$$

with $V_1(x,T) = l(x)$ over $(x,t) \in \mathbb{R}^n \times [0,T]$. Likewise, V_2 is a viscosity solution to the terminal value problem

$$\frac{\partial V_2}{\partial t}(x,t) + \min\left\{0, \inf_{u \in U} \frac{\partial V_2}{\partial x}(x,t) f(x,u)\right\} = 0 \tag{5.30}$$

with $V_2(x,T) = l(x)$ over $(x,t) \in \mathbb{R}^n \times [0,T]$.

Our treatment uses the standard definition of viscosity solution [115]. For example, we say that V_1 is a viscosity solution of (5.29) if and only if it satisfies the following three conditions:

1. $V_1(x,T) = l(x)$.
2. For all $(x_0, t_0) \in \mathbb{R}^n \times [0,T)$ and for all smooth $W : \mathbb{R}^n \times [0,T) \to \mathbb{R}$, if $V_1 - W$ attains a local maximum at (x_0, t_0), then
$$\frac{\partial W}{\partial t}(x_0,t_0) + \min\left\{0, \sup_{u \in U} \frac{\partial W}{\partial x}(x_0,t_0) f(x_0,u)\right\} \geq 0.$$
3. For all $(x_0, t_0) \in \mathbb{R}^n \times [0,T)$ and for all smooth $W : \mathbb{R}^n \times [0,T) \to \mathbb{R}$, if $V_1 - W$ attains a local minimum at (x_0, t_0), then
$$\frac{\partial W}{\partial t}(x_0,t_0) + \min\left\{0, \sup_{u \in U} \frac{\partial W}{\partial x}(x_0,t_0) f(x_0,u)\right\} \leq 0.$$

Recall that a viscosity solution is not necessarily a differentiable function. However, it can be shown that wherever the viscosity solution is differentiable it satisfies the partial differential equation in the classical sense [115].

5.3.2 Example

The following trivial example illustrates some of the points raised above. Consider the one dimensional system

$$\begin{cases} \dot{x} = u \\ x \in \mathbb{R} \\ u \in U = [1, 2]. \end{cases}$$

For this system we will study the viability and invariance of the set

$$K = (-\infty, -1] \cup [1, \infty)$$

Let

$$l(x) = x^2 - 1$$

and notice that $K = \{x \in \mathbb{R} \mid l(x) \geq 0\}$. We gloss over the technical point that l fails to satisfy Assumption 5.14; one can introduce a saturation on $l(x)$ to resolve this problem, but this would unnecessarily complicate the solution.

Given a horizon $T \geq 0$ it is easy to see that for all $t \in [0, T]$

$$\begin{aligned} \text{Viab}(t, K) &= (-\infty, -1 - (T - t)] \cup [1, \infty) \\ \text{Inv}(t, K) &= (-\infty, -1 - 2(T - t)] \cup [1, \infty). \end{aligned}$$

Obviously, since whatever u does x is forced to increase, any initial condition $x < 1$ will sooner of later have to leave K. Therefore, one expects that any $x < 1$, will sooner or later find itself outside $\text{Viab}(t, K)$ and $\text{Inv}(t, K)$, provided the horizon T is sufficiently large.

We start with viability and check what equation (5.29) predicts for this system. Consider the candidate value function

$$V_1(x, t) = \begin{cases} x^2 - 1 & x > 0 \\ -1 & -(T - t) \leq x \leq 0 \\ (x + T - t)^2 - 1 & x < -(T - t). \end{cases}$$

Exercise 5.5 *Show that V_1 is a classical solution to the terminal value problem* (5.29) *over* $\mathbb{R} \times [0, T]$.

A plot of V_1 and its sub-zero level set are shown in Figure 5.2. The solution seems to agree with our intuition: sooner or later all states to the left of $x = 1$ are doomed to end up in $\{x \in \mathbb{R}^n \mid V_1(x, t) < 0\}$ and therefore leave K.

We now turn our attention to invariance, and show how the set $\text{Inv}(t, K)$ can be characterized using the solution of appropriate partial differential equations for this example. Consider the candidate value function

$$V_2(x, t) = \begin{cases} x^2 - 1 & x > 0 \\ -1 & -2(T - t) \leq x \leq 0 \\ (x + 2(T - t))^2 - 1 & x < -2(T - t) \end{cases} \tag{5.31}$$

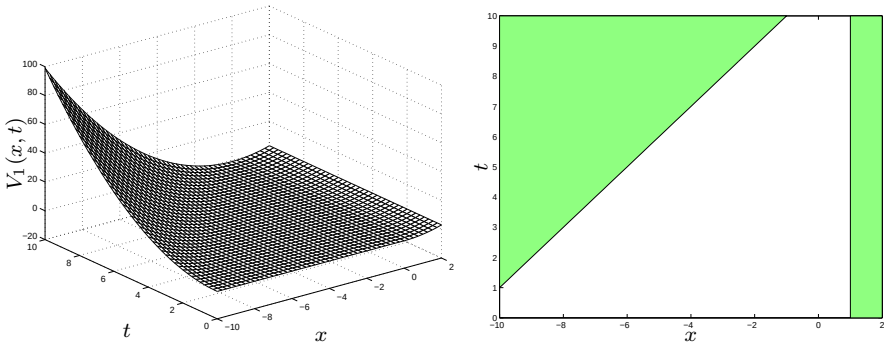

Figure 5.2. The value function $V_1(x,t)$ for $T = 10$. The dark region in the right plot is the level set $\{(x,t) \in \mathbb{R} \times [0,T] \mid V_1(x,t) > 0\}$.

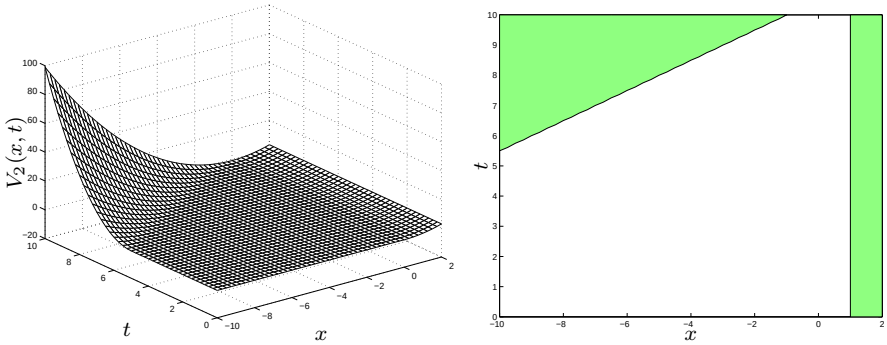

Figure 5.3. The value function $V_2(x,t)$ for $T = 10$. The dark region in the right plot is the level set $\{(x,t) \in \mathbb{R} \times [0,T] \mid V_2(x,t) \geq 0\}$.

Exercise 5.6 *Show that V_2 is a classical solution to the terminal value problem* (5.30) *over $\mathbb{R} \times [0,T]$.*

A plot of V_2 and its zero level set are shown in Figure 5.3. Once again, the solution agrees with our intuition for the set $\mathrm{Inv}(t,K)$. In Section 5.3.4 we show that this relation is true in general.

5.3.3 Solution to the SupMin *Problem*

We start by showing that V satisfies an appropriate version of the optimality principle.

Theorem 5.17 (SupMin Optimality Conditions) *For all $(x,t) \in \mathbb{R}^n \times [0,T]$ and all $h \in [0, T-t]$, $V_1(x,t) \leq V_1(x,t+h)$ and $V_1(x,T) = l(x)$. Moreover,*

$$V_1(x,t) = \sup_{u(\cdot) \in \mathcal{U}_{[t,t+h]}} \left[\min \left\{ \min_{\tau \in [t,t+h]} l(\phi(\tau,t,x,u(\cdot))), V_1(\phi(t+h,t,x,u(\cdot)),t+h) \right\} \right].$$

Proof: The fact that $V_1(x,T) = l(x)$ is immediate from the definition of V_1. Moreover,

$$V_1(x,t) = \sup_{u(\cdot)\in\mathcal{U}_{[t,T]}} \min_{\tau\in[t,T]} l(\phi(\tau,t,x,u(\cdot))), \text{ and}$$
$$V_1(x,t+h) = \sup_{u(\cdot)\in\mathcal{U}_{[t+h,T]}} \min_{\tau\in[t+h,T]} l(\phi(\tau,t+h,x,u(\cdot))).$$

Assume, for the sake of contradiction, that $V_1(x,t) > V_1(x,t+h)$. Then there exists $u_1(\cdot)\in\mathcal{U}_{[t,T]}$ such that for all $u_2(\cdot)\in\mathcal{U}_{[t+h,T]}$,

$$\min_{\tau\in[t,T]} l(\phi(\tau,t,x,u_1(\cdot))) > \min_{\tau\in[t+h,T]} l(\phi(\tau,t+h,x,u_2(\cdot))). \tag{5.32}$$

Choose $u_2(\cdot)\in\mathcal{U}_{[t+h,T]}$ according to $u_2(\tau) = u_1(\tau-h)$ for $\tau\in[t+h,T]$. By uniqueness, $\phi(\tau,t+h,x,u_2(\cdot)) = \phi(\tau-h,t,x,u_1(\cdot))$ for all $\tau\in[t+h,T]$. Therefore, by (5.32),

$$\begin{aligned}\min_{\tau\in[t,T]} l(\phi(\tau,t,x,u_1(\cdot))) &> \min_{\tau\in[t+h,T]} l(\phi(\tau,t+h,x,u_2(\cdot)))\\ &= \min_{\tau\in[t+h,T]} l(\phi(\tau-h,t,x,u_1(\cdot)))\\ &= \min_{\tau\in[t,T-h]} l(\phi(\tau,t,x,u_1(\cdot))),\end{aligned}$$

which is a contradiction. Therefore, $V_1(x,t)\le V_1(x,t+h)$.

For the second part, we show that for all $\epsilon>0$,

$$\begin{aligned}V_1(x,t) \ge \sup_{u(\cdot)\in\mathcal{U}_{[t,t+h]}} \Big[\min\Big\{ &\min_{\tau\in[t,t+h]} l(\phi(\tau,t,x,u(\cdot))),\\ &V_1(\phi(t+h,t,x,u(\cdot)),t+h)\Big\}\Big] - \epsilon\\ V_1(x,t) \le \sup_{u(\cdot)\in\mathcal{U}_{[t,t+h]}} \Big[\min\Big\{ &\min_{\tau\in[t,t+h]} l(\phi(\tau,t,x,u(\cdot))),\\ &V_1(\phi(t+h,t,x,u(\cdot)),t+h)\Big\}\Big] + \epsilon.\end{aligned}$$

Since $\epsilon>0$ is arbitrary, if these two claims hold then

$$V_1(x,t) = \sup_{u(\cdot)\in\mathcal{U}_{[t,t+h]}} \left[\min\left\{ \min_{\tau\in[t,t+h]} l(\phi(\tau,t,x,u(\cdot))), V_1(\phi(t+h,t,x,u(\cdot)),t+h)\right\}\right].$$

Case 1: Consider an arbitrary $u_1(\cdot)\in\mathcal{U}_{[t,t+h]}$. Fix $\epsilon>0$ and choose $u_2(\cdot)\in\mathcal{U}_{[t+h,T]}$ such that

$$V_1(\phi(t+h,t,x,u_1(\cdot)),t+h) \le \min_{\tau\in[t+h,T]} l(\phi(\tau,t+h,\phi(t+h,t,x,u_1(\cdot)),u_2(\cdot))+\epsilon.$$

Define $u(\cdot)\in\mathcal{U}_{[t,T]}$ by

$$u(t) = \begin{cases} u_1(t) & \text{if } t\in[t,t+h)\\ u_2(t) & \text{if } t\in[t+h,T].\end{cases}$$

By uniqueness,

$$\phi(\tau,t,x,u(\cdot)) = \begin{cases} \phi(\tau,t,x,u_1(\cdot)) & \text{if } \tau \in [t,t+h) \\ \phi(\tau,t+h,\phi(t+h,t,x,u_1(\cdot)),u_2(\cdot)) & \text{if } \tau \in [t+h,T]. \end{cases}$$

By definition of the value function

$$\begin{aligned} V_1(x,t) &\geq \min_{\tau\in[t,T]} l(\phi(\tau,t,x,u(\cdot))) \\ &= \min\left\{ \min_{\tau\in[t,t+h]} l(\phi(\tau,t,x,u(\cdot))), \min_{\tau\in[t+h,T]} l(\phi(\tau,t,x,u(\cdot))) \right\} \\ &= \min\left\{ \min_{\tau\in[t,t+h]} l(\phi(\tau,t,x,u_1(\cdot))), \right. \\ &\qquad \left. \min_{\tau\in[t+h,T]} l(\phi(\tau,t+h,\phi(t+h,t,x,u_1(\cdot)),u_2(\cdot))) \right\} \\ &\geq \min\left\{ \min_{\tau\in[t,t+h]} l(\phi(\tau,t,x,u_1(\cdot))), \right. \\ &\qquad \left. V_1(\phi(t+h,t,x,u_1(\cdot)),t+h) - \epsilon \right\} \\ &\geq \min\left\{ \min_{\tau\in[t,t+h]} l(\phi(\tau,t,x,u_1(\cdot))), V_1(\phi(t+h,t,x,u_1(\cdot)),t+h) \right\} - \epsilon. \end{aligned}$$

But $u_1(\cdot)$ is arbitrary, therefore,

$$V_1(x,t) \geq \sup_{u(\cdot)\in\mathcal{U}_{[t,t+h]}} \left[\min\left\{ \min_{\tau\in[t,t+h]} l(\phi(\tau,t,x,u(\cdot))), V_1(\phi(t+h,t,x,u(\cdot)),t+h) \right\} \right] - \epsilon.$$

Case 2: Fix $\epsilon > 0$ and select $u(\cdot) \in \mathcal{U}_{[t,T]}$ such that

$$V_1(x,t) \leq \min_{\tau\in[t,T]} l(\phi(\tau,t,x,u(\cdot))) + \epsilon.$$

By definition

$$V_1(\phi(t+h,t,x,u(\cdot)),t+h) \geq \min_{\tau\in[t+h,T]} l(\phi(\tau,t,x,u(\cdot))).$$

Therefore,

$$\begin{aligned} V_1(x,t) &\leq \min\left\{ \min_{\tau\in[t,t+h]} l(\phi(\tau,t,x,u(\cdot))), \min_{\tau\in[t+h,T]} l(\phi(\tau,t,x,u(\cdot))) \right\} + \epsilon \\ &\leq \min\left\{ \min_{\tau\in[t,t+h]} l(\phi(\tau,t,x,u(\cdot))), V_1(\phi(t+h,t,x,u(\cdot)),t+h) \right\} + \epsilon \\ &\leq \sup_{u(\cdot)\in\mathcal{U}_{[t,t+h]}} \left[\min\left\{ \min_{\tau\in[t,t+h]} l(\phi(\tau,t,x,u(\cdot))), V_1(\phi(t+h,t,x,u(\cdot)),t+h) \right\} \right] + \epsilon. \end{aligned}$$

■

Theorem 5.17 makes two assertions. The first is that the "value" of a given state x can only decrease as the "time to go" increases. Starting from x the minimum value that l experiences over a certain time horizon is less than

or equal to the minimum value that l would experience if we stopped the evolution at any time before the horizon expires. It is this property that forces the level sets of V_1 to change monotonically as t decreases (see, for example, Figure 5.2). It is also the reason why the extra min was introduced in the standard Hamilton-Jacobi-Bellman equation to produce the terminal value problem (5.29). In particular, the two statements of the first part together imply that for all $(x,t) \in \mathbb{R}^n \times [0,T]$

$$V_1(x,t) \leq l(x). \tag{5.33}$$

The second part of the theorem is a variant of the standard principle of optimality: it relates the optimal cost to go from (x,t) to the optimal cost to go from $(x(t+h), t+h)$ and the minimum value experienced by l over the interval $[t, t+h]$.

Next, we show that under Assumption 5.14 the value function V_1 is "nice".

Lemma 5.18 *There exists a constant $C > 0$ such that $|V_1(x,t)| \leq C$ and $|V_1(x,t) - V_1(\hat{x},\hat{t})| \leq C(|x-\hat{x}| + (t-\hat{t}))$, for all $(x,t),(\hat{x},\hat{t}) \in \mathbb{R}^n \times [0,T]$.*

Proof: V_1 is bounded since l is bounded.

Fix $x, \hat{x} \in \mathbb{R}^n$ and $t \in [0,T]$. Consider $\epsilon > 0$ and choose $\hat{u}(\cdot) \in \mathcal{U}_{[t,T]}$ such that

$$V_1(\hat{x},t) \leq \min_{\tau \in [t,T]} l(\phi(\tau, t, \hat{x}, \hat{u}(\cdot))) + \epsilon.$$

By definition,

$$V_1(x,t) \geq \min_{\tau \in [t,T]} l(\phi(\tau, t, x, \hat{u}(\cdot))).$$

Therefore,

$$V_1(\hat{x},t) - V_1(x,t) \leq \min_{\tau \in [t,T]} l(\phi(\tau, t, \hat{x}, \hat{u}(\cdot))) - \min_{\tau \in [t,T]} l(\phi(\tau, t, x, \hat{u}(\cdot))) + \epsilon.$$

For all $\tau \in [t,T]$

$$\begin{aligned}
&|\phi(\tau,t,x,\hat{u}(\cdot)) - \phi(\tau,t,\hat{x},\hat{u}(\cdot))| \\
&\quad = \left| (x-\hat{x}) + \int_t^\tau [f(\phi(s,t,x,\hat{u}(\cdot)),\hat{u}(s)) - f(\phi(s,t,\hat{x},\hat{u}(\cdot)),\hat{u}(s)]\, ds \right| \\
&\quad \leq |x-\hat{x}| + \int_t^\tau |f(\phi(s,t,x,\hat{u}(\cdot)),\hat{u}(s)) - f(\phi(s,t,\hat{x},\hat{u}(\cdot)),\hat{u}(s)|\, ds \\
&\quad \leq |x-\hat{x}| + C_f \int_t^\tau |\phi(s,t,x,\hat{u}(\cdot)) - \phi(s,t,\hat{x},\hat{u}(\cdot))| ds.
\end{aligned}$$

By the Gronwall-Bellman Lemma (see, for example, [7]) there exists a constant $C_x > 0$ such that $|\phi(\tau,t,x,\hat{u}(\cdot)) - \phi(\tau,t,\hat{x},\hat{u}(\cdot))| \leq C_x|x-\hat{x}|$ for all $\tau \in [t,T]$.

Let $\tau_0 \in [t, T]$ be such that

$$l(\phi(\tau_0, t, x, \hat{u}(\cdot))) = \min_{\tau \in [t,T]} l(\phi(\tau, t, x, \hat{u}(\cdot))).$$

Then,

$$\begin{aligned} V_1(\hat{x}, t) - V_1(x, t) &\leq l(\phi(\tau_0, t, \hat{x}, \hat{u}(\cdot))) - l(\phi(\tau_0, t, x, \hat{u}(\cdot))) + \epsilon \\ &\leq C_l |\phi(\tau_0, t, \hat{x}, \hat{u}(\cdot))) - \phi(\tau_0, t, x, \hat{u}(\cdot))| + \epsilon \\ &\leq C_l C_x |x - \hat{x}| + \epsilon. \end{aligned}$$

The same argument with the roles of x and $\hat{x}$ reversed establishes that

$$V_1(x, t) - V_1(\hat{x}, t) \leq C_l C_x |x - \hat{x}| + \epsilon.$$

Since the argument holds for all $\epsilon > 0$, there exists $C > 0$ such that

$$|V_1(x, t) - V_1(\hat{x}, t)| \leq C|x - \hat{x}|.$$

Finally, consider $x \in \mathbb{R}^n$ and $t, \hat{t} \in [0, T]$. Without loss of generality assume $t < \hat{t}$. By Theorem 5.17

$$V_1(x, t) - V_1(x, \hat{t}) \leq 0.$$

To establish a lower bound on $V_1(x, t) - V_1(x, \hat{t})$, let $\epsilon > 0$ and choose $\hat{u}(\cdot) \in \mathcal{U}_{[\hat{t},T]}$ such that

$$V_1(x, \hat{t}) \leq \min_{\tau \in [\hat{t},T]} l(\phi(\tau, \hat{t}, x, \hat{u}(\cdot))) + \epsilon,$$

Define $u(\cdot) \in \mathcal{U}_{[t,T]}$ by

$$u(\tau) = \begin{cases} \hat{u}(\tau + \hat{t} - t) & \text{if } \tau \in [t, T - (\hat{t} - t)) \\ \hat{u}(T) & \text{if } \tau \in [T - (\hat{t} - t), T] \end{cases}$$

By uniqueness, $\phi(\tau, t, x, u(\cdot)) = \phi(\tau + \hat{t} - t, \hat{t}, x, \hat{u}(\cdot))$ for all $\tau \in [t, T - (\hat{t} - t)]$. By definition,

$$V_1(x, t) \geq \min_{\tau \in [t,T]} l(\phi(\tau, t, x, u(\cdot))).$$

Therefore,

$$V_1(x, \hat{t}) - V_1(x, t) \leq \min_{\tau \in [\hat{t},T]} l(\phi(\tau, \hat{t}, x, \hat{u}(\cdot))) - \min_{\tau \in [t,T]} l(\phi(\tau, t, x, u(\cdot))) + \epsilon.$$

Let $\tau_0 \in [t, T]$ be such that

$$l(\phi(\tau_0, t, x, u(\cdot))) = \min_{\tau \in [t,T]} l(\phi(\tau, t, x, u(\cdot))).$$

If $\tau_0 \in [t, T - (\hat{t} - t)]$, then $l(\phi(\tau_0, t, x, u(\cdot))) = l(\phi(\tau_0 + \hat{t} - t, \hat{t}, x, \hat{u}(\cdot)))$ and therefore

$$V_1(x, \hat{t}) - V(x, t) \leq \epsilon.$$

If $\tau_0 \in [T - (\hat{t} - t), T]$ then

$$\begin{aligned} V_1(x,\hat{t}) - V_1(x,t) &\leq l(\phi(T,\hat{t},x,\hat{u}(\cdot))) - l(\phi(\tau_0,t,x,u(\cdot))) + \epsilon \\ &= l(\phi(T-(\hat{t}-t),t,x,u(\cdot))) - l(\phi(\tau_0,t,x,u(\cdot))) + \epsilon \\ &\leq C_l |\phi(T-(\hat{t}-t),t,x,u(\cdot)) - \phi(\tau_0,t,x,u(\cdot))| + \epsilon \\ &\leq C_l C_f |\tau_0 - (T-(\hat{t}-t))| + \epsilon \\ &\leq C_l C_f |\hat{t} - t| + \epsilon. \end{aligned}$$

The desired bound follows since this inequality holds for all $\epsilon > 0$. ∎

Next, introduce the Hamiltonian $H_1 : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ defined by

$$H_1(p,x) = \min\left\{0, \sup_{u \in U} p^T f(x,u)\right\}. \tag{5.34}$$

The following fact, together with the boundedness and continuity of V_1 established in Lemma 5.18, will be used to establish that the viscosity solution to (5.29) is unique.

Lemma 5.19 *There exists a constant $C > 0$ such that $|H_1(p,x) - H_1(q,x)| < C|p-q|$ and $|H_1(p,x) - H_1(p,y)| < C|p||x-y|$, for all $p, q \in \mathbb{R}^n$ and all $x, y \in \mathbb{R}^n$.*

Proof: Notice that

$$|H_1(p,x) - H_1(q,x)| = \left|\min\left\{0, \sup_{u\in U} p^T f(x,u)\right\} - \min\left\{0, \sup_{u\in U} q^T f(x,u)\right\}\right|.$$

Therefore, to show the first statement of the lemma it suffices to show that

$$\left|\sup_{u\in U} p^T f(x,u) - \sup_{u\in U} q^T f(x,u)\right| < |p-q|,$$

i.e. study the case where $\sup_{u\in U} p^T f(x,u) < 0$ and $\sup_{u\in U} q^T f(x,u) < 0$. A similar comment extends to the second statement of the lemma.

From this point on the proof is the same as for the standard Hamilton-Jacobi equation for optimal control.

$$\begin{aligned} \sup_{u\in U} p^T f(x,u) - \sup_{u\in U} q^T f(x,u) &= \sup_{u\in U} (p-q)^T f(x,u) \\ &\leq |p-q| C_f. \end{aligned}$$

A similar argument with the roles of p and q reversed establishes the first part. Similarly,

$$\begin{aligned} \sup_{u\in U} p^T f(x,u) - \sup_{u\in U} p^T f(y,u) &\leq \sup_{u\in U} p^T \left(f(x,u) - f(y,u)\right) \\ &\leq |p| \sup_{u\in U} |f(x,u) - f(y,u)| \\ &\leq |p| C_f |x-y|. \end{aligned}$$

∎

Finally, the following fact (see, for example, [116], page 546) saves us the trouble of checking the viscosity solution conditions at the initial time.

Lemma 5.20 *Assume that V_1 satisfies the viscosity conditions for equation (5.29) over $\mathbb{R}^n \times (0,T)$. Then for all $W : \mathbb{R}^n \times [0,T] \to \mathbb{R}$ such that $V_1 - W$ attains a local maximum (minimum) at $(x_0,t_0) \in \mathbb{R}^n \times [0,T)$*

$$\frac{\partial W}{\partial t}(x_0,t_0) + H_1\left(\frac{\partial W}{\partial x}(x_0,t_0), x\right) \geq 0 \ \ (\leq 0).$$

Theorem 5.21 (SupMin Value Function Characterization) *V_1 is the unique bounded and uniformly continuous viscosity solution of the terminal value problem*

$$\frac{\partial V}{\partial t}(x,t) + H_1\left(\frac{\partial V}{\partial x}(x,t), x\right) = 0$$

over $(x,t) \in \mathbb{R}^n \times [0,T]$ with boundary condition $V(x,T) = l(x)$.

Proof: Recall that $V_1(x,T) = l(x)$. Therefore, under Lemma 5.20, it suffices to show that

1. For all $(x_0,t_0) \in \mathbb{R}^n \times (0,T)$ and for all smooth $W : \mathbb{R}^n \times (0,T) \to \mathbb{R}$, if $V_1 - W$ attains a local maximum at (x_0,t_0), then
$$\frac{\partial W}{\partial t}(x_0,t_0) + \min\left\{0, \sup_{u\in U} \frac{\partial W}{\partial x}(x_0,t_0) f(x_0,u)\right\} \geq 0.$$
2. For all $(x_0,t_0) \in \mathbb{R}^n \times (0,T)$ and for all smooth $W : \mathbb{R}^n \times (0,T) \to \mathbb{R}$, if $V_1 - W$ attains a local minimum at (x_0,t_0), then
$$\frac{\partial W}{\partial t}(x_0,t_0) + \min\left\{0, \sup_{u\in U} \frac{\partial W}{\partial x}(x_0,t_0) f(x_0,u)\right\} \leq 0.$$

Uniqueness then follows by Lemmas 5.18 and 5.19 and a standard uniqueness result for viscosity solutions [116].

Part 1: Consider an arbitrary $(x_0,t_0) \in \mathbb{R}^n \times (0,T)$ and a smooth $W : \mathbb{R}^n \times (0,T) \to \mathbb{R}$ such that $V_1 - W$ attains a local maximum at (x_0,t_0). Then, there exists $\delta_1 > 0$ such that for all $(x,t) \in \mathbb{R}^n \times (0,T)$ with $|x - x_0|^2 + (t-t_0)^2 < \delta_1$

$$(V_1 - W)(x_0,t_0) \geq (V_1 - W)(x,t). \tag{5.35}$$

We would like to show that

$$\frac{\partial W}{\partial t}(x_0,t_0) + \min\left\{0, \sup_{u\in U} \frac{\partial W}{\partial x}(x_0,t_0) f(x_0,u)\right\} \geq 0.$$

Assume, for the sake of contradiction, that this is not the case. Then, for some $\theta > 0$,

$$\frac{\partial W}{\partial t}(x_0,t_0) + \min\left\{0, \sup_{u\in U} \frac{\partial W}{\partial x}(x_0,t_0) f(x_0,u)\right\} < -2\theta < 0. \tag{5.36}$$

We distinguish two cases.
Case 1.1: $\sup_{u\in U} \frac{\partial W}{\partial x}(x_0,t_0)f(x_0,u) < 0$. Then

$$\frac{\partial W}{\partial t}(x_0,t_0) + \sup_{u\in U} \frac{\partial W}{\partial x}(x_0,t_0)f(x_0,u) < -2\theta.$$

Moreover, there exists an $\epsilon > 0$ such that for all $u \in U$

$$\frac{\partial W}{\partial x}(x_0,t_0)f(x_0,u) < -\epsilon$$

Therefore, since W is smooth, there exists $\delta_2 \in (0,\delta_1)$ such that for all $(x,t) \in \mathbb{R}^n \times (0,T)$ with $|x-x_0|^2 + (t-t_0)^2 < \delta_2$ and all $u \in U$,

$$\frac{\partial W}{\partial t}(x,t)+\min\left\{0, \frac{\partial W}{\partial x}(x,t)f(x,u)\right\} = \frac{\partial W}{\partial t}(x,t)+\frac{\partial W}{\partial x}(x,t)f(x,u) < -\theta < 0.$$

Consider an arbitrary $u(\cdot) \in \mathcal{U}_{[t_0,T]}$. By continuity of the solution with respect to time, there exists $\delta_3 > 0$ such that for all $t \in [t_0, t_0+\delta_3]$,

$$|\phi(t,t_0,x_0,u(\cdot)) - x_0|^2 + (t-t_0)^2 < \delta_2. \tag{5.37}$$

Therefore, by equation (5.35),

$$\begin{aligned}
V_1(\phi(t_0+\delta_3,&t_0,x_0,u(\cdot)),t_0+\delta_3) - V_1(x_0,t_0)\\
&\leq W(\phi(t_0+\delta_3,t_0,x_0,u(\cdot)),t_0+\delta_3) - W(x_0,t_0)\\
&= \int_{t_0}^{t_0+\delta_3} \frac{d}{dt} W(\phi(t,t_0,x_0,u(\cdot)),t)dt\\
&= \int_{t_0}^{t_0+\delta_3} \frac{\partial W}{\partial t}(\phi(t,t_0,x_0,u(\cdot)),t)dt\\
&\quad + \int_{t_0}^{t_0+\delta_3} \frac{\partial W}{\partial x}(\phi(t,t_0,x_0,u(\cdot)),t)f(\phi(t,t_0,x_0,u(\cdot)),u(t))dt\\
&< -\theta\delta_3.
\end{aligned}$$

By Theorem 5.17

$$V_1(x_0,t_0) = \sup_{u(\cdot)\in\mathcal{U}_{[t_0,t_0+\delta_3]}} \left[\min\left\{\min_{t\in[t_0,t_0+\delta_3]} l(\phi(t,t_0,x_0,u(\cdot))), V_1(\phi(t_0+\delta_3,t_0,x_0,u(\cdot)),t_0+\delta_3)\right\}\right].$$

Therefore, there exists $u(\cdot) \in \mathcal{U}_{[t_0,t_0+\delta_3]}$ such that

$$\begin{aligned}
V_1(x_0,t_0) &\leq \min\left\{\min_{t\in[t_0,t_0+\delta_3]} l(\phi(t,t_0,x_0,u(\cdot))), V_1(\phi(t_0+\delta_3,t_0,x_0,u(\cdot)),t_0+\delta_3)\right\} + \frac{\theta\delta_3}{2}\\
&\leq V_1(\phi(t_0+\delta_3,t_0,x_0,u(\cdot)),t_0+\delta_3) + \frac{\theta\delta_3}{2}
\end{aligned}$$

which is a contradiction.
Case 1.2: $\sup_{u\in U} \frac{\partial W}{\partial x}(x_0,t_0)f(x_0,u) \geq 0$. By equation (5.36),

$$\frac{\partial W}{\partial t}(x_0,t_0) < -2\theta < 0.$$

Since W is smooth, there exists $\delta_2 \in (0, \delta_1)$ such that for all $(x,t) \in \mathbb{R}^n \times (0,T)$ with $|x - x_0|^2 + (t - t_0)^2 < \delta_2$,

$$\frac{\partial W}{\partial t}(x,t) < -\theta < 0.$$

By equation (5.35),

$$\begin{aligned} V_1(x_0, t_0 + \delta_2) - V_1(x_0, t_0) &\leq W(x_0, t_0 + \delta_2) - W(x_0, t_0) \\ &= \int_{t_0}^{t_0+\delta_2} \frac{\partial W}{\partial t}(x_0, t) dt \\ &< -\theta\delta_2. \end{aligned}$$

This contradicts Theorem 5.17.

Part 2: Consider an arbitrary $(x_0, t_0) \in \mathbb{R}^n \times (0,T)$ and a smooth $W : \mathbb{R}^n \times (0,T) \to \mathbb{R}$ such that $V_1 - W$ attains a local minimum at (x_0, t_0). Then, there exists $\delta_1 > 0$ such that for all $(x,t) \in \mathbb{R}^n \times (0,T)$ with $|x - x_0|^2 + (t - t_0)^2 < \delta_1$

$$(V_1 - W)(x_0, t_0) \leq (V_1 - W)(x,t). \tag{5.38}$$

We would like to show that

$$\frac{\partial W}{\partial t}(x_0, t_0) + \min\left\{0, \sup_{u \in U} \frac{\partial W}{\partial x}(x_0, t_0) f(x_0, u)\right\} \leq 0.$$

Assume, for the sake of contradiction, that this is not the case. Then, for some $\theta > 0$,

$$\frac{\partial W}{\partial t}(x_0, t_0) + \min\left\{0, \sup_{u \in U} \frac{\partial W}{\partial x}(x_0, t_0) f(x_0, u)\right\} > 2\theta > 0.$$

Therefore, there exists $\hat{u} \in U$ such that

$$\frac{\partial W}{\partial t}(x_0, t_0) + \min\left\{0, \frac{\partial W}{\partial x}(x_0, t_0) f(x_0, \hat{u})\right\} > 2\theta > 0.$$

By smoothness of W, there exists $\delta_2 \in (0, \delta_1)$ such that for all $(x,t) \in \mathbb{R}^n \times (0,T)$ with $|x - x_0|^2 + (t - t_0)^2 < \delta_2$

$$\frac{\partial W}{\partial t}(x,t) + \min\left\{0, \frac{\partial W}{\partial x}(x,t) f(x, \hat{u})\right\} > \theta > 0. \tag{5.39}$$

By continuity of the solution with respect to t, there exists $\delta_3 > 0$ such that for all $t \in [t_0, t_0 + \delta_3]$,

$$|\phi(t, t_0, x_0, \hat{u}) - x_0|^2 + (t - t_0)^2 < \delta_2. \tag{5.40}$$

Therefore, by equation (5.38), for all $t \in [t_0, t_0 + \delta_3]$,

$$\begin{aligned}
&V_1(\phi(t, t_0, x_0, \hat{u}), t) - V_1(x_0, t_0) \\
&\quad \geq W(\phi(t, t_0, x_0, \hat{u}), t) - W(x_0, t_0) \\
&\quad = \int_{t_0}^{t} \frac{\partial W}{\partial t}(\phi(\tau, t_0, x_0, \hat{u}), \tau) d\tau \\
&\quad + \int_{t_0}^{t} \frac{\partial W}{\partial x}(\phi(\tau, t_0, x_0, \hat{u}), \tau) f(\phi(\tau, t_0, x_0, \hat{u}), \hat{u}) d\tau \\
&\quad \geq \int_{t_0}^{t} \frac{\partial W}{\partial t}(\phi(\tau, t_0, x_0, \hat{u}), \tau) d\tau \\
&\quad + \int_{t_0}^{t} \min \left\{ 0, \frac{\partial W}{\partial x}(\phi(\tau, t_0, x_0, \hat{u}), \tau) f(\phi(\tau, t_0, x_0, \hat{u}), \hat{u}) \right\} d\tau \\
&\quad > \theta(t - t_0).
\end{aligned}$$

In particular,

$$V_1(\phi(t_0 + \delta_3, t_0, x_0, \hat{u}), t_0 + \delta_3) - V_1(x_0, t_0) > \theta \delta_3. \tag{5.41}$$

Recall that, by Theorem 5.17,

$$V_1(x_0, t_0) \geq \min \left\{ \min_{t \in [t_0, t_0 + \delta_3]} l(\phi(t, t_0, x_0, \hat{u})), V_1(\phi(t_0 + \delta_3, t_0, x_0, \hat{u}), t_0 + \delta_3) \right\}.$$

Case 2.1: $V_1(\phi(t_0 + \delta_3, t_0, x_0, \hat{u}), t_0 + \delta_3) \leq \min_{t \in [t_0, t_0 + \delta_3]} l(\phi(t, t_0, x_0, \hat{u}))$. Then $V_1(x_0, t_0) \geq V_1(\phi(t_0 + \delta_3, t_0, x_0, \hat{u}), t_0 + \delta_3)$ and therefore

$$V_1(\phi(t_0 + \delta_3, t_0, x_0, \hat{u}), t_0 + \delta_3) - V_1(x_0, t_0) \leq 0.$$

This contradicts equation (5.41).
Case 2.2: $V_1(\phi(t_0 + \delta_3, t_0, x_0, \hat{u}), t_0 + \delta_3) > \min_{t \in [t_0, t_0 + \delta_3]} l(\phi(t, t_0, x_0, \hat{u}))$. Then

$$V_1(x_0, t_0) \geq \min_{t \in [t_0, t_0 + \delta_3]} l(\phi(t, t_0, x_0, \hat{u})). \tag{5.42}$$

Recall that for all $t \in [t_0, t_0 + \delta_3]$ with $t > t_0$

$$V_1(\phi(t, t_0, x_0, \hat{u}), t) - V_1(x_0, t_0) \geq \theta(t - t_0) > 0 \tag{5.43}$$

(in fact, $V_1(\phi(\cdot, t_0, x_0, \hat{u}), \cdot)$ is monotone increasing as a function of $t \in [t_0, t_0 + \delta_3]$). By Theorem 5.17, for all $t \in [t_0, t_0 + \delta_3]$

$$l(\phi(t, t_0, x_0, \hat{u})) \geq V_1(\phi(t, t_0, x_0, \hat{u}), t) \geq V_1(x_0, t_0).$$

Hence,

$$\min_{t \in [t_0, t_0 + \delta_3]} l(\phi(t, t_0, x_0, \hat{u})) \geq V_1(x_0, t_0)$$

and, by equation (5.42),

$$V_1(x_0, t_0) = \min_{t \in [t_0, t_0 + \delta_3]} l(\phi(t, t_0, x_0, \hat{u})).$$

The minimum occurs at $t = t_0$ and the minimizer is unique. If this were not the case, then there would exist $\tau \in [t_0, t_0 + \delta_3]$ with $\tau > t_0$ such that

$$\min_{t \in [t_0, t_0+\delta_3]} l(\phi(t, t_0, x_0, \hat{u})) = l(\phi(\tau, t_0, x_0, \hat{u})).$$

Then

$$V_1(x_0, t_0) = l(\phi(\tau, t_0, x_0, \hat{u})) \geq V_1(\phi(\tau, t_0, x_0, \hat{u}), \tau),$$

which would contradict equation (5.43). Therefore,

$$V_1(x_0, t_0) = \min_{t \in [t_0, t_0+\delta_3]} l(\phi(t, t_0, x_0, \hat{u})) = l(x_0).$$

By Theorem 5.17,

$$V_1(x_0, t_0) \leq V_1(x_0, t_0 + \delta_3) \leq l(x_0).$$

Therefore, $V_1(x_0, t_0+\delta_3) = l(x_0) = V_1(x_0, t_0)$. However, by equation (5.38),

$$\begin{aligned} V_1(x_0, t_0 + \delta_3) - V_1(x_0, t_0) &\geq W(x_0, t_0 + \delta_3) - W(x_0, t_0) \\ &= \int_{t_0}^{t_0+\delta_3} \frac{\partial W}{\partial t}(x_0, t) dt \\ &\geq \int_{t_0}^{t_0+\delta_3} \left(\theta - \min \left\{ 0, \frac{\partial W}{\partial x}(x_0, t) f(x_0, \hat{u}) \right\} \right) dt \\ &\geq \theta \delta_3. \end{aligned}$$

This contradiction completes the proof. ∎

Finally, we note the independence of $\mathrm{Viab}(t, K)$ on the function l used to characterize the set K.

Exercise 5.7 *Let $l : \mathbb{R}^n \to \mathbb{R}$ and $\hat{l} : \mathbb{R}^n \to \mathbb{R}$ be two Lipschitz continuous, bounded functions such that $\{x \in \mathbb{R}^n \mid l(x) > 0\} = \{x \in \mathbb{R}^n \mid \hat{l}(x) > 0\}$. Let $V_1 : \mathbb{R}^n \times [0, T] \to \mathbb{R}$ and $\hat{V}_1 : \mathbb{R}^n \times [0, T] \to \mathbb{R}$ be the viscosity solutions of (5.29) with boundary conditions $V_1(x, T) = l(x)$ and $\hat{V}_1(x, T) = \hat{l}(x)$ respectively. Then $\{x \in \mathbb{R}^n \mid V_1(x, t) > 0\} = \{x \in \mathbb{R}^n \mid \hat{V}_1(x, t) > 0\}$ for all $t \in [0, T]$.*

5.3.4 Solution of the InfMin *Problem*

The procedure for establishing the properties of the value function V_2 is effectively the same. We start by showing that V_2 satisfies an appropriate version of the optimality principle.

Theorem 5.22 (InfMin Optimality Conditions) *For all $(x, t) \in \mathbb{R}^n \times [0, T]$ and all $h \in [0, T - t]$ $V_2(x, t) \leq V_2(x, t + h)$ and $V_2(x, T) = l(x)$. Moreover,*

$$V_2(x, t) = \inf_{u(\cdot) \in \mathcal{U}_{[t,t+h]}} \left[\min \left\{ \min_{\tau \in [t, t+h]} l(\phi(\tau, t, x, u(\cdot))), V_2(\phi(t + h, t, x, u(\cdot)), t + h) \right\} \right].$$

The implications of the theorem are similar to those of Theorem 5.17. The first part states that the "value" of a given state can only decrease as the "time to go" increases. Just as with V_1, it is this property that forces the level sets of V_2 to change monotonically with t (see Figure 5.3) and results in the extra minimization in equation (5.30). The second part is again a variant of the principle of optimality.

Next, we show that under Assumption 5.14 the value function V_2 is well behaved.

Lemma 5.23 *There exists a constant $C > 0$ such that $|V_2(x,t)| \leq C$ and $|V_2(x,t) - V_2(\hat{x},\hat{t})| \leq C(|x-\hat{x}| + (t-\hat{t}))$, for all $(x,t),(\hat{x},\hat{t}) \in \mathbb{R}^n \times [0,T]$.*

We introduce the Hamiltonian

$$H_2(p,x) = \min\left\{0, \inf_{u\in U} p^T f(x,u)\right\}, \tag{5.44}$$

and show that

Lemma 5.24 *There exists a constant $C > 0$ such that $|H_2(p,x) - H_2(q,x)| < C|p-q|$ and $|H_2(p,x) - H_2(q,y)| < C|p||x-y|$, for all $p,q \in \mathbb{R}^n$ and all $x,y \in \mathbb{R}^n$.*

Putting all the above facts together leads to the proof of the characterization theorem for the value function of the InfMin optimal control problem.

Theorem 5.25 (InfMin Value Function Characterization) *V_2 is the unique bounded and uniformly continuous viscosity solution to the terminal value problem*

$$\frac{\partial V}{\partial t}(x,t) + H_2\left(\frac{\partial V}{\partial x}(x,t), x\right) = 0$$

over $(x,t) \in \mathbb{R}^n \times [0,T]$ with $V(x,t) = l(x)$.

5.4 Viability Perspective (John)

We summarize some concepts from non-smooth analysis and viability theory that will be needed to introduce the results. Recall that a function $g(\cdot) : \mathbb{R}^n \to \mathbb{R}$ is called *lower semi-continuous* if for all $x \in \mathbb{R}^n$ and all $\epsilon > 0$, there exists $\delta > 0$ such that for all $x' \in \mathbb{R}^n$ with $|x - x'| < \delta$, $g(x) \leq g(x') + \epsilon$. The *epigraph* of g is the set

$$\mathrm{Epi}(g) = \{(x,y) \in \mathbb{R}^{n+1} \mid y \geq g(x)\}.$$

A standard result in non-smooth analysis shows that g is lower semi-continuous if and only if $\mathrm{Epi}(g)$ is a closed subset of $\mathbb{R}^{n+1}$.

For a closed set, $K \subseteq \mathbb{R}^n$, and a point $x \in K$, we use $T_K(x)$ to denote the *contingent cone* (or *Bouligand tangent cone*) to K at x, i.e. the set of

$v \in \mathbb{R}^n$ for which there exists a sequence of real numbers $h_n > 0$ converging to 0 and a sequence of $v_n \in \mathbb{R}^n$ converging to v satisfying

$$\forall\, n \geq 0,\;\; x + h_n v_n \;\in\; K.$$

For an arbitrary set, K, let 2^K denote the power set of K, i.e. the set of all subsets of K. A set valued map $F(\cdot) : \mathbb{R}^n \to 2^{\mathbb{R}^n}$ is called *upper semi-continuous* if for all $x \in \mathbb{R}^n$ and all $\epsilon > 0$ there exists $\delta > 0$ such that for all x' with $|x - x'| < \delta$, $F(x') \subseteq F(x) + \epsilon B$. (As usual B denotes a closed ball in $\mathbb{R}^n$, of radius 1 centered at the origin.) We say that a set valued map F is *Marchaud* if

1. F is upper semi-continuous;
2. For all $x \in \mathbb{R}^n$, $F(x)$ is convex, compact and nonempty;
3. The growth of F is linear, that is there exists $c > 0$ such that for all $x \in \mathbb{R}^n$

$$\sup\{|v| \mid v \in F(x)\} \;\leq\; c(|x| + 1).$$

We say F is *Lipschitz* if there exists a constant $\lambda > 0$ (known as the *Lipschitz constant*) such that for all $x, x' \in \mathbb{R}^n$

$$F(x) \subseteq F(x') + \lambda \|x - x'\| B.$$

Viability theory deals with the properties of the set of solutions of differential inclusions. Notice that the control system of equation (A.2) can be thought of as a *differential inclusion* by setting

$$F(x) = \{f(x, u) \mid u \in U\}.$$

For a set valued map $F(\cdot) : \mathbb{R}^n \to 2^{\mathbb{R}^n}$, a solution of the differential inclusion

$$\dot{x} \in F(x) \tag{5.45}$$

over an interval $[0, T]$ starting at $x_0 \in \mathbb{R}^n$ is an absolutely continuous function $x(\cdot) : [0, T] \to \mathbb{R}^n$ such that $x(0) = x_0$ and $\dot{x}(t) \in F(x(t))$ for almost every $t \in [0, T]$.

Definition 5.26 (Viable and Invariant Sets) *A solution $x(\cdot) : [0, T] \to \mathbb{R}^n$ of* (C.1) *is called viable in a set $K \subseteq \mathbb{R}^n$, if $x(t) \in K$ for all $t \in [0, T]$. A set $K \subseteq \mathbb{R}^n$ is called locally viable under the differential inclusion* (C.1) *if for all $x_0 \in \mathbb{R}^n$ there exists $T > 0$ and a solution $x(\cdot) : [0, T] \to \mathbb{R}^n$ of* (C.1) *starting at x_0 that is viable in K. K is called viable under* (C.1) *if the above holds for all $T > 0$. K is called invariant under* (C.1) *if for all $x_0 \in K$ all solutions of* (C.1) *starting at x_0 are viable in K.*

The following characterizations of viable and invariant sets can be found in [114].

Theorem 5.27 (Viability Conditions) *Assume F is Marchaud. A closed set $K \subseteq \mathbb{R}^n$ is viable under the differential inclusion $\dot{x} \in F(x)$ if and only if for all $x \in K$, $T_K(x) \cap F(x) \neq \emptyset$.*

Theorem 5.28 (Invariance Conditions) *Assume F is Marchaud and Lipschitz. A closed set $K \subseteq \mathbb{R}^n$ is invariant under the differential inclusion $\dot{x} \in F(x)$ if and only if for all $x \in K$, $F(x) \subseteq T_K(x)$.*

If a set is not viable/invariant, one would often like to determine the largest subset of it that is.

Definition 5.29 (Viability and Invariance Kernels) *Consider a set $K \subseteq \mathbb{R}^n$. The viability kernel, $Viab_F(K)$, of K under the differential inclusion* (C.1) *is the set of states $x_0 \in K$ for which there exists an infinite solution $x(\cdot) : [0, \infty) \to \mathbb{R}^n$ of* (C.1) *starting at x_0 that is viable in K. The invariance kernel, $Inv_F(K)$, of K is the set of states $x_0 \in K$ for which all solutions of* (C.1) *starting at x_0 are viable in K.*

The following characterizations of the viability and invariance kernels can be found in [114].

Theorem 5.30 (Viability Kernel Characterization) *Assume F is Marchaud and K is closed. $Viab_F(K)$ is the largest closed subset of K (possibly the empty set) that satisfies the conditions of Theorem 5.27.*

Theorem 5.31 (Invariance Kernel Characterization) *Assume F is Marchaud and Lipschitz and K is closed. $Inv_F(K)$ is the largest closed subset of K (possibly the empty set) that satisfies the conditions of Theorem 5.28.*

5.5 Pursuit evasion differential games (Claire)

5.6 Bibliography and Further Reading

The treatment of the optimal control interpretation of reachability concepts comes from [117] and the differential games results are from [118].

The control SupMin problem is a special case of the optimal control problem treated by [119, 120], where l is also allowed to depend on t and u. In [119] the value function of the problem is shown to satisfy a set of discontinuous, quasi-variational inequalities. Though this approach is conceptually appealing, the discontinuity and the implicit dependence of the Hamiltonian on the value function severely limit its usefulness from the numerical computation point of view (as the authors of [119] point out). The authors of [120] simplify this characterization to the following continuous variational inequality

$$\sup_{u \in U} \min \left\{ l(x) - V(x,t), \frac{\partial V}{\partial t}(x,t) + \frac{\partial V}{\partial x}(x,t) f(x,u) \right\} = 0. \tag{5.46}$$

The main advantage of equation (5.46) is that the Hamiltonian is continuous. In [120] specialized numerical schemes were developed to exploit this fact and approximate the solutions to the variational inequality (5.46).

The InfMin problem was also treated in [120] and in [121] from a viability point of view. In [120] a continuous variational inequality

$$\min\left\{l(x) - V(x,t), \frac{\partial V}{\partial t}(x,t) + \inf_{u\in U} \frac{\partial V}{\partial x}(x,t) f(x,u)\right\} = 0. \tag{5.47}$$

was also proposed to address the InfMin problem.

Other methods in the optimal control literature that can also be adapted to characterize the sets $\mathrm{Viab}(t,K)$ and $\mathrm{Inv}(t,K)$. For example, one can treat the problem as maximizing or minimizing the "exit time" from the set K. It can be shown that this involves solving the terminal value problem (5.53) over only the set K, with rather complicated boundary conditions [122, 123]. A different method for characterizing the set Inv using a standard optimal control approach is discussed in Problems 5.6 and 5.7. Yet another approach is to solve the modified terminal value problem

$$-\frac{\partial V}{\partial t}(x,t) = \begin{cases} \sup_{u\in U} \frac{\partial V}{\partial x}(x,t) f(x,u) & \text{if } x \in K \\ \min\left\{0, \sup_{u\in U} \frac{\partial V}{\partial x}(x,t) f(x,u)\right\} & \text{if } x \in K^c. \end{cases} \tag{5.48}$$

This approach was proposed in [105] and will be discussed further in Chapter **??**.

The authors of [124] also consider a Hamiltonian similar to that of equation (5.48) for differential games. Related work on differential games includes [125] (extending the results of [119]) and [118] (based on the classical results of [126]).

The infinite horizon variant of the SupMin optimal control problem is studied in [127] for the case of classical controls and in [128] for relaxed controls. The infinite horizon InfMin problem is studied in [120], again for relaxed controls. In all these references characterizations of the value function as viscosity solutions to variational inequalities are derived. An alternative characterization of both infinite horizon problems using viability theory is discussed below in Problems 5.2–5.5. This characterization is motivated by a connection between viability theory and the value function of optimal control problems established in [129, 130].

5.7 Problems

Problem 5.1 *Assume F is Marchaud. Then for all $x \in \mathbb{R}^n$ there exists $K > 0$ such that all solutions to $\dot{x} \in F(x)$ starting at x satisfy $|x(t) - x| \leq kt$ for all t sufficiently small.*

Problem 5.2 *Consider the following infinite horizon variants of the* SupMin *and* InfMin *optimal control problems:*

$$V_1^*(x) = \inf_{u(\cdot)\in\mathcal{U}_{[0,\infty)}} \sup_{t\in[0,\infty)} l(\phi(t,0,x,u(\cdot))) \tag{5.49}$$

$$V_2^*(x) = \sup_{u(\cdot)\in\mathcal{U}_{[0,\infty)}} \sup_{t\in[0,\infty)} l(\phi(t,0,x,u(\cdot))). \tag{5.50}$$

Assume that $l(\cdot)$ is Lipschitz and bounded from above, $f(\cdot,\cdot)$ is Lipschitz and bounded and $F(\cdot)$ is Marchaud. Show that for all $x \in \mathbb{R}^n$ and all $t \geq 0$, the following properties hold:

1. $l(x) \leq V_1^*(x) \leq V_2^*(x)$.
2. $V_1^*(x) = \inf_{u(\cdot)\in\mathcal{U}_{[0,t]}} \max\left\{\sup_{\tau\in[0,t]} l(\phi(\tau,0,x,u(\cdot))), V_1^*(\phi(t,0,x,u(\cdot)))\right\}$.
3. $V_2^*(x) = \sup_{u(\cdot)\in\mathcal{U}_{[0,t]}} \max\left\{\sup_{\tau\in[0,t]} l(\phi(\tau,0,x,u(\cdot))), V_2^*(\phi(t,0,x,u(\cdot)))\right\}$.

Problem 5.3 *Consider again the value functions V_1^* and V_2^* of Problem 5.2. Assume an auxiliary state variable, $y \in \mathbb{R}$, is appended to the state and consider the extended dynamics*

$$(\dot{x},\dot{y}) \in \hat{F}(x,y) \text{ with } \begin{cases} \dot{x} \in F(x) \\ \dot{y} = 0. \end{cases} \tag{5.51}$$

Show that Epi(V_1^*) is the viability kernel of Epi(l) under the differential inclusion (5.51)*. You may assume that Epi(V_1^*) is a closed set.*

Problem 5.4 *Under the assumptions of 5.2, show that Epi(V_2^*) is the invariance kernel of Epi(l) under the differential inclusion* (5.51)*. You may again assume that Epi(V_2^*) is a closed set.*

Problem 5.5 *In the setting of Problem 5.2, assume that a closed set $K \subseteq \mathbb{R}^n$ is such that $K = \{x \in \mathbb{R}^n \mid l(x) \leq 0\}$. Show that $Viab_F(K) = \{x \in \mathbb{R}^n \mid V_1^*(x) \leq 0\}$ and $Inv_F(K) = \{x \in \mathbb{R}^n \mid V_2^*(x) \leq 0\}$.*

Problem 5.6 *Let $K = \{x \in \mathbb{R}^n \mid l(x) \geq 0\}$ and consider the value function*

$$V_3(x,t) = \inf_{u(\cdot)\in\mathcal{U}_{[t,T]}} l(\phi(T,t,x,u(\cdot))).$$

Show that $Inv(t,L) = \bigcap_{\tau\in[t,T]}\{x \in \mathbb{R}^n \mid V_3(x,\tau) \geq 0\}$. Hence show that $V_2(x,t) = \min_{\tau\in[t,T]} V_3(x,\tau)$ for all $(x,t) \in \mathbb{R}^n \times [0,T]$,

Problem 5.7 *In the setting of problem* **??***, a standard optimal control argument shows that V_3 is the unique viscosity solution to the terminal value problem*

$$\frac{\partial V_3}{\partial t}(x,t) + \inf_{u\in U} \frac{\partial V_3}{\partial x}(x,t) f(x,u) = 0 \tag{5.52}$$

with $V_3(x,T) = l(x)$ *over* $(x,t) \in \mathbb{R}^n \times [0,T]$. *Show that the function* $\min_{\tau\in[t,T]} V_3(x,\tau)$ *is the unique bounded, uniformly continuous viscosity solution to the terminal value problem* (5.30). *Compute and plot the function* V_3 *for the example of Section 5.3.2.*

Problem 5.8 *Motivated by Problems 5.6 and 5.7 one might expect to be able to compute* $Viab(t,K)$ *by taking intersections of the level sets of the value function*

$$V_4(x,t) = \sup_{u(\cdot)\in\mathcal{U}_{[t,T]}} l(\phi(T,t,x,u(\cdot))).$$

Using standard optimal control results, one can show that V_4 *is a viscosity solution of*

$$\frac{\partial V_4}{\partial t}(x,t) + \sup_{u\in U} \frac{\partial V_4}{\partial x}(x,t) f(x,u) = 0 \tag{5.53}$$

with $V_4(x,T) = l(x)$ *over* $(x,t) \in \mathbb{R}^n \times [0,T]$. *Compute and plot the value function* V_4 *for the example of Section 5.3.2*[1]. *Show that* $Viab(t,K) \neq \bigcap_{\tau\in[t,T]}\{x \in \mathbb{R}^n \mid V_4(x,\tau) \geq 0\}$. *What goes wrong when we try to extend the proof in problem 5.6 to the function* V_4*?*

Problem 5.9 *Prove Theorem 5.22.*

Problem 5.10 *Prove Lemma 5.23.*

Problem 5.11 *Prove Theorem 5.25.*

[1]Even though one guess what the solution is like, it turns out that it is a viscosity solution, so more work is needed to prove that our guess is indeed right.

Chapter 6

Controller Synthesis for Hybrid Systems

This chapter is the counterpart of Chapter 5 developing the tools for the design of controllers for hybrid systems. Recall as in that chapter that a control problem involves:

1. A plant, to be controlled
2. A specification
3. A controller

and that *controller synthesis* involves coming up with a methodology for designing controllers that meet the specification. The discussion in this chapter is based on [110] and [36]. Recall from Chapter 3 that the *plant* is modeled by an open hybrid automaton, $H = (Q, X, V, \mathrm{I}nit, f, I, E, G, R, \phi)$, where

- Q is a finite collection of discrete state variables;
- X is a finite collection of continuous state variables;
- V is a finite collection of input variables. We assume $V = V_D \cup V_C$, where V_D contains discrete and V_C contains continuous variables.
- $\mathrm{I}nit \subseteq \mathbf{Q} \times \mathbf{X}$ is a set of initial states;
- $f : \mathbf{Q} \times \mathbf{X} \times \mathbf{V} \to \mathbb{R}^n$ is an input dependent vector field;
- $I : \mathbf{Q} \to 2^{\mathbf{X} \times \mathbf{V}}$ assigns to each $q \in \mathbf{Q}$ an input dependent invariant set;
- $E \subset \mathbf{Q} \times \mathbf{Q}$ is a collection of discrete transitions;

- $G : E \to 2^{\mathbf{X}\times\mathbf{V}}$ assigns to each $e = (q, q') \in E$ a guard;
- $R : E \times \mathbf{X} \times \mathbf{V} \to 2^{\mathbf{X}}$ assigns to each $e = (q, q') \in E$, $x \in \mathbf{X}$ and $v \in \mathbf{V}$ a reset relation; and,
- $\phi : \mathbf{Q} \times \mathbf{X} \to 2^{\mathbf{V}}$ assigns to each state a set of admissible inputs.

To avoid technical difficulties we introduce the additional assumption that $\phi(q, x) \neq \emptyset$ and f is Lipschitz in x and continuous in v.

In this chapter, the control objective or specification is assumed to be encoded by means of a safety property $(Q \cup X, \Box F)$ (always F) with $F \subseteq \mathbf{Q}\times\mathbf{X}$. Finally, we will assume that the input variables of H are partitioned into *controls*, U and *disturbances*, D:

$$V = U \cup D$$

The disturbances represent uncontrolled inputs such as noise, unmodeled dynamics, the actions of other subsystems (in a distributed system), etc.

6.1 The Controller Structure

A controller can be defined as a map from state executions to sets of allowable inputs:

$$C : \mathcal{X}^* \to 2^{\mathbf{U}}$$

The interpretation is that, given a finite execution, the controller restricts the valuations of the control input variables that are allowed at the final state. With this in mind we can define the set of *closed loop causal executions* as:

$$\mathcal{H}_C = \{(\tau, q, x, (u, d)) \in \mathcal{H} | \forall t \in \tau, u(t) \in C((\tau, q, x) \downarrow_t)\}$$

Clearly, $\mathcal{H}_C \subseteq \mathcal{H}$. We say C satisfies property $(Q \cup X, \Box F)$ if:

$$\Box F(\chi) = \text{True for all } \chi \in \mathcal{H}_C$$

To prevent technical problems, assume that for all $(\{[\tau_i, \tau_i']\}_{i=0}^N, q, x, v) \in \mathcal{H}$, $\emptyset \neq C(x) \subseteq \phi(q(\tau_N'), x(\tau_N'))|_U$. This ensures that the controller will not attempt to "cheat" by stalling the execution. This is not enough however. The controller may still be able to cheat by:

1. Blocking the execution at a state (q, x) by applying $u \in \phi(q, x)|_U$ such that for all $d \in \phi(q, x)|_D$ $(x, (u, d)) \notin I(q)$ and for all $e \in E$ either $(x, (u, d)) \notin G(e)$ or $R(e, x, (u, d)) = \emptyset$.
2. Forcing the execution to be Zeno (take infinite number of transitions in a finite amount of time). Recall the water tanks example.

Both of these caveats have to do with modeling over-abstraction: the model of the plant should be such that the controller is not able to cheat in this

way. The first loophole can be eliminated by fairly simple assumptions on the plant, similar to the non-blocking assumptions for autonomous hybrid automata. The second loophole is more difficult to deal with. Typically one assumes that all loops among discrete states require a non zero amount of time. This assumption may be somewhat restrictive and is difficult to enforce by manipulating the primitives of the model. Here we will overlook these technical problems.

6.1.1 Controller Properties

Given a plant hybrid automaton and a property our goal is to find a controller that satisfies the specification.

Memoryless Controllers

A controller, C, is called *memoryless* (or *pure feedback*) if for all $\chi, \chi' \in \mathcal{H}^*$ ending at the same state, $C(\chi) = C(\chi')$. A memoryless controllers can be characterized by a feedback map:

$$g : \mathbf{Q} \times \mathbf{X} \to 2^{\mathbf{U}}$$

To prevent technical problems, we again restrict our attention to memoryless controllers such that for all $(q, x) \in \mathbf{Q} \times \mathbf{X}$ $\emptyset \neq g(q, x) \subseteq \phi(q, x)$. Given a plant, H, and a memoryless controller, g, we can defined the closed loop open hybrid automaton, $H_g = (Q,\ X,\ V,\ \mathrm{I}nit,\ f,\ I,\ E,\ G,\ R, \phi_g)$, where $\phi_g(q, x) = \phi(q, x) \cap g(q, x)$. It is easy to show that:

Proposition 6.1 *If C is memoryless controller with feedback map g, then $\mathcal{H}_g = \mathcal{H}_C$.*

This property allows one to nest controller synthesis problems, provided they can be solved by memoryless controllers. In general, is unclear whether the set of closed loop causal executions is the set of executions of some hybrid automaton.

For properties of the form $(Q \cup X, \Box F)$, it turns out that it it suffices to look for a solution among memoryless controllers.

Proposition 6.2 *A controller that satisfies property $(Q \cup X, \Box F)$ exists if and only if a memoryless controller that satisfies $(Q \cup X, \Box F)$ exists.*

Proof: The if part is obvious. For the only if part, assume that there exists a controller C that solves the synthesis problem $(H, \Box F)$, but there does not exist a feedback controller that solves the synthesis problem. Therefore, there must exist $(q, x) \in F$ and two different finite executions $\chi_1 = (\tau_1, q_1, x_1, (u_1, d_1)) \in \mathcal{H}_C$ and $\chi_2 = (\tau_2, q_2, x_2, (u_2, d_2)) \in \mathcal{H}_C$ ending in (q, x) such that $C(q_1, x_1) \neq C(q_2, x_2)$. Moreover, the "information" about whether (q, x) was reached via χ_1 or whether it was reached via χ_2 must be essential for subsequent control decisions.

More formally, assume (q, x) is reached via χ_2, and let χ' denote a subsequent execution, that is assume that the concatenation $\chi_2\chi'$ belongs to $\mathcal{H}$. Note that, since χ_1 also ends in (q, x), $\chi_1\chi'$ also belongs to $\mathcal{H}$. Let $\chi_2\chi' = (\tau_2', q_2', x_2', (u_2', d_2'))$ and $\chi_1\chi' = (\tau_1', q_1', x_1', (u_1', d_1'))$. Assume that for all $t \in \tau_2' \setminus \tau_2$, a control $u(t) \in C((q_1', x_1') \downarrow_t)$ is applied (instead of a control $u(t) \in C((q_2', x_2') \downarrow_t)$). Then, as the fact that (q, x) was reached via χ_2 is essential, there must exist a subsequent execution χ' such that $\chi_2\chi' \in \mathcal{H}$ (in fact $\chi_2\chi' \in \mathcal{H} \setminus \mathcal{H}_C$) and $\Box F(\chi_2\chi') = \text{False}$. This implies that there exists $t \in \tau_2'$ such that $(q_2'(t), x_2'(t)) \in F^c$. Since C is assumed to solve the synthesis problem and $\chi_2 \in \mathcal{H}_C$, $\Box F(\chi_2) = \text{True}$, therefore $t \in \tau_2' \setminus \tau_2$.

However, since for all $t \in \tau_2' \setminus \tau_2$, $u(t) \in C((q_1', x_1') \downarrow_t)$, and $(\tau_1', q_1', x_1', (u_1', d_1')) \in \mathcal{H}$, we have that $\chi_1\chi' \in \mathcal{H}_C$. But the above discussion indicates that there exists $t \in \tau_1'$ (in fact $t \in \tau_1' \setminus \tau_1$) such that $(q_1'(t), x_1'(t)) \in F^c$. This contradicts the assumption that C solves the synthesis problem $(H, \Box F)$. ∎

Motivated by Proposition 6.2, we restrict our attention to feedback controllers. For brevity, we refer to the problem of finding a controller for a plant H that satisfies a specification $(Q \cup X, \Box F)$ as the *controller synthesis problem* $(H, \Box F)$.

Controlled Invariant Sets

Typically, for a controller synthesis problem one treats the set of initial conditions, I*nit*, as variable and attempts to establish the largest set of states for which there exists a controller that satisfies the specification. This set of initial conditions turns out to be a "controlled invariant set".

Definition 6.3 (Controlled Invariant) *A set $W \subseteq \mathbf{Q} \times \mathbf{X}$ is called controlled invariant if there exists a controller that solves the controller synthesis problem $(H', \Box W)$ when H' is identical to H except for* I*nit*$'$ *which is equal to W.*

A controlled invariant set W is called *maximal* if it is not a proper subset of another controlled invariant set. We say a controller *renders W invariant* if it solves the controller synthesis problem $(H', \Box W)$ where I*nit*$' = W$.

Proposition 6.4 *A controller that solves the synthesis problem $(H, \Box F)$ exists if and only if there exists a unique maximal controlled invariant $W \subseteq \mathbf{Q} \times \mathbf{X}$ such that* I*nit* $\subseteq W \subseteq F$.

Proof: If there exists any control invariant $W \subseteq F$ (in particular, if there exists a unique maximal one) then, by definition, the synthesis problem $(H, \Box F)$ can be solved for $I = W$.

For the only if part, if the synthesis problem can be solved for some I*nit*, there exists a set $\widehat{\text{I}nit}$ and a feedback controller g such that for all d and for all $(q_0, x_0) \in \widehat{\text{I}nit}$ the execution $(\tau, q, x, (u, d))$ with $u(t) \in g(q(t), x(t))$

for all $t \in \tau$ satisfies $(q(t), x(t)) \in F$ for all $t \in \tau$. Consider the set:

$$W = \bigcup_{d} \bigcup_{(q_0,x_0)\in\widehat{Init}} \bigcup_{t\in\tau} (q(t), x(t))$$

Then clearly $W \subseteq F$. Moreover, for any $(q_0, x_0) \in W$ consider the execution $(\tau, q, x, (u, d))$ with arbitrary $d \in \mathcal{D}$ and $u(t) \in g(q(t), x(t))$. Then, by definition of W, $(q(t), x(t)) \in W$ for all $t \in \tau$. Therefore, controller g renders the set W invariant.

Having established the existence of controlled invariant subsets of F, consider now two such sets $W_1 \subseteq F$ and $W_2 \subseteq F$. We show that their union is also a controlled invariant subset of F. Clearly $W_1 \cup W_2 \subseteq F$. For $i = 1, 2$, as W_i is controlled invariant, there exists a feedback controller g_i that solves the controller synthesis problem $(H, \Box W_i)$, with $Init = W_i$. Consider the feedback controller g with:

$$g(q, x) = \begin{cases} g_1(q, x) & \text{if } (q, x) \in W_1 \\ g_2(q, x) & \text{otherwise} \end{cases}$$

Consider an arbitrary $(q_0, x_0) \in W_1 \cup W_2$. Then either $(q_0, x_0) \in W_1$ or $(q_0, x_0) \in (W_1 \cup W_2) \setminus W_1 \subseteq W_2$. In the first case, all executions are guaranteed to satisfy $\Box W_1$ as g_1 renders W_1 invariant. For the second case, consider an arbitrary execution $\chi = (\tau, q, x, (u, d))$ with $u(t) \in g(q(t), x(t))$ for all $t \in \tau$. Since g_2 solves the controller synthesis problem $(H, \Box W_2)$ with $Init = W_2$, either $\Box(W_2 \setminus W_1)(\chi) = \text{True}$ or $(q, x) \in W_2 \setminus W_1$ until $(q, x) \in W_1$, which brings us back to the first case. Hence, g solves the controller synthesis problem $(H, \Box(W_1 \cup W_2))$ with $Init = W_1 \cup W_2$, and the set $W_1 \cup W_2$ is controlled invariant.

Summarizing, the class of controlled invariant subsets of F is closed under union. Hence, it possesses a unique maximal element. ∎

Least Restrictive Controllers

We would like to derive a memoryless controller that solves the problem while imposing minimal restrictions on the controls it allows. There are at least two reasons why such a controller is desirable:

1. As discussed above, safety properties can sometimes be satisfied using trivial controllers (that cause deadlocks or zeno executions for example). Imposing as few restrictions as possible allows us to find a meaningful controller whenever possible.

2. In many cases multiple, prioritized specifications are given for a particular problem. Imposing fewer restrictions on the controls when designing controllers for higher priority specifications allows us greater flexibility when trying to satisfy lower priority specifications.

Memoryless controllers that solve the synthesis problem $(H, \Box F)$ can be partially ordered by the relation:

$$g_1 \preceq g_2 \Leftrightarrow g_1(x) \subseteq g_2(x) \text{ for all } x \in \mathbf{X}$$

Definition 6.5 *A memoryless controller that solves* $(H, \Box F)$ *is called* least restrictive *if it is maximal among the controllers that solve* $(H, \Box F)$.

There is a conjecture that for every controller synthesis problem $(H, \Box F)$ either there is no solution or there exists a unique least restrictive controller that solves the problem. As of now there is no proof of this fact however.

Some Remarks on "Implementation"

The notion of a controller introduced above may be inadequate when it comes to implementation. For one thing, the set valued map g allows nondeterministic choices of control inputs. Since in practice only one input can be applied to the system at any time, this nondeterminism has to somehow be resolved when it comes time to implement such a controller. The set valued map can in this sense be thought of as a family of single valued controllers; implementation involves choosing one controller from this family.

Normally, one would "implement" a controller by another hybrid automaton, which, when composed with the plant automaton yields the desired behavior. To do this one would need to introduce output variables to the hybrid automaton and define formal semantics for composition, as in Lecture 8. The process is slightly more complicated for the models considered here because of the presence of the state dependent input constraints, encoded by ϕ. We will assume that the entire state is available to the controller. In general this will not be the case. If a controller is to be implemented by a hybrid automaton, the information the controller has about the plant is obtained through the valuations of the output variables of the plant, which are not necessarily in one to one correspondence with the valuations of the state variables. The controller synthesis problem under partial observation (output feedback) is much more complicated than the full observation (state feedback) problem addressed here (partly because it makes it harder to define composition as discussed above).

6.2 Game Theoretic Approach to Controller Synthesis

To guarantee that a safety specification is met despite the action of the disturbances we cast the design problem as a zero sum dynamic game. The two players in the game are the control u and the disturbance d and they compete over cost a function that encodes the safety specification. We seek

the best possible control action and the worst possible disturbance. Note that if the specifications can be met for this pair then they can also be met for any other choice of the disturbance.

Consider a controller synthesis problem $(H, \Box F)$. The game can be cast in the standard min-max setting by introducing a cost function induced by the discrete metric. The discrete metric is simply a map $m : (\mathbf{Q} \times \mathbf{X}) \times (\mathbf{Q} \times \mathbf{X}) \longrightarrow \mathbb{R}$ defined by:

$$m((q_1, x_1), (q_2, x_2)) = \begin{cases} 0 & \text{if } (q_1, x_1) = (q_2, x_2) \\ 1 & \text{if } (q_1, x_1) \neq (q_2, x_2) \end{cases}$$

It is easy to check that m satisfies the axioms of a metric. The metric induces a map on subsets of $\mathbf{Q} \times \mathbf{X}$ by defining:

$$\begin{aligned} M : 2^{\mathbf{Q}\times\mathbf{X}} \times 2^{\mathbf{Q}\times\mathbf{X}} &\longrightarrow \mathbb{R} \\ (W_1, W_2) &\longmapsto \min_{((q_1,x_1),(q_2,x_2))\in W_1\times W_2} m((q_1, x_1), (q_2, x_2)) \end{aligned}$$

In other words, $M(W_1, W_2) = 0$ if $W_1 \cap W_2 \neq \emptyset$ and $M(W_1, W_2) = 1$ if $W_1 \cap W_2 = \emptyset$.

Consider an execution, $\chi = (\tau, q, x, (u, d))$, of the hybrid automaton H starting at an initial state $(q_0, x_0) \in I$. Define the *cost* of this execution by:

$$\begin{aligned} J : \mathcal{H} &\longrightarrow \mathbb{R} \\ \chi &\longmapsto \min_{t\in\tau} M(\{(q(t), x(t))\}, F^c) \end{aligned}$$

Note that J can only take on two values, $J(\chi) \in \{0, 1\}$. Therefore, J implicitly defines a property of the hybrid automaton. In fact:

Proposition 6.6 *$J(\chi) = 1$ if and only if $\Box F(\chi) = True$.*

Intuitively, u tries to maximize the cost function J (prevent the state from leaving F). Because we have no control over the actions of d, we assume that it tries to minimize J (force the state to leave F). As we would like to establish conditions under which $\Box F$ is guaranteed to be satisfied we bias the game in favor of the disturbance whenever there is ambiguity over how the game will proceed. For example, multiple executions may be possible for the same initial condition, control and disturbance trajectories, due to nondeterminism. Moreover, the order in which the two players play in the game may be important, if one player is assumed to have access to the decision of the other player before choosing his/her action. In both these cases we would like to give the disturbance the "benefit of the doubt".

Consider the max-min solution:

$$J^*(q_0, x_0) = \max_g \min_d \left(\min_{\chi=(\tau,q,x,(u,d))\in\mathcal{H}_g} J(\chi) \right) \tag{6.1}$$

Motivated by Proposition 6.7 we restrict our attention to feedback strategies for u in equation (6.1). Following the standard game theoretic convention, the "player" who appears first in the right hand side of equation

(6.1) (the controller g) is also assumed to play first. The player who appears second (the disturbance d) is assumed to have access to the strategy of the first player, when called upon to make his/her decision. The minimum over χ removes all nondeterminism. Therefore, provided a solution to this equation can be found, J^* is a well defined function of the initial state (q_0, x_0). In addition, the minimum over χ implicitly restricts attention to control and disturbance trajectories that satisfy the state based input constraint ϕ. Using J^* we define a set $W^* \subseteq \mathbf{Q} \times \mathbf{X}$ by:

$$W^* = \{(q_0, x_0) \in \mathbf{Q} \times \mathbf{X} | J^*(q_0, x_0) = 1\} \tag{6.2}$$

Proposition 6.7 *W^* is the maximal controlled invariant subset of F.*

Proof: We first show W^* is controlled invariant. Assume for the sake of contradiction that it is not. Then for all g there exists an $(q_0, x_0) \in W^*$, a d, a $\chi = (\tau, q, x, (u, d)) \in \mathcal{H}_g$ and a $t \in \tau$ with $(q(t), x(t)) \notin W^*$. By Proposition 6.6, $J(\chi) = 0$, which, by equation (6.1), implies that $J^*(q_0, x_0) = 0$. This contradicts the assumption that $(q_0, x_0) \in W^*$.

Next we show that W^* is maximal. Assume for the sake of contradiction that it is not, that is there exists another controlled invariant $\hat{W}$ with $W^* \subset \hat{W} \subseteq F$. Then, by definition, there exists a controller g such that $\mathcal{H}_g$ satisfies $\Box F$ with $I = \hat{W}$. In other words, for all $(q_0, x_0) \in \hat{W}$, for all d and for all $\chi = (\tau, q, x, (u, d)) \in \mathcal{H}_g$, $\Box F(\chi) =$ True, or, equivalently, $J(\chi) = 1$. But, from equation (6.1) this would imply that $J^*(q_0, x_0) = 1$. This contradicts the assumption that $W^* \subset \hat{W}$. ∎

If a solution to equation 6.1 can be computed, then there exists a feedback controller (namely one that achieves the maximum) that renders W^* invariant. We would like to find a least restrictive such controller. Our ability to solve the synthesis problem using this technique hinges on finding a solution to equation 6.1. In some cases this can be done by brute force; this typically involves guessing a controller and showing that it achieves the maximum. More systematically, this can be done using optimal control techniques, in particular dynamic programming.

6.2.1 Example: The Steam Boiler

The problem of controlling a steam boiler was introduced first in [131, 132]. The model presented here is taken from [133]. The controller synthesis for this model can be found in [110]. The continuous dynamics of the boiling process are summarized by the differential equations:

$$\begin{aligned} \dot{w} &= p_1 + p_2 - r \\ \dot{r} &= d \end{aligned}$$

The dynamics of pump i are summarized by the open hybrid automaton of Figure 6.2. Notice that p_i is both an output variable of the pump and an

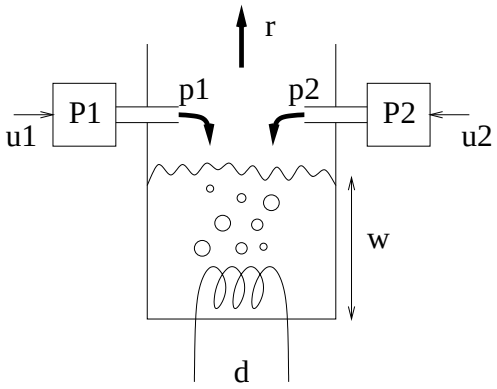

Figure 6.1. The Steam Boiler

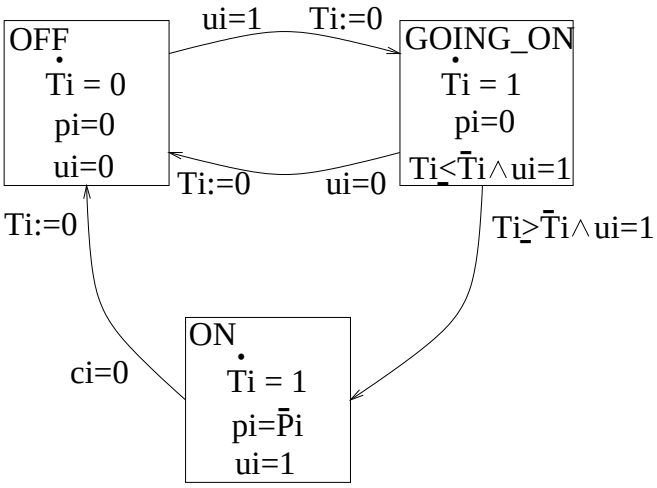

Figure 6.2. The pump hybrid automaton

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:

$$x = (w, r, T_1, T_2) \in \mathbb{R}^4$$

9 discrete states:

$$q \in \{(OFF, OFF), (OFF, GOING\_ON), \ldots, (ON, ON)\}$$

2 discrete input variables:

$$(u_1, u_2) \in \{0, 1\} \times \{0, 1\} = \mathbf{U}$$

and one continuous input variable:

$$d \in [-D_1, D_2] = \mathbf{D}$$

s 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:

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

Proposition 6.8 *If* $Init \subseteq \mathbf{Q} \times \mathbb{R} \times [0,R] \times \mathbb{R}^2$, *then for all* $\chi = (\tau, q, x, u_1, u_2, d)$ *and for all* $t \in \tau$, $r(t) \in [0, R]$.

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 $(Q \cup X, \Box F)$ with

$$F = \mathbf{Q} \times [M_1, M_2] \times \mathbb{R}^3$$

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:

$$J^*(q_0, x_0) = \max_g \min_d \left(\min_{\chi=(\tau,q,x,(u,d)) \in \mathcal{H}_g} J(\chi) \right)$$

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 ϕ 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:

$$J_1(x^0, u_1, u_2, d) = \inf_{t \geq 0} w(t) \quad \text{and} \quad J_1(x^0, u_1, u_2, d) = -\sup_{t \geq 0} w(t) \tag{6.3}$$

Clearly:

$$J = 1 \Leftrightarrow (J_1 \geq M_1) \wedge (J_2 \geq -M_2)$$

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:

$$u_i^*(q,x) = 1 \text{ for all } (q,x), \quad \text{and} \quad d^*(q,x) = \begin{cases} D_2 & \text{if } r < R \\ 0 & \text{if } r = R \end{cases} \tag{6.4}$$

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

Lemma 6.9 *(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 [110]. ∎

A *saddle solution* is a solution to equation (6.1) 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):

$$J_1^*(q,x) = \max_{(u_1,u_2)} \min_d J_1((q,x),(u_1,u_2),d) = \min_d \max_{(u_1,u_2)} J_1((q,x),(u_1,u_2),d)$$

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:

$$W_1^* = \{(q,x) \ : \ J_1^*(q,x) \geq M_1\} = \{(q,x) \ : \ w \geq \hat{w}(r,T_1,T_2)\}$$

Two level sets of this function are shown in Figure 6.3.

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 6.10 *The feedback controller g_1^1 given by:*

$$\begin{aligned} &u_1 \in \{0,1\} \text{ and } u_2 \in \{0,1\} \text{ if } [w > \hat{w}(r,0,0)] \vee [w < \hat{w}(r,T_1,T_2)] \\ &u_1 = 1 \text{ and } u_2 \in \{0,1\} \text{ if } \hat{w}(r,0,0) \geq w > \hat{w}(r,T_1,0) \\ &u_1 \in \{0,1\} \text{ and } u_2 = 1 \text{ if } \hat{w}(r,0,0) \geq w > \hat{w}(r,0,T_2) \\ &u_1 = 1 \text{ and } u_2 = 1 \text{ if } w = \hat{w}(r,T_1,T_2) \end{aligned}$$

is the unique, least restrictive, non-blocking, feedback controller that renders W_1^{1*} invariant.

Proof: See [110]. ∎

Note that the first term applies to states in the interior of the safe set ($w > \hat{w}(r,0,0)$) as well as all the states outside the safe set ($w < \hat{w}(r,T_1,T_2)$). The expression for $\hat{w}$ (see [110]) suggests that $\hat{w}$ is monotone in T_1 and T_2.

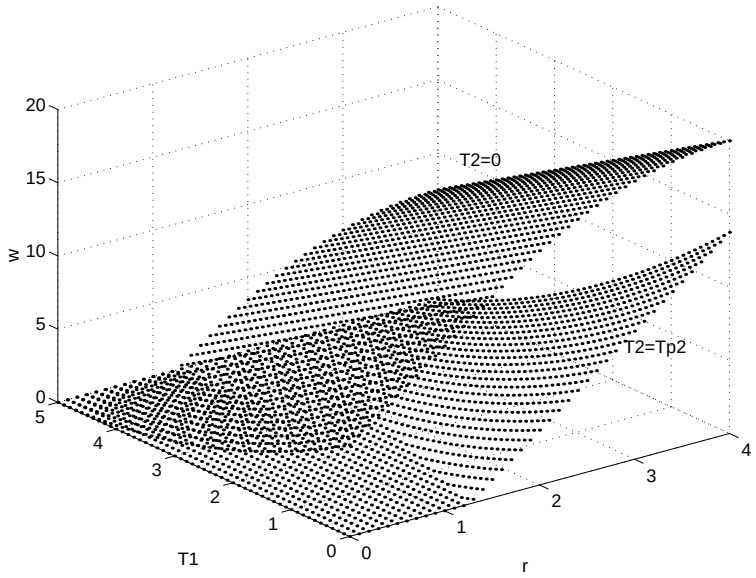

Figure 6.3. Lower limit on w to avoid draining

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).

6.2.2 Game Theoretic Controller Synthesis for Finite State Machines

Consider a plant automaton $H = (Q, X, V, \mathrm{I}nit, f, I, E, G, R, \phi)$, with:

- $Q = \{q\}$, $\mathbf{Q} < \infty$ (finite discrete states);
- $X = \{x\}$, $\mathbf{X} = \{0\}$ (trivial continuous state);
- $V = \{u, d\}$, $\mathbf{V} < \infty$ (finite inputs);
- $\mathrm{I}nit \subseteq \mathbf{Q} \times \{0\}$ (drop the trivial dependence on the continuous state from now on);
- $f(q, x, v) = 0$ (trivial continuous dynamics);
- $I(q) = \emptyset$ (no time evolution);
- $E \subset \mathbf{Q} \times \mathbf{Q}$;
- $G(e) \subseteq \{0\} \times \mathbf{V}$ (drop the trivial dependence on the continuous state from now on);
- $R(e, 0, v) = \{0\}$; and,

- $\phi(q, 0) = \mathbf{V}$.

Notice the similarity between this hybrid automaton and the finite automata considered earlier in the class. Recall that a finite automaton is a collection $(\mathbf{Q}, \Sigma, \Delta, q_0, Q_F)$ consisting of a finite set of states, a finite alphabet, a transition relation, an initial state and a set of final states. Comparing the two definitions we see that $\mathbf{Q}$ plays the same role in both cases, $\mathbf{V}$ plays the role of Σ, Init plays the role of q_0, while $Q_F = \mathbf{Q}$ (restrictions on where we should and should not end up will be encoded in terms of the specification later on). The transition relation is given by:

$$(q, v, q') \in \Delta \Leftrightarrow ((q, q') \in E) \wedge (v \in G(q, q'))$$

Recall that the transition relation can also be encoded by means of a transition map:

$$\delta(q, v) = \{q' \in \mathbf{Q} \ : \ ((q, q') \in E) \wedge (v \in G(q, q'))\}$$

To prevent trivial safe solutions we add a non-blocking assumption:

$$\forall q \in \mathbf{Q}, \ \forall u \in \mathbf{U}, \ \exists d \in \mathbf{D} \ \text{ such that } \delta(q, (u, d)) \neq \emptyset$$

Remarks:

1. The dependence on the trivial continuous state will be dropped to simplify the notation.

2. Strictly speaking the hybrid automaton definition of a finite state system is slightly more general, since many initial states are allowed. In any case, Init will not be very important, since we will be trying to determine the largest set if initial conditions for which a safety specification can be satisfied.

3. All the action takes place at a single point in time (the executions are over time sequences of the form $[0, 0][0, 0], \ldots$). An alternative interpretation is that time has been abstracted away. This can be encoded by setting $I(q) = \{0\}$ for all q; the interpretation is that the transition can take place after an unspecified amount of delay. In either case, the execution can be reduced down to a pair of sequences, one for the states ($q[i]$) and one for the inputs ($v[i]$) such that:

$$q[0] \in \text{I}nit \ \text{ and } \ \forall i \geq 0, q[i+1] \in \delta(q[i], v[i])$$

4. Because time has been abstracted away and $\phi(q) = \mathbf{V}$, the non-blocking assumption eliminates all possibilities of cheating by the controller (zeno executions are meaningless in this setting).

5. Players u and d play simultaneously. Subsumes turn based play [134] and priority play [109] (see [134] and [135]).

6.3 Controller Synthesis for Hybrid Systems

In this section, we bring the discrete and continuous parts together, and develop controller synthesis for hybrid systems. Our treatment follows [36] and [110].

Recall that the *plant* is modeled by an open hybrid automaton, $H = (Q, X, V, \mathrm{I}nit, f, I, E, G, R, \phi)$, where:

- Q is a finite collection of discrete state variables;
- X is a finite collection of continuous state variables;
- V is a finite collection of input variables. We assume $V = V_D \cup V_C$, where V_D contains discrete and V_C contains continuous variables.
- $\mathrm{I}nit \subseteq \mathbf{Q} \times \mathbf{X}$ is a set of initial states;
- $f : \mathbf{Q} \times \mathbf{X} \times \mathbf{V} \to \mathbb{R}^n$ is an input dependent vector field;
- $I : \mathbf{Q} \to 2^{\mathbf{X} \times \mathbf{V}}$ assigns to each $q \in \mathbf{Q}$ an input dependent invariant set;
- $E \subset \mathbf{Q} \times \mathbf{Q}$ is a collection of discrete transitions;
- $G : E \to 2^{\mathbf{X} \times \mathbf{V}}$ assigns to each $e = (q, q') \in E$ a guard;
- $R : E \times \mathbf{X} \times \mathbf{V} \to 2^{\mathbf{X}}$ assigns to each $e = (q, q') \in E$, $x \in \mathbf{X}$ and $v \in \mathbf{V}$ a reset relation; and,
- $\phi : \mathbf{Q} \times \mathbf{X} \to 2^{\mathbf{V}}$ assigns to each state a set of admissible inputs.

Also recall that the set the input variables are partitioned into *controls* (that can be used to steer the system) and *disturbances* (whose values can not be controlled):

$$V = U \cup D$$

Assumption 6.11 *To avoid technical problems we assume that:*

1. *f is Lipschitz continuous in x and continuous in v.*
2. *for all $(q, x) \in \mathbf{Q} \times \mathbf{X}$, $\phi(q, x) = \mathbf{U} \times \mathbf{D} \neq \emptyset$.*
3. *for all $q \in \mathbf{Q}$ and for all $v \in \mathbf{V}$, $I(q)|_X$ is an open set.*
4. *for all $(q, x) \in \mathbf{Q} \times \mathbf{X}$, and for all $u \in \mathbf{U}$ there exists $d \in \mathbf{D}$ such that:*

$$[(x, u, d) \in I(q)] \vee [((x, u, d) \in G(q, q') \wedge (R(q, q', x, u, d) \neq \emptyset)]$$

Part 1 is standard, and is needed for existence of continuous evolution. Part 2 implies that acceptable control and disturbance inputs always exists, and that the choice of u can not influence the possible choices of d and vice versa. Part 3 implies that we need not worry about what happens on the boundary of the invariant set (otherwise we would have to assume

transverse invariants, define the Out set, etc.). Finally, part 4 (together with part 3) implies that the controller can not block the system execution. Notice that this assumption is not symmetric for u and d. The reason is that, since we are dealing with safety specifications it will never be to the benefit of d to stop the execution.

As before we will try to establish the largest controlled invariant subset of a given set $F \subseteq \mathbf{Q} \times \mathbf{X}$, and design a controller that renders this set invariant. To avoid technical problems we assume that:

Assumption 6.12 *F is a closed set.*

We will again take an adverserial approach and treat the design as a game between u and d. Whenever possible we will give the advantage to d, in particular:

1. In the order of play (u will be the "leader" of the game).
2. When resolving non-determinism.

6.4 Definitions of Operators

Notice that the disturbance has two choices. It can:

1. Try to make the system "jump" outside F.
2. Try to "steer" the system outside F along continuous evolution.

The control also has two choices:

1. Try to "jump" to another state in F when the disturbance tries to steer out of F.
2. Try to "steer" the system and keep it in F along continuous evolution.

To charactrize alternative 1 for the disturbance, introduce the *uncontrollable predecessor operator*, $\mathrm{P}re_d : 2^{\mathbf{Q}\times\mathbf{X}} \to 2^{\mathbf{Q}\times\mathbf{X}}$, which, given a set $K \subseteq \mathbf{Q} \times \mathbf{X}$ returns:

$$\mathrm{P}re_d(K) = K^c \cup \{(q,x) \in \mathbf{Q}\times\mathbf{X} \ : \quad \forall u \in \mathbf{U}\ \exists d \in \mathbf{D}, q' \in \mathbf{Q} \text{ such that } [(x,u,d) \in G(q,q')] \wedge [R(q,q',x,u,d) \cap K^c \neq \emptyset]\}$$

For a given K, $\mathrm{P}re_d(K)$ returns the set of states that are either in the complement of K, or whatever u does may find themselves in the complement of K by a discrete transition.

To charactrize alternative 1 for the control, introduce the *controllable predecessor operator*, $\mathrm{P}re_u : 2^{\mathbf{Q}\times\mathbf{X}} \to 2^{\mathbf{Q}\times\mathbf{X}}$, which, given a set $K \subseteq \mathbf{Q}\times\mathbf{X}$

returns:

$$\begin{aligned}\mathrm{P}re_u(K) = K \cap \{(q,x) \in \mathbf{Q}\times\mathbf{X} \ : \quad & \exists u \in \mathbf{U} \text{ such that } \forall d \in \mathbf{D}, \\ & [\exists q' \in \mathbf{Q} \text{ such that } (x,u,d) \in G(q,q')] \wedge \\ & [(x,u,d) \notin I(q)] \wedge \\ & [(x,u,d) \in G(q,q') \rightarrow R(q,q',x,u,d) \subseteq K]\}\end{aligned}$$

For a given K, $\mathrm{P}re_u(K)$ returns the set of states that are already in K and there exists a control action that can force a transition that keeps the state in K.

Some simple facts about these two operators:

Proposition 6.13 *For all* $K \subseteq \mathbf{Q}\times\mathbf{X}$, $\mathrm{P}re_u(K) \subseteq K$, $\mathrm{P}re_d(K) \supseteq K^c$ *and* $\mathrm{P}re_u(K) \cap \mathrm{P}re_d(K) = \emptyset$

Remarks:

- The two operators are asymmetric.
- The order of the quantifiers is consistent with u being the leader in the game.
- Since all non-determinism is resolved in favor of d, u has to work harder:
  - it has to ensure a transition back into K exists (first condition in the definition of $\mathrm{P}re_u(K)$),
  - it has to be able to "force" the transition (no mention of $I(q)$ in the definition of $\mathrm{P}re_d(K)$),
  - it has to ensure all possible transitions stay in K (last condition in the definition of $\mathrm{P}re_u(K)$).

Finally, to characterize alternative 2 for bot u and d introduce the *reach-avoid operator*, $\mathrm{P}re_u : 2^{\mathbf{Q}\times\mathbf{X}} \times 2^{\mathbf{Q}\times\mathbf{X}} \rightarrow 2^{\mathbf{Q}\times\mathbf{X}}$, which, given two disjoint sets $K \subseteq \mathbf{Q}\times\mathbf{X}$ and $L \subseteq \mathbf{Q}\times\mathbf{X}$ returns:

$$\begin{aligned}\mathrm{R}each(K,L) = \{(q_0,x_0) \in \mathbf{Q}\times\mathbf{X} \ : \quad & \forall u \in \mathcal{U}\ \exists d \in \mathcal{D}, t \geq 0 \text{ such that} \\ & [(q(t),x(t)) \in K)] \wedge [\forall t' \in [0,t]\ ((q(t'),x(t')) \notin L]\end{aligned}$$

where $(q,x) : [0,t] \rightarrow \mathbf{Q}\times\mathbf{X}$ is a segment of continuous evolution with inputs u and d, i.e.:

$$\begin{aligned}(q(0),x(0)) &= (q_0,x_0) \text{ and } \forall t' \in [0,t] \\ q(t') &= q_0 \\ (x(t'),u(t'),d(t')) &\in I(q_0) \\ \dot{x}(t') &= f(q_0,x(t'),u(t'),d(t'))\end{aligned}$$

Given two disjoint sets K and L, the operator R*each* returns the set of states for which whatever u does, d can chose an appropriate trajectory to bring the state of the system to K along continuous evolution, without going first through L.

Proposition 6.14 *For all* $K, L \subseteq \mathbf{Q} \times \mathbf{X}$ *with* $K \cap L = \emptyset$, $K \subseteq \mathrm{R}each(K, L) \subseteq L^c$.

Notice that the definition of R*each* is somewhat informal, since:

- u is implicitly assumed to be a feedback strategy (in fact, without loss of generality a *memoryless* feedback strategy, see Lecture 21).
- u and d are allowed to be piecewise continuous (recall the justification for this given in Lecture 26).

6.5 Basic Algorithm

Using the above definitions, the following algorithm can now be formulated for computing the largest controlled invariant subset of a given set F.

Algorithm 4 (Controlled Invariant Set)

Initialization:
$W^0 = F$, $W^1 = \emptyset$, $i = 0$
while $W^i \neq W^{i+1}$ **do**
begin
$W^{i-1} = W^i \setminus \mathrm{R}each(\mathrm{P}re_d(W^i), \mathrm{P}re_u(W^i))$
$i = i - 1$
end

Pictorially, the operations involved in one step of the algorithm is illustrated in Figure 6.4.

Proposition 6.15 *If the algorithm terminates in a finite number of steps, the fixed point* W^* *is the maximal controlled invariant subset of* F.

Proof: Clearly, $W^* \subseteq \ldots \subseteq W^{i-1} \subseteq W^i \subseteq \ldots \subseteq F$. Assume that the algorithm terminates in a finite number of steps. Then:

$$\begin{aligned} & W^* = W^* \setminus \mathrm{R}each(\mathrm{P}re_d(W^*), \mathrm{P}re_u(W^*)) \\ \Leftrightarrow \quad & W^* \cap \mathrm{R}each(\mathrm{P}re_d(W^*), \mathrm{P}re_u(W^*)) \\ \Leftrightarrow \quad & \mathrm{R}each(\mathrm{P}re_d(W^*), \mathrm{P}re_u(W^*)) \subseteq (W^*)^c \end{aligned}$$

But:

$$\begin{aligned} & (W^*)^c \subseteq \mathrm{P}re_d(W^*) \subseteq \mathrm{R}each(\mathrm{P}re_d(W^*), \mathrm{P}re_u(W^*)) \subseteq (W^*)^c \\ \Leftrightarrow \quad & \mathrm{R}each(\mathrm{P}re_d(W^*), \mathrm{P}re_u(W^*)) = \mathrm{P}re_d(W^*) = (W^*)^c \end{aligned}$$

Consider an arbitrary $(q_0, x_0) \in W^*$. Then:

$$(q_0, x_0) \notin \mathrm{R}each(\mathrm{P}re_d(W^*), \mathrm{P}re_u(W^*))$$

Taking the negation of the expression for R*each* this becomes:

$\exists u \in \mathcal{U}$ such that $\forall d \in \mathcal{D}, \forall t \geq 0 \;\; [(q(t), x(t)) \notin \mathrm{P}re_d(W^*))] \vee [\exists t' \in [0, t] \; ((q(t'), x(t')) \in \mathrm{P}re_u(W^*)]$

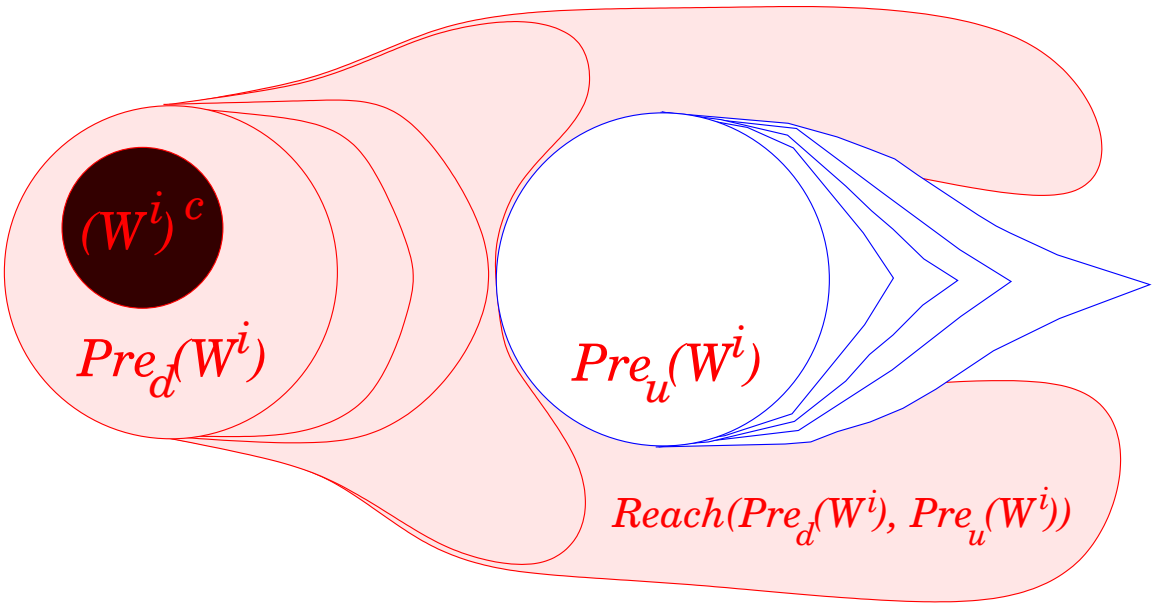

Figure 6.4. One step of the algorithm.

Replacing $\mathrm{P}re_d(W^*)$ by $(W^*)^c$ and substituting the definition of $\mathrm{P}re_u(W^*)$:

$$\begin{aligned}\exists u \in \mathcal{U} \text{ such that } \forall d \in \mathcal{D},\ \forall t \geq 0 \quad & [(q(t), x(t)) \in W^*] \vee \\ & [\exists t' \in [0,t] \text{ such that } \exists u(t') \in \mathbf{U} \text{ such that } \forall d(t') \in \mathbf{D}, \\ & (\exists q' \in \mathbf{Q} \text{ such that } (x(t'), u(t'), d(t')) \in G(q(t'), q')) \wedge \\ & ((x(t'), u(t'), d(t')) \notin I(q(t'))) \wedge \\ & ((x(t'), u(t'), d(t')) \in G(q(t'), q') \rightarrow R(q(t'), q', x(t'), u(t'), d(t')) \subseteq W\end{aligned}$$

In other works, if $(q_0, x_0) \in W^*$ either u can keep the system forever in W^* without any discrete transitions taking place, or it can drive the system so that $(q(\tau_1), x(\tau_1)) \in W^*$ (the first discrete transition the state is again in W^*). Controlled invariance of W^* follows by induction.

To show maximality, consider an arbitrary $(q_0, x_0) \in (W^*)^c$. Then, since the algorithm terminated in a finite number of steps, there exists i such that:

$$\begin{aligned} & (q_0, x_0) \in W^i \setminus W^{i-1} \\ \Leftrightarrow\ & (q_0, x_0) \in \mathrm{R}each(\mathrm{P}re_d(W^i), \mathrm{P}re_u(W^i)) \\ \Leftrightarrow\ & \forall u \in \mathcal{U}\ \exists d \in \mathcal{D}, t \geq 0 \text{ such that } [(q(t), x(t)) \in \mathrm{P}re_d(W^i)] \wedge [\forall t' \in [0,t],\ ((q(t'), x(t')) \notin \mathrm{P}re_u(W^i)]\end{aligned}$$

Substituting the definition of $\mathrm{P}re_d$ leads to:

$$\begin{aligned}\forall u \in \mathcal{U}\ \exists d \in \mathcal{D},\ \exists t \geq 0 \text{ such that } \quad & [\forall t' \in [0,t]\ ((q(t'), x(t')) \notin \mathrm{P}re_u(W^i)] \wedge \\ & [((q(t), x(t)) \in (W^i)^c) \vee \\ & (\forall u(t) \in \mathbf{U}\ \exists d(t) \in \mathbf{D},\ q' \in \mathbf{Q} \text{ such that } [(x(t), u(t), d(t)) \in G(q(t), \\ & [R(q(t), q', x(t), u(t), d(t)) \cap (W^i)^c \neq \emptyset])\end{aligned}$$

Since $W^j \subseteq W^i$, for all $j \leq i$, the above shows that in finite time the state may end up either in F^c or in W^j for some $j > i$ whatever the control does (at best it ends up in W^{i+1}). Therefore, by induction, the state leaves F in finite time. ■

Chapter 7
Computational Methods (Claire)

7.1 Flight Level Control: A Numerical Case Study

To illustrate the results of Section 5.3.3, we consider the problem of maintaining an aircraft at a desired flight level. Commercial aircraft at cruising altitudes are typically assigned a flight level by Air Traffic Control (ATC). The flight levels are separated by a few hundred feet (e.g. 500 or 1000, depending on altitude and the type of airspace). Air traffic moves in different directions at different flight levels (north to south in one level, east to west in another, etc.). This arrangement is desirable because it greatly simplifies the task of ATC: the problem of ensuring aircraft separation, which is normally three dimensional, can most of the time be decomposed to a number of two dimensional problems.

Changes in the flight level happen occasionally and have to be cleared by ATC. At all other times the aircraft have to ensure that they remain within certain bounds (e.g. ± 250 feet) of their assigned level. At the same time, they also have to maintain certain limits on their speed, flight path angle, acceleration, etc. imposed by limitations of the engine and airframe, passenger comfort requirements, or to avoid dangerous situations such as aerodynamic stall. In this section we formulate a a SupMin optimal control problem that allows us to address such constraints.

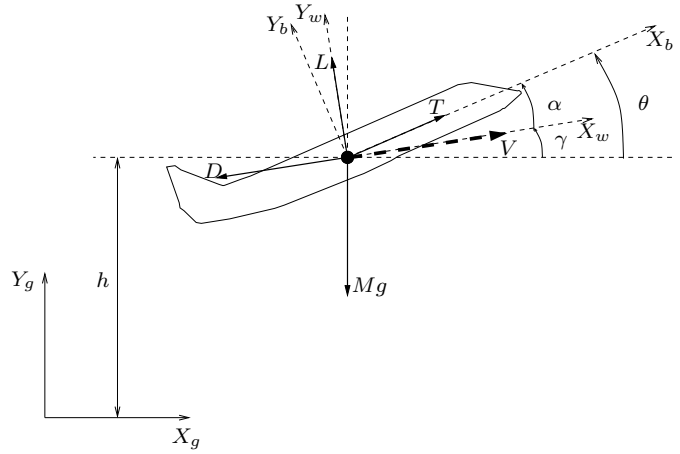

Figure 7.1. Coordinate frames and forces for the aircraft model.

7.1.1 Aircraft Model

We restrict our attention to the movement of the aircraft in the vertical plane and describe the motion using a point mass model. Such models are commonly used in ATC research (see, for example, [110, 136]). They are fairly simple, but still capture the essential features of aircraft flight. The analysis presented here is an extension to three dimensions of an aerodynamic envelope protection problem studied in [110].

Three coordinate frames are used to describe the motion of the aircraft: the ground frame (X_g–Y_g), the body frame (X_b–Y_b) and the wind frame (X_w–Y_w). The angles of rotation between the frames are denoted by θ (ground to body frame, known as the *pitch angle*), γ (ground to wind frame, known as the *flight path angle*) and α (wind to body frame, known as the *angle of attack*). $V \in \mathbb{R}$ denotes the speed of the aircraft (aligned with the positive X_w direction) and h its altitude. Figure 7.1 shows the different forces applied to the aircraft: its weight (mg, acting in the negative Y_g direction), the aerodynamic lift (L, acting in the positive Y_w direction), the aerodynamic drag (D, acting in the negative X_w direction) and the thrust exerted by the engine (T, acting in the positive X_b direction).

A force balance leads to the following equations of motion

$$\begin{aligned} m\dot{V} &= T\cos(\alpha) - D - mg\sin(\gamma) \\ mV\dot{\gamma} &= L + T\sin(\alpha) - mg\cos(\gamma). \end{aligned}$$

From basic aerodynamics, the lift and drag can be approximated by

$$L = \frac{C_L S \rho V^2}{2}(1 + c\alpha) = a_L V^2 (1 + c\alpha)$$
$$D = \frac{C_D S \rho V^2}{2} = a_D V^2,$$

where, C_L, C_D, and c are (dimension-less) lift and drag coefficients, S is the wing surface area, ρ is the air density and, as is commonly done in practice, the dependence of the drag on the angle of attack has been suppressed.

A three state model with $x_1 = V$, $x_2 = \gamma$ and $x_3 = h$ suffices for our purposes. The system is controlled by two inputs, the thrust, $u_1 = T$, and the pitch angle[1], $u_2 = \theta$. We assume rectangular bounds on the inputs, $u \in U = [T_{\min}, T_{\max}] \times [\theta_{\min}, \theta_{\max}]$. After a small angle approximation on α (valid for airliners, which usually operate around trimmed flight conditions) the equations of motion become

$$\dot{x} = f(x,u) = \begin{bmatrix} -\frac{a_D}{m} x_1^2 - g\sin(x_2) \\ \frac{a_L}{m} x_1 (1 - c x_2) - g\frac{\cos(x_2)}{x_1} \\ x_1 \sin(x_2) \end{bmatrix} + \begin{bmatrix} \frac{1}{m} \\ 0 \\ 0 \end{bmatrix} u_1 + \begin{bmatrix} 0 \\ \frac{a_L c}{m} x_1 \\ 0 \end{bmatrix} u_2 \tag{7.1}$$

7.1.2 Cost Function and Optimal Controls

For safety reasons, certain combinations of speed and flight path angle should be avoided, because they may result in aerodynamic stall. Part of the task of the controller is therefore to keep V and γ within a safe "aerodynamic envelope". Following [110], we consider a simplified rectangular envelope; improvements that can be introduced to make the envelope more realistic are discussed in [137, 105]. We require that

$$V_{\min} \le x_1 \le V_{\max} \text{ and } \gamma_{\min} \le x_2 \le \gamma_{\max},$$

for some $V_{\min} \le V_{\max}$ and $\gamma_{\min} \le \gamma_{\max}$. In addition, to ensure that the aircraft does not stray away from its flight level we require that for some $h_{\min} \le h_{\max}$,

$$h_{\min} \le x_3 \le h_{\max}.$$

We set[2] $K = [V_{\min}, V_{\max}] \times [\gamma_{\min}, \gamma_{\max}] \times [h_{\min}, h_{\max}]$.

[1] In practice, one can only control the second derivative of the pitch angle using the aerodynamic surfaces. This makes the model weakly non-minimum phase. Here we ignore this complication.

[2] Strictly speaking, to follow the development on Section **??** one needs to assume that the set K is open. It is easy to see, however, that allowing K to be closed makes no difference in this case.

To encode these constraints as a cost in a SupMin problem we define a function $l(\cdot) : \mathbb{R}^3 \to \mathbb{R}$ by

$$l(x) = \min\{x_1 - V_{\min}, x_2 - \gamma_{\min}, x_3 - h_{\min}, V_{\max} - x_1, \gamma_{\max} - x_2, h_{\max} - x_3\}.$$

Notice that $l(x) \geq 0$ for $x \in K$ and $l(x) < 0$ for $x \notin K$. Clearly, l is Lipschitz continuous. To keep l bounded (and since we are only interested in the behaviour around the set K) we "saturate" the function l outside the set $[V_{\min} - \delta V, V_{\max} + \delta V] \times [\gamma_{\min} - \delta\gamma, \gamma_{\max} + \delta\gamma] \times [h_{\min} - \delta h, h_{\max} + \delta h]$ for $\delta V, \delta\gamma, \delta h > 0$.

The problem is now in a form that we can apply the results of Section 5.3.3. The Hamiltonian of equation (5.34) becomes

$$\begin{aligned} H_1(p,x) = \min\Big\{0, p_1\Big(-\frac{a_D}{m}x_1^2 - g\sin(x_2)\Big) \\ + p_2\left(\frac{a_L}{m}x_1(1-cx_2) - g\frac{\cos(x_2)}{x_1}\right) + p_3 x_1 \sin(x_2) \\ + \frac{1}{m}p_1\hat{u}_1 + \frac{a_L c}{m}x_1 p_2 \hat{u}_2\Big\}. \end{aligned}$$

The optimal controls are given by

$$\hat{u}_1 = \begin{cases} T_{\min} & \text{if } p_1 < 0 \\ T_{\max} & \text{if } p_1 > 0 \end{cases} \quad \text{and} \quad \hat{u}_1 = \begin{cases} \theta_{\min} & \text{if } p_2 < 0 \\ \theta_{\max} & \text{if } p_2 > 0 \end{cases}$$

(recall that $x_1 > 0$). The singularities at $p_1 = 0$ and $p_2 = 0$ play very little role in the numerical computation and so will not be investigated further here; a more thorough treatment (for the 2 dimensional case with state x_1 and x_2) can be found in [110].

7.1.3 Numerical Results

The resulting optimal Hamiltonian was coded in a numerical tool developed at Stanford University [138, 139] for computing viscosity solutions to Hamilton-Jacobi equations using the algorithms of [140, 141]. The results are shown in Figures 7.2 and 7.3. The parameters used were $a_L = 65.3Kg/m$, $a_D = 3.18Kg/m$, $m = 160 \cdot 10^3 Kg$, $g = 9.81m/s^2$, $c = 6$, $\theta_{\min} = -20°$, $\theta_{\max} = 25°$, $T_{\min} = 60 \cdot 10^3 N$, and $T_{\max} = 120 \cdot 10^3 N$. They correspond to an Airbus A330 aircraft cruising at 35000 feet. The parameters used in the function l were $V_{\min} = 92m/s$, $V_{\max} = 170m/s$, $\gamma_{\min} = -20°$, $\gamma_{\max} = 25°$, $h_{\min} = -150m$, $h_{\max} = 150m$, $\delta V = 5m/s$, $\delta\gamma = 2.5°$, $\delta h = 10m$. The computation was performed on a $100 \times 100 \times 100$ grid and required 10298 seconds on a Pentium III, $800MHz$ processor running Red Hat Linux.

Figure 7.2 shows the level set $\text{Viab}(0, K) = \{x \in \mathbb{R}^3 \mid V_1(x, 0) \geq 0\}$ for two different values of the horizon, $T = 1.0s$ (left) and $T = 2.0s$ (right). As expected from Part **??** of Theorem 5.17, these sets are nested (the level set "shrinks" as T increases). For $T \approx 2.0s$ the shrinking stops; the level sets for

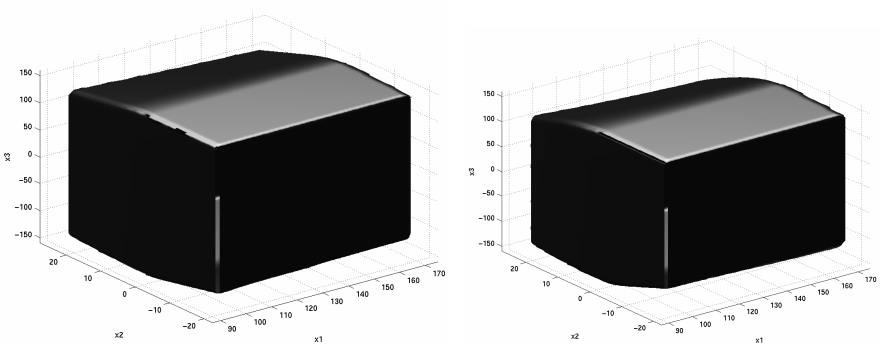

Figure 7.2. Two level sets of the value function $V_1(x, 0)$, for $T = 1s$ (left) and $T = 2s$ (right).

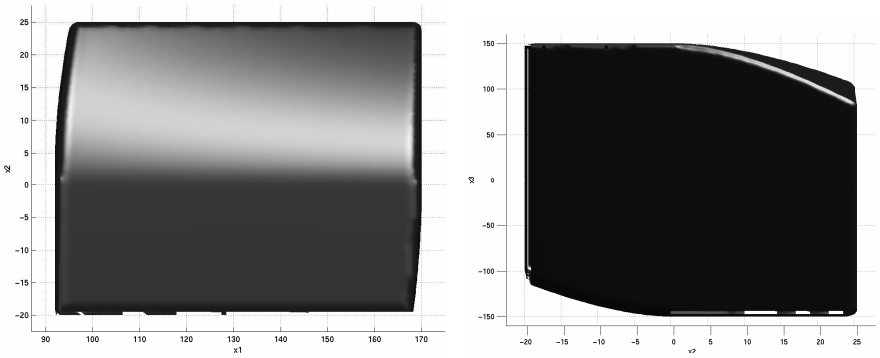

Figure 7.3. Projection of the $T = 2s$ level set along the x_3 axis (left) and along the x_1 axis (right).

values $T \geq 2$ are all the same. The general shape of the level sets suggests that certain states (e.g. combining high altitude with high flight path angle, low speed with high flight path angle etc.) are unsafe and should be avoided. If the aircraft ever gets to such a state, then, whatever the controller does from then on, it will sooner or later violate its flight envelope requirements. If the initial condition is inside the level set unsafe states can be avoided by applying the optimal controls of Section 7.1.2 whenever the state trajectory hits the boundary of the level set (see [142, 143] for practical problems associated with such a control strategy).

Better intuition about the unsafe states can be obtained if the level set for $T = 2.0s$ is projected along the axes (Figure 7.3). The projection along the x_2 axis leads to the square $[V_{min}, V_{max}] \times [h_{min}, h_{max}]$ in the x_1–x_3 coordinates. This suggests that any combination of speed and altitude within these bounds is safe for some value of flight path angle. The projection along the x_3 axis leads to the set shown on the left in Figure 7.3; the

shape of the set is in fact the same for all altitudes. Combinations of low speed with high flight path angle and high speed with low flight path angle are unsafe; the aircraft is bound to violate the speed restrictions for such combinations. The projection along the x_1 axis is shown on the right in Figure 7.3. Combinations of high altitude with high flight path angle and low altitude with low flight path angle are unsafe for all speeds; the aircraft is bound to violate the flight level limitations for such combinations. The situation gets worse as the speed increases.

7.2 Bibliography and Further Reading

d/dt, checkmate, Saint-Pierre, Asarin et.al.

Chapter 8

Stochastic Hybrid Systems (John)

Chapter 9
Stability of Hybrid Systems

For the purpose of studying stability of hybrid automata, we will again drop reference to inputs (and outputs) of the system and focus on the state trajectory. Later, we will bring the inputs (and outputs) back in when we talk about stabilizing controllers.

9.1 Review of Stability for Continuous Systems

For reference to the following material, see [7, **?**], Chapters 3 and 2 respectively.

Consider the following continuous system:

$$\dot{x} = f(x), \qquad x(0) = x_0 \tag{9.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$ is globally Lipschitz continuous.

- **Definition (Equilibrium of (9.1)):** $x = x_e$ is an *equilibrium point* of (9.1) if $f(x_e) = 0$. Without loss of generality in the following we will assume that $x_e = 0$.
- **Definition (Stability of (9.1)):** The equilibrium point $x_e = 0$ of (9.1) is *stable* (in the sense of Lyapunov) if for all $\epsilon > 0$ there exists a $\delta > 0$ such that

  $$||x_0|| < \delta \Rightarrow ||x(t)|| < \epsilon, \forall t \geq 0 \tag{9.2}$$

  where $x : [0, \infty) \to \mathbb{R}^n$ is the solution to (9.1), starting at x_0.

The equilibrium point $x_e = 0$ of (9.1) is *asymptotically stable* if it is stable and δ can be chosen so that

$$||x_0|| < \delta \Rightarrow \lim_{t\to\infty} ||x(t)|| = 0 \tag{9.3}$$

- More definitions for stability: exponentially stable, globally (asymptotically, exponentially) stable, locally (asymptotically, exponentially) stable, unstable ...
- Note that the definition of stability allows $x_e = 0$ to be stable without $x(t)$ converging to 0; note also that for system (9.1) with a single unstable equilibrium point, for all x_0 the solution can be bounded.

Consider a continuously differentiable (C^1) function $V : \mathbb{R}^n \to \mathbb{R}$. The rate of change of V along solutions of (9.1) is computed as:

$$\dot{V}(x) = \sum_{i=1}^{n} \frac{\partial V}{\partial x_i}\dot{x}_i = \sum_{i=1}^{n} \frac{\partial V}{\partial x_i} f_i(x) = \frac{\partial V}{\partial x} f(x) \tag{9.4}$$

This function is denoted the *Lie derivative* of V with respect to f.

Theorem 9.1 (Lyapunov Stability Theorem) *Let $x_e = 0$ be an equilibrium point of $\dot{x} = f(x), x(0) = x_0$ and $D \subset \mathbb{R}^n$ a set containing $x_e = 0$. If $V : D \to \mathbb{R}$ is a C^1 function such that*

$$V(0) = 0 \tag{9.5}$$
$$V(x) > 0, \forall x \in D\backslash\{0\} \tag{9.6}$$
$$\dot{V}(x) \leq 0, \forall x \in D \tag{9.7}$$

then x_e is stable. Furthermore, if $x_e = 0$ is stable and

$$\dot{V}(x) < 0, \forall x \in D\backslash\{0\} \tag{9.8}$$

then x_e is asymptotically stable.

Note that the Lyapunov function defines level sets $\{x \in \mathbb{R}^n : V(x) \leq c\}$ for $c > 0$ (see Figure 9.1). If a state trajectory enters one of these sets, it has to stay inside it, since $\dot{V}(x) \leq 0$ implies that if $V(x) = c$ at $t = t_0$, then $V(x(t)) \leq V(x(0)) \leq c$, $\forall t \geq t_0$.

Proof: For stability, we need to prove that for all $\epsilon > 0$ there exists $\delta > 0$ such that:

$$||x_0|| < \delta \Rightarrow ||x(t)|| < \epsilon, \forall t \geq 0 \tag{9.9}$$

We will use the following notation: for any $r > 0$, let $B_r = \{x \in \mathbb{R}^n : ||x|| < r\}$, $S_r = \{x \in \mathbb{R}^n : ||x|| = r\}$, and $\Omega_r = \{x \in \mathbb{R}^n : V(x) < r\}$.

See Figure 9.2. Choose $r_1 \in (0, \epsilon)$ such that $B_{r_1} \subseteq D$ (we do this because there is no guarantee that $B_\epsilon \subseteq D$). Let $c_1 = \min_{x\in S_{r_1}} V(x)$. Choose $c_2 \in (0, c_1)$. Note that there is no guarantee that $\Omega_{c_2} \subset B_{r_1}$. *Why not?* However, if $\delta > 0$ is chosen so that $B_\delta \subseteq \Omega_{c_2}$, then $V(x_0) < c_2$. Since V is non-increasing along system executions, executions that start inside

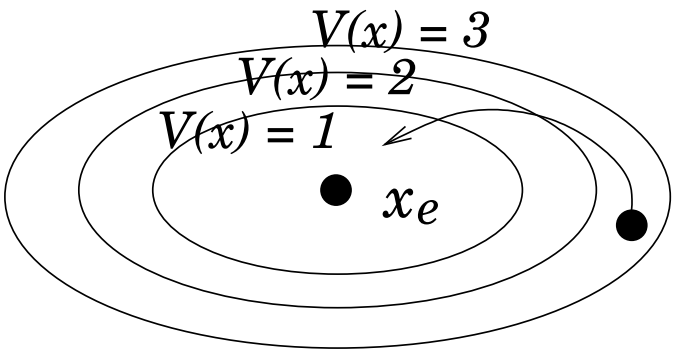

Figure 9.1. Level sets $V(x) = 1$, $V(x) = 2$, and $V(x) = 3$ for a Lyapunov function V; thus if a state trajectory enters one of these sets, it has to stay inside it since $\dot{V}(x) \leq 0$.

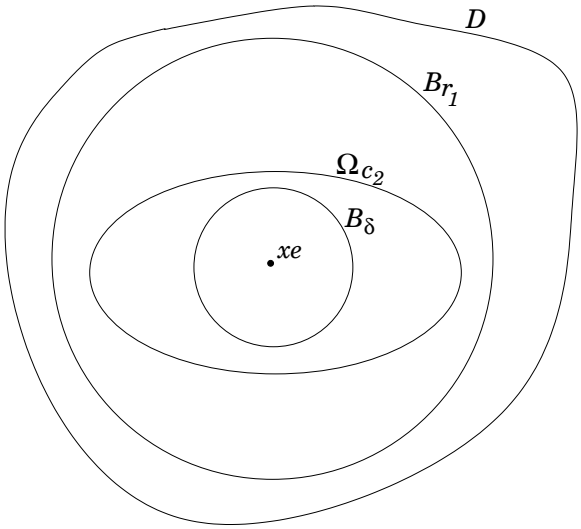

Figure 9.2. Figure for Proof of Lyapunov Stability Theorem (for continuous systems); WLOG $x_e = 0$.

B_δ cannot leave Ω_{c_2}. Thus for all $t > 0$ we have $x(t) \in B_{r_1} \subset B_\epsilon$. Thus $||x(t)|| \leq \epsilon$ for all $t > 0$. ∎

- **Example (Pendulum)** Consider the pendulum, unit mass, unit length, where x_1 is the angle of the pendulum with the vertical, and x_2 is the angular velocity of the pendulum.

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -g \sin x_1 \end{aligned}$$

To show that $x_e = 0$ (pendulum pointing downwards) is a stable equilibrium, consider the candidate Lyapunov function:

$$V(x) = g(1 - \cos x_1) + \frac{x_2^2}{2} \tag{9.10}$$

defined over the set $\{x \in \mathbb{R}^n : -\pi < x_1 < \pi\}$. Clearly, $V(0) = 0$, and $V(x) > 0, \forall x \in \{x \in \mathbb{R}^n : -\pi < x_1 < \pi\} \backslash \{0\}$. Also,

$$\dot{V}(x) = [g \sin x_1 \;\; x_2] \begin{bmatrix} x_2 \\ -g \sin x_1 \end{bmatrix} = 0$$

so the equilibrium point $x_e = 0$ is stable. Is it asymptotically stable?

- Finding Lyapunov functions in general is HARD. Often a solution is to try to use the *energy* of the system as a Lyapunov function (as in the example above). However, for linear systems, finding Lyapunov functions is easy: For a stable linear system $\dot{x} = Ax$, a Lyapunov function is given by $V(x) = x^T P x$, where P is a positive definite symmetric matrix which solves the *Lyapunov Equation* $A^T P + PA = -I$. (Recall that a matrix P is said to be positive definite if $x^T P x > 0$ for all $x \neq 0$. It is called positive semidefinite if $x^T P x \geq 0$ for all $x \neq 0$.)

9.2 Stability of Hybrid Systems

Consider an autonomous hybrid automaton $H = (S, \mathrm{I}nit, f, \mathrm{D}om, R)$.

Definition 9.2 (Equilibrium of a Hybrid Automaton) *The continuous state $x_e = 0 \in \mathbb{R}^n$ is an equilibrium point of H if:*

1. *$f(q, 0) = 0$ for all $q \in \mathbf{Q}$, and*
2. *$R(q, 0) \subset Q \times 0$.*

- Thus, discrete transitions are allowed out of $(q, 0)$, as long as the system jumps to a (q', x) in which $x = x_e = 0$.
- It follows from the above definition that if $(q_0, 0) \in \mathrm{I}nit$ and $(\tau, (q, x))$ represents the hybrid execution starting at $(q_0, 0)$, then $x(t) = 0$ for all $t \in \tau$.

As we did for continuous systems, we would like to characterize the notion that if the continuous state starts close to the equilibrium point, it stays close, or converges, to it.

- **Definition (Stable Equilibrium of a Hybrid Automaton):** Let $x_e = 0$ be an equilibrium point of the hybrid automaton H. Then $x_e = 0$ is stable if for all $\epsilon > 0$ there exists $\delta > 0$ such that for all

$(\tau, (q, x))$ starting at (q_0, x_0),

$$||x_0|| < \delta \Rightarrow ||x(t)|| < \epsilon, \forall t \in \tau \tag{9.11}$$

- **Definition (Asymptotically Stable Equilibrium of a Hybrid Automaton):** Let $x_e = 0 \in X$ be an equilibrium point of the hybrid automaton H. Then $x_e = 0$ is asymptotically stable if it is stable and δ can be chosen so that for all $(\tau, (q, x))$ starting at (q_0, x_0),

$$||x_0|| < \delta \Rightarrow \lim_{t \to \tau_\infty} ||x(t)|| = 0 \tag{9.12}$$

- **Remark:** In the above, $(\tau, (q, x))$ is considered to be an infinite execution, with $\tau_\infty = \sum_i (\tau_i' - \tau_i)$. Notice that $\tau_\infty < \infty$ if χ is Zeno and $\tau_\infty = \infty$ otherwise.

One would expect that a hybrid system for which each discrete state's continuous system is stable would be stable, at least if $R(q, x) \in Q \times \{x\}$ for all x. But this is NOT NECESSARILY the case:

- **Example:** Consider the hybrid automaton H with:
  - $Q = \{q_1, q_2\}$, $X = \mathbb{R}^2$
  - $Init = Q \times \{x \in X : ||x|| > 0\}$
  - $f(q_1, x) = A_1 x$ and $f(q_2, x) = A_2 x$, with:

$$A_1 = \begin{bmatrix} -1 & 10 \\ -100 & -1 \end{bmatrix}, A_2 = \begin{bmatrix} -1 & 100 \\ -10 & -1 \end{bmatrix}$$

  - $Dom = \{q_1, \{x \in \mathbb{R}^2 : x_1 x_2 \leq 0\}\} \cup \{q_2, \{x \in \mathbb{R}^2 : x_1 x_2 \geq 0\}\}$
  - $R(q_1, \{x \in \mathbb{R}^2 : x_1 x_2 \geq 0\}) = (q_2, x)$ and $R(q_2, \{x \in \mathbb{R}^2 : x_1 x_2 \leq 0\}) = (q_1, x)$

- **Remark 1:** Since $f(q_1, 0) = f(q_2, 0) = 0$ and $R(q_1, 0) = (q_2, 0), R(q_2, 0) = (q_1, 0)$, $x_e = 0$ is an equilibrium of H.

- **Remark 2:** Since the eigenvalues of both systems are at $-1 \pm j\sqrt{1000}$, the continuous systems $\dot{z} = A_i x$ for $i = 1, 2$ are asymptotically stable. (See phase portraits for each in Figure 9.3.)

- **Remark 3:** $x_e = 0$ is unstable for H! If the switching is flipped, then $x_e = 0$ is stable! (See phase portraits for each in Figure 9.4.)

- **Remark 4:** This simple example (drawn from [144]) shows that in general we cannot expect to analyze the stability of a hybrid system by studying the continuous components separately.

Theorem 9.3 (Lyapunov Stability Theorem (for hybrid systems)) *Consider a hybrid automaton H with $x_e = 0$ an equilibrium point, and $R(q, x) \in Q \times \{x\}$. Assume that there exists an open set $D \subset Q \times \mathbb{R}^n$ such that $(q, 0) \in D$ for some $q \in Q$. Let $V : D \to \mathbb{R}$ be a C^1 function in its second argument such that for all $q \in Q$:*

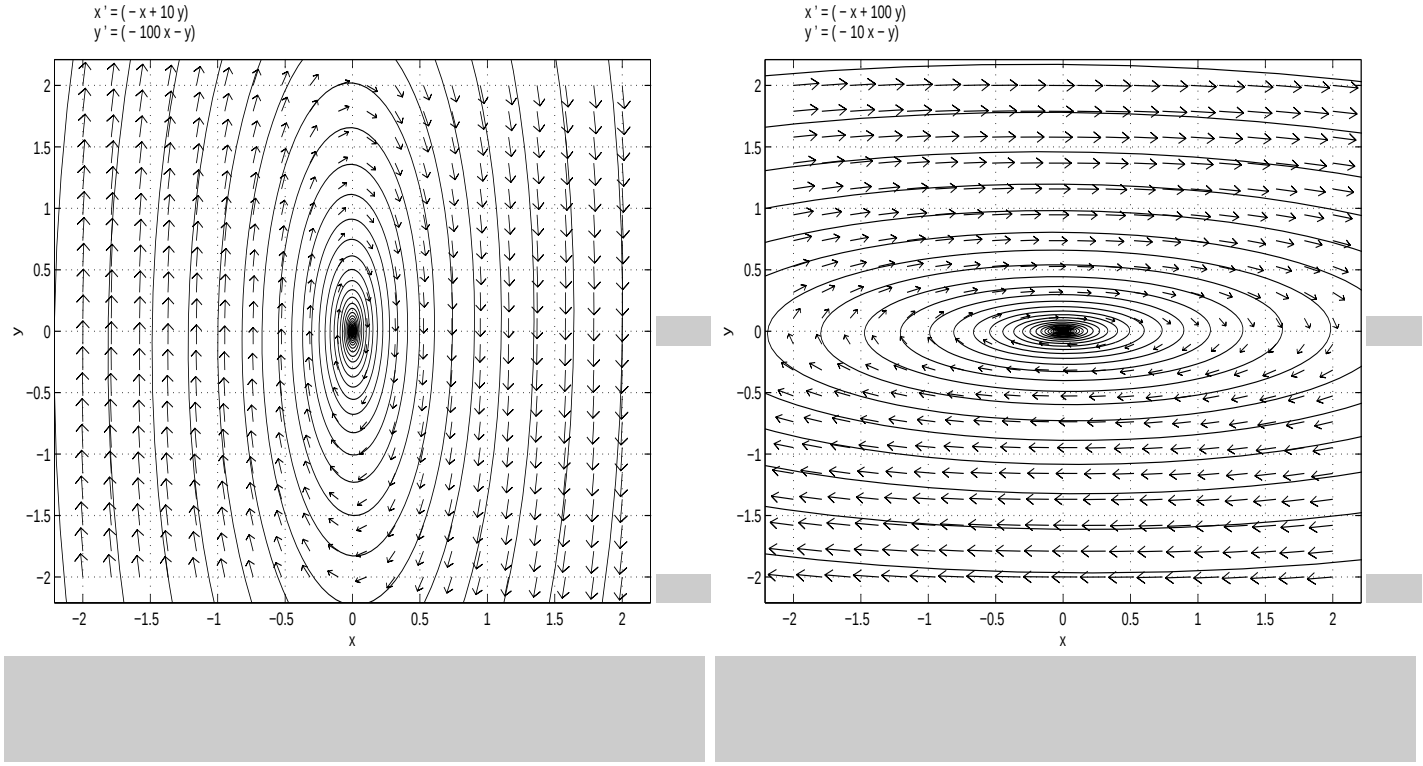

Figure 9.3. (a) Phase portrait of $\dot{x} = A_1 x$; (b) Phase portrait of $\dot{x} = A_2 x$. Figure generated using phase plane software from `http://math.rice.edu/polking/odesoft/`, freely downloadable.

1. $V(q,0) = 0$;
2. $V(q,x) > 0$ *for all* x, $(q,x) \in D \backslash \{0\}$, *and*
3. $\frac{\partial V(q,x)}{\partial x} f(q,x) \leq 0$ *for all* x, $(q,x) \in D$

If for all (τ, q, x) *starting at* (q_0, x_0) *where* $(q_0, x_0) \in \text{Init} \cap D$, *and all* $q' \in Q$, *the sequence* $\{V(q(\tau_i), x(\tau_i)) : q(\tau_i) = q'\}$ *is non-increasing (or empty), then* $x_e = 0$ *is a stable equilibrium of* H.

- We call such function "Lyapunov-like" (see Figure 9.5).
- A drawback of this Theorem is that the sequence $\{V(q(\tau_i), x(\tau_i))\}$ must be evaluated (which may require integrating the vector field and then we lose the fundamental advantage of Lyapunov theory)
- Also, it is in general difficult to derive such a function V

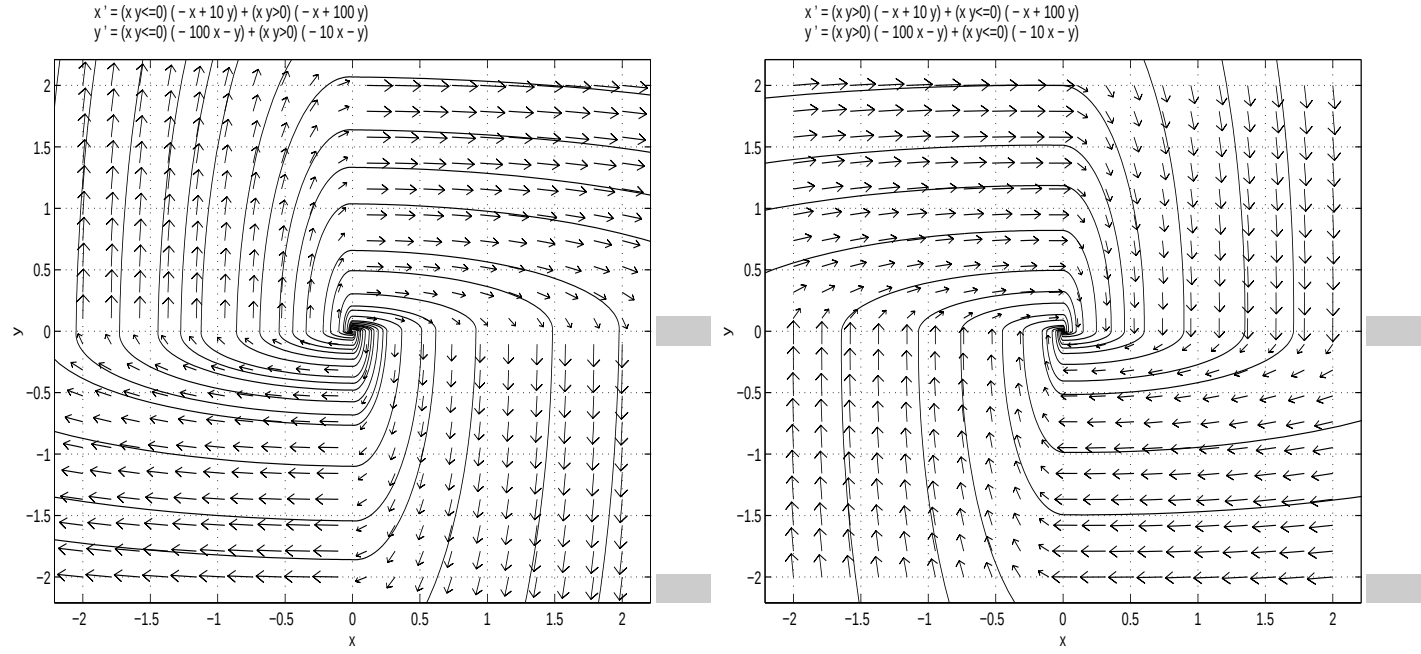

Figure 9.4. (a) Phase portrait of H; (b) Phase portrait of H, switching conditions flipped.

- HOWEVER, for certain classes of hybrid automata, which have vector fields linear in x, computationally attractive methods exist to derive Lyapunov-like functions V

Proof: We sketch the proof for $Q = \{q_1, q_2\}$ and $(q, \cdot) \notin R(q, \cdot)$. Define the sets in Figure 9.6 similar to the previous proof, ie.

$$\Omega_{c_{2_i}} = \{x \in B_{r_{1_i}} : V(q_i, x) < c_{2_i}\} \tag{9.13}$$

where $c_{2_i} \in (0, c_{1_i})$ where $c_{1_i} = \min_{x \in S_{r_{1_i}}} V(q_i, x)$. Now let $r = \min\{\delta_1, \delta_2\}$, and inside B_r, in each of q_1 and q_2, repeat the construction above, ie.

$$\Omega_{\overline{c}_{2_i}} = \{x \in B_r : V(q_i, x) < \overline{c}_{2_i}\} \tag{9.14}$$

where $\overline{c}_{2_i} \in (0, \min_{x \in S_r} V(q_i, x))$. Also, let $B_{\overline{\delta}_i} \subset \Omega_{\overline{c}_{2_i}}$. Let $\delta = \min\{\overline{\delta}_1, \overline{\delta}_2\}$. Consider the hybrid trajectory (τ, q, x) starting at $x_0 \in B_\delta$, and assume that the initial discrete state q_0 is equal to q_1. By the corresponding continuous Lyapunov theorem, $x(t) \in \Omega_{\overline{c}_{2_1}}$ for $t \in [\tau_0, \tau_0']$. Therefore,

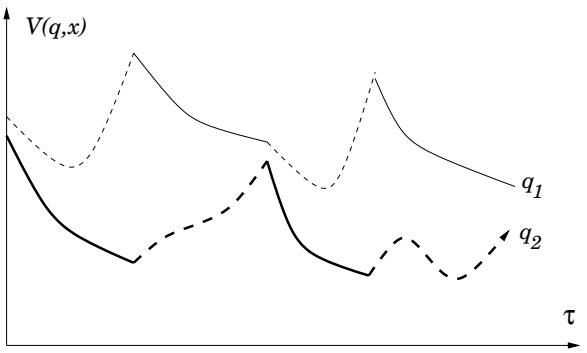

Figure 9.5. Showing $V(q_1, x)$ and $V(q_2, x)$. Solid segments on q_i mean that the system is in q_i at that time, dotted segments mean the system is in q_j, $j \neq i$.

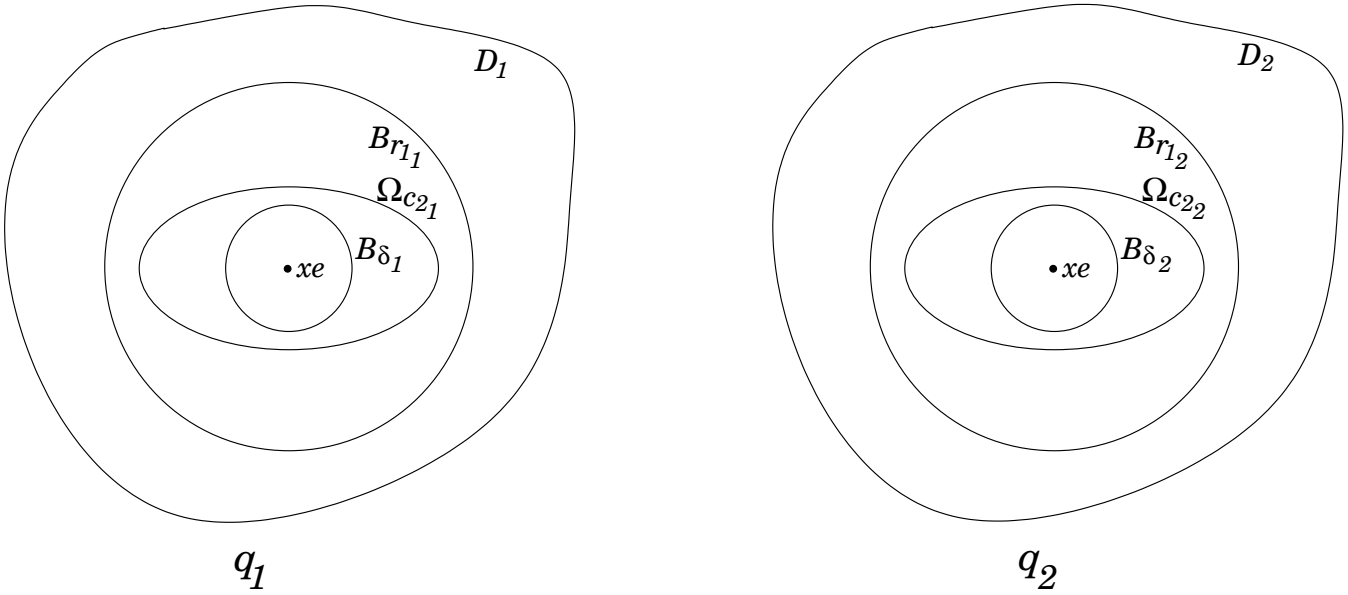

Figure 9.6. Figure for Proof of Lyapunov Stability Theorem (for hybrid systems).

$x(\tau_1) = x(\tau_0') \in \Omega_{\overline{c}_{2_2}}$ (where equality holds because of the restricted definition of the transition map). By the continuous Lyapunov Theorem again, $x(t) \in \Omega_{\overline{c}_{2_2}}$ and thus $x(t) \in B_\epsilon$ for $t \in [\tau_1, \tau_1']$. By the assumption of the non-increasing sequence, $x(\tau_1') = x(\tau_2) \in \Omega_{\overline{c}_{2_1}}$. The result follows by induction. ■

Corollary 9.4 (A more restrictive Lyapunov Stability Theorem (for hybrid systems)) *Consider a hybrid automaton H with $x_e = 0$ an equilibrium point, and*

$R(q,x) \in Q \times \{x\}$. Assume that there exists an open set $D \subset \mathbb{R}^n$ such that $0 \in D$. Let $V : D \to \mathbb{R}$ be a C^1 function such that for all $q \in Q$:

1. *$V(0) = 0$;*
2. *$V(x) > 0$ for all $x \in D \backslash \{0\}$, and*
3. *$\frac{\partial V(x)}{\partial x} f(q,x) \leq 0$ for all $x \in D$*

Then $x_e = 0$ is a stable equilibrium of H.

Proof: Define $\hat{V} : \mathbf{Q} \times \mathbb{R}^n \to \mathbb{R}$ by $\hat{V}(q,x) = V(x)$ for all $q \in \mathbf{Q}$, $x \in \mathbb{R}^n$ and apply Theorem 3. ∎

9.3 Lyapunov Stability for Piecewise Linear Systems

Theorem 9.5 (Lyapunov Stability for Linear Systems) *The equilibrium point $x_e = 0$ of $\dot{x} = Ax$ is asymptotically stable if and only if for all matrices $Q = Q^T > 0$ there exists a unique matrix $P = P^T > 0$ such that*

$$PA + A^T P = -Q \tag{9.15}$$

Proof: For the "if" part of the proof, consider the Lyapunov function (from Lecture Notes 4) $V(x) = x^T P x$. Thus,

$$\dot{V} = x^T P \dot{x} + \dot{x}^T P x = x^T (PA + A^T P) x) = -x^T Q x < 0 \tag{9.16}$$

For the "only if" part of the proof, consider

$$P = \int_0^\infty e^{A^T t} Q e^{At} dt \tag{9.17}$$

which is well-defined since $\mathrm{Re}(\lambda_i(A)) < 0$. Clearly,

$$\begin{aligned} PA + A^T P &= \int_0^\infty e^{A^T t} Q e^{At} A dt + \int_0^\infty A^T e^{A^T t} Q e^{At} dt & (9.18) \\ &= \int_0^\infty \frac{d}{dt} e^{A^T t} Q e^{At} dt = -Q & (9.19) \end{aligned}$$

Also, P is unique: to prove this, assume there exists another solution $\hat{P} \neq P$. Then,

$$\begin{aligned} 0 &= e^{A^T t}(Q - Q)e^{At} & (9.20) \\ &= e^{A^T t}[(P - \hat{P})A + A^T(P - \hat{P})]e^{At} & (9.21) \\ &= \frac{d}{dt} e^{A^T t}(P - \hat{P})e^{At} & (9.22) \end{aligned}$$

which means that $e^{A^T t}(P - \hat{P})e^{At}$ is constant for all $t \geq 0$. Thus,

$$e^{A^T 0}(P - \hat{P})e^{A0} = \lim_{t\to\infty} e^{A^T t}(P - \hat{P})e^{At} \tag{9.23}$$

thus $P = \hat{P}$, which contradicts the assumption and thus concludes the proof. ■

- Equation (9.15) is called a Lyapunov equation. We may also write the matrix condition in the Theorem of Lyapunov Stability for Linear Systems as the following inequality:

$$A^T P + PA < 0 \tag{9.24}$$

which is called a *linear matrix inequality* (LMI), since the left hand side is linear in the unknown P.

Example: Switched Linear System, Revisited.
Consider the linear hybrid system example from page 5 (with the switching flipped):

- $Q = \{q_1, q_2\}$, $X = \mathbb{R}^2$
- I$nit = Q \times \{x \in X : ||x|| > 0\}$
- $f(q_1, x) = A_1 x$ and $f(q_2, x) = A_2 x$, with:

$$A_1 = \begin{bmatrix} -1 & 10 \\ -100 & -1 \end{bmatrix}, A_2 = \begin{bmatrix} -1 & 100 \\ -10 & -1 \end{bmatrix}$$

- D$om = \{q_1, \{x \in \mathbb{R}^2 : x_1 x_2 \geq 0\}\} \cup \{q_2, \{x \in \mathbb{R}^2 : x_1 x_2 \leq 0\}\}$
- $R(q_1, \{x \in \mathbb{R}^2 : x_1 x_2 \leq 0\}) = (q_2, x)$ and $R(q_2, \{x \in \mathbb{R}^2 : x_1 x_2 \geq 0\}) = (q_1, x)$

Proposition 9.6 *$x = 0$ is an equilibrium of H.*

Proof: $f(q_1, 0) = f(q_2, 0) = 0$ and $R(q_1, 0) = (q_2, 0), R(q_2, 0) = (q_1, 0)$. ■

Proposition 9.7 *The continuous systems $\dot{x} = A_i x$ for $i = 1, 2$ are asymptotically stable.*

Recall that $P_i > 0$ (positive definite) if and only if $x^T P_i x > 0$ for all $x \neq 0$, and I is the identity matrix.

Consider the candidate Lyapunov function:

$$V(q, x) = \begin{cases} x^T P_1 x & \text{if } q = q_1 \\ x^T P_2 x & \text{if } q = q_2 \end{cases}$$

Check that the conditions of the Theorem hold. For all $q \in \mathbf{Q}$:

1. $V(q, 0) = 0$
2. $V(q, x) > 0$ for all $x \neq 0$ (since the P_i are positive definite)

3. $\frac{\partial V}{\partial x}(q,x)f(q,x) \leq 0$ for all x since:

$$\begin{aligned}\frac{\partial V}{\partial x}(q,x)f(q,x) &= \frac{d}{dt}V(q,x(t)) \\ &= \dot{x}^T P_i x + x^T P_i \dot{x} \\ &= x^T A_i^T P_i x + x^T P_i A_i x \\ &= x^T (A_i^T P_i + P_i A_i) x \\ &= -x^T I x \\ &= -\|x\|^2 \leq 0\end{aligned}$$

It remains to test the non-increasing sequence condition. Notice that the level sets of $x^T P_i x$ are ellipses centered at the origin. Therefore each level set intersects the switching line $x_i = 0$ (for $i = 1, 2$) at exactly two points, $\hat{x}$ and $-\hat{x}$. Assume that $x(\tau_i) = \hat{x}$ and $q(\tau_i) = q_1$. The fact that $V(q_1, x(t))$ does not increase for $t \in [\tau_i, \tau_i']$ (where $q(t) = q_1$) implies that the next time a switching line is reached, $x(\tau_i') = \alpha(-\hat{x})$ for some $\alpha \in (0, 1]$. Therefore, $\|x(\tau_{i+1})\| = \|x(\tau_i')\| \leq \|x(\tau_i)\|$. By a similar argument, $\|x(\tau_{i+2})\| = \|x(\tau_{i+1}')\| \leq \|x(\tau_{i+1})\|$. Therefore, $V(q(\tau_i), x(\tau_i)) \leq V(q(\tau_{i+2}), x(\tau_{i+2}))$.

9.3.1 Globally Quadratic Lyapunov Function

The material in this section and the next is drawn from the work of Mikael Johansson [73, **?**], and also from [**?**, 70, **?**].

Theorem 9.8 (Globally Quadratic Lyapunov Function) *Consider a hybrid automaton $H = (S, \mathrm{I}nit, f, \mathrm{D}om, R)$ with equilibrium $x_e = 0$. Assume that for all i:*

- $f(q_i, x) = A_i x, A_i \in \mathbb{R}^{n \times n}$
- $\mathrm{I}nit \subseteq \mathrm{D}om$
- *for all $x \in \mathbb{R}^n$*

$$|R(q_i, x)| = \begin{cases} 1 & \text{if } (q_i, x) \in \partial \mathrm{D}om \\ 0 & \text{otherwise} \end{cases} \tag{9.25}$$

and $\{(q', x') \in R(q_i, x)\} \Rightarrow \{(q', x') \in \mathrm{D}om, x' = x\}$. Furthermore, assume that for all $\chi \in \mathcal{E}_H^\infty$, $\tau_\infty(\chi) = \infty$. Then, if there exists $P = P^T > 0$ such that

$$A_i^T P + P A_i < 0, \forall i \tag{9.26}$$

$x_e = 0$ is asymptotically stable.

Proof: First note that there exists $\gamma > 0$ such that

$$A_i^T P + P A_i + \gamma I \leq 0, \forall i \tag{9.27}$$

Also, note that with the given assumptions there exists a unique, infinite, and non-Zeno execution $\chi = (\tau, q, x)$ for every $(q_0, x_0) \in \mathrm{I}nit$. For all $i \geq 0$,

the continuous evolution $x : \tau \to \mathbb{R}^n$ of such an execution satisfies the following time-varying linear differential equation:

$$\dot{x}(t) = \sum_i \mu_i(t) A_i x(t), t \in [\tau_i, \tau_i'] \tag{9.28}$$

where $\mu_i : \tau \to [0,1]$ is a function such that for $t \in [\tau_i, \tau_i']$, $\sum_i \mu_i(t) = 1$. Letting $V(q,x) = x^T P x$, we have that for $t \in [\tau_i, \tau_i']$,

$$\begin{aligned} \dot{V}(q(t), x(t)) &= \sum_i [\mu_i(t) x(t)^T (A_i^T P + P A_i) x(t)] && (9.29) \\ &\leq -\gamma ||x(t)||^2 \sum_i \mu_i(t) && (9.30) \\ &= -\gamma ||x(t)||^2 && (9.31) \end{aligned}$$

Now, since $V(q,x) = x^T P x$, we have that

$$\lambda_{min} ||x||^2 \leq V(q,x) \leq \lambda_{max} ||x||^2 \tag{9.32}$$

where $0 < \lambda_{min} \leq \lambda_{max}$ are the smallest and largest eigenvalues of P respectively. It follows that

$$\dot{V}(q(t), x(t)) \leq -\frac{\gamma}{\lambda_{max}} V(q(t), x(t)), t \in [\tau_i, \tau_i'] \tag{9.33}$$

and hence

$$V(q(t), x(t)) \leq V(q(\tau_i), x(\tau_i)) e^{-\gamma(t-\tau_i)/\lambda_{max}}, t \in [\tau_i, \tau_i'] \tag{9.34}$$

Thus, from (9.32):

$$\lambda_{min} ||x(t)||^2 \leq \lambda_{max} ||x(\tau_i)||^2 e^{-\gamma(t-\tau_i)/\lambda_{max}}, t \in [\tau_i, \tau_i'] \tag{9.35}$$

Since the execution χ by assumption is non-Zeno, we have that $\tau_i \to \infty$ as $i \to \infty$. Hence, $||x(t)||$ goes to zero exponentially as $t \to \tau_\infty$, which implies that the equilibrium point $x_e = 0$ is asymptotically (actually exponentially) stable. ■

Example 1: Consider the hybrid automaton H of Figure 9.7, with

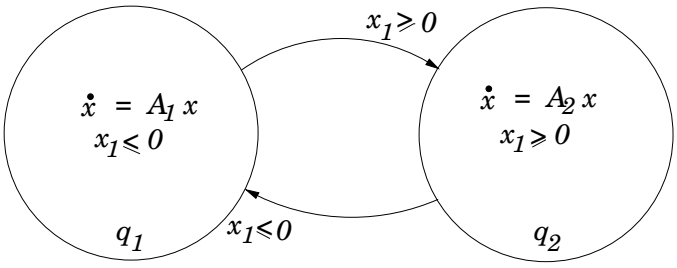

Figure 9.7. Example 1

$$A_1 = \begin{bmatrix} -1 & 1 \\ -1 & -1 \end{bmatrix}, A_2 = \begin{bmatrix} -2 & 1 \\ -1 & -2 \end{bmatrix} \tag{9.36}$$

Since the eigenvalues of A_1 are $\lambda(A_1) = \{-1 \pm i\}$ and of A_2 are $\lambda(A_2) = \{-2 \pm i\}$, both $\dot{x} = A_1 x$ and $\dot{x} = A_2 x$ have an asymptotically stable focus. H satisfies the assumptions of the previous theorem; indeed, $A_1^T + A_1 < 0$ and $A_2^T + A_2 < 0$, so the inequalities in the theorem are satisfied for $P = I$. Hence, the origin is an asymptotically stable equilibrium point for H.

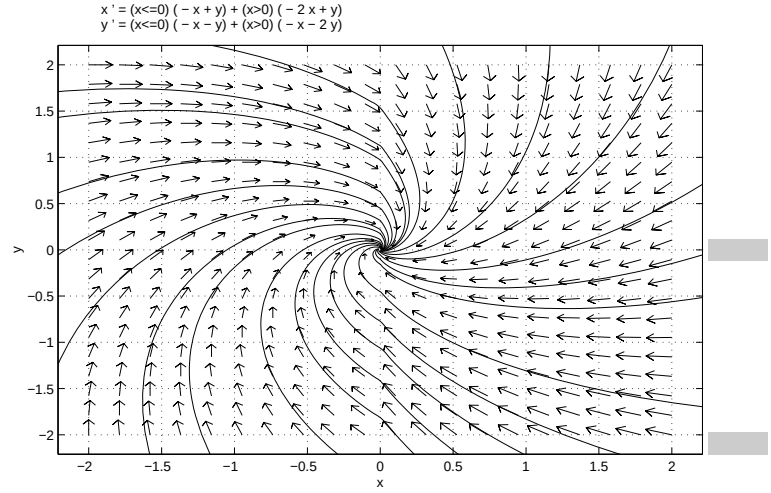

Figure 9.8. Example 1

Example 2 [?]: Consider the hybrid automaton from Figure 1 again, but let

$$A_1 = \begin{bmatrix} -5 & -4 \\ -1 & -2 \end{bmatrix}, A_2 = \begin{bmatrix} -2 & -4 \\ 20 & -2 \end{bmatrix} \tag{9.37}$$

The eigenvalues of A_1 are $\lambda(A_1) = \{-6, -1\}$ and of A_2 are $\lambda(A_2) = \{-2 \pm 4\sqrt{5}i\}$ so that $\dot{x} = A_1 x$ has an asymptotically stable node and $\dot{x} = A_2 x$ has an asymptotically stable focus. The evolution of the continuous state is shown in Figure 9.9 for four different initial states. The origin seems to be a stable equilibrium point – indeed, the Lyapunov function indicated by the dashed level sets proves asymptotic stability of the origin. Yet from the shape of its level sets, we see that the Lyapunov function is not globally quadratic, but piecewise quadratic, in the sense that it is quadratic in each discrete mode. (In fact, we can show for this example that it is not possible to find a quadratic Lyapunov function).

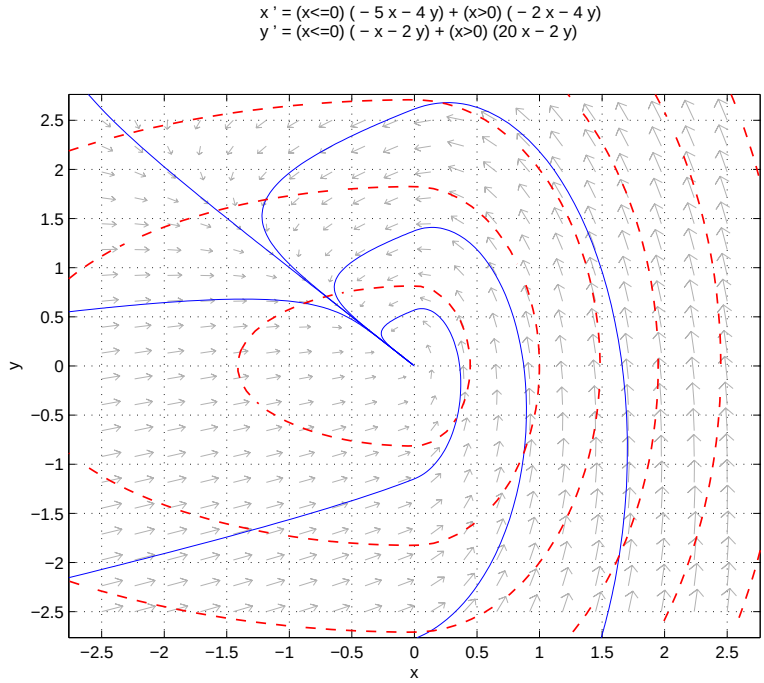

Figure 9.9. Continuous evolution for a hybrid automaton that does not have a globally quadratic Lyapunov function. Still, the origin is an asymptotically stable equilibrium point, which can be proved by using a Lyapunov function quadratic in each discrete state.

9.3.2 Piecewise Quadratic Lyapunov Function

If we assume that the hybrid automaton is restricted further so that the domains are given by polyhedra, then we can make some more general statements about the stability of the hybrid system:

$$\mathrm{D}om = \cup_i \{q_i\} \times \{x \in \mathbb{R}^n : E_{i1}x \geq 0, \ldots, E_{in}x \geq 0\} \tag{9.38}$$

where

$$E_i = \begin{bmatrix} E_{i1} \\ \vdots \\ E_{in} \end{bmatrix} \in \mathbb{R}^{n \times n} \tag{9.39}$$

It then follows that $(q_i, 0) \in \mathrm{D}om$ for all i. Suppose that the reset relation R satisfies:

$$|R(q_i, x)| = \begin{cases} 1 & \text{if } (q_i, x) \in \partial \mathrm{D}om \\ 0 & \text{otherwise} \end{cases} \tag{9.40}$$

such that

$$(q_k, x') \in R(q_i, x) \Rightarrow F_k x = F_i x, q_k \neq q_i, x' = x \tag{9.41}$$

where $F_k, F_i \in \mathbb{R}^{m \times n}$ are given matrices (which hence define the boundaries of D*om*). The LMI condition in Theorem 1 would require that

$$x^T(A_i^T P + P A_i)x < 0, \forall x \neq 0, (q_i, x) \in Q \times \mathbb{R}^n \tag{9.42}$$

It is however sufficient to require that

$$x^T(A_i^T P + P A_i)x < 0, \forall x \neq 0, (q_i, x) \in \mathrm{D}om \tag{9.43}$$

This can be done by specifying a matrix S_i such that $x^T S_i x \geq 0$ for all x with $(q_i, x) \in \mathrm{D}om$. Then,

$$A_i^T P + P A_i + S_i < 0 \tag{9.44}$$

still implies that

$$x^T(A_i^T P + P A_i)x < 0, \forall x \neq 0, (q_i, x) \in \mathrm{D}om \tag{9.45}$$

but $x^T(A_i^T P + P A_i)x < 0$ does not have to hold for $x \neq 0$ with $(q_k, x) \in \mathrm{Inv}$ and $i \neq k$. The matrix S_i may be given as $S_i = E_i^T U_i E_i$, where E_i is given by the representation of H and $U_i = U_i^T \in \mathbb{R}^{n \times n}$ is chosen to have non-negative elements.

We can also let V depend on the discrete state, ie. $V(q_i, x) = x^T P_i x$ for $(q_i, x) \in \mathrm{Inv}$. We choose $P_i = F_i^T M F_i$, where F_i is given by the representation of H, and $M = M^T \in \mathbb{R}^{n \times n}$ is to be chosen.

Theorem 9.9 (Piecewise Quadratic Lyapunov Function) *$H = (S, \mathrm{I}nit, f, \mathrm{D}om, R)$ with equilibrium $x_e = 0$. Assume that for all i:*

- $f(q_i, x) = A_i x, A_i \in \mathbb{R}^{n \times n}$
- $\mathrm{D}om = \cup_i \{q_i\} \times \{x \in \mathbb{R}^n : E_{i1} x \geq 0, \ldots, E_{in} x \geq 0\}$
- $\mathrm{I}nit \subseteq \mathrm{D}om$
- *for all $x \in \mathbb{R}^n$*

$$|R(q_i, x)| = \begin{cases} 1 & \text{if } (q_i, x) \in \partial \mathrm{D}om \\ 0 & \text{otherwise} \end{cases} \tag{9.46}$$

such that

$$(q_k, x') \in R(q_i, x) \Rightarrow F_k x = F_i x, q_k \neq q_i, x' = x \tag{9.47}$$

where $F_k, F_i \in \mathbb{R}^{n \times n}$.

Furthermore, assume that for all $\chi \in \mathcal{E}_H^\infty$, $\tau_\infty(\chi) = \infty$. Then, if there exists $U_i = U_i^T$, $W_i = W_i^T$, and $M = M^T$ such that $P_i = F_i^T M F_i$ satisfies:

$$A_i^T P_i + P_i A_i + E_i^T U_i E_i \quad < \quad 0 \tag{9.48}$$

$$P_i - E_i^T W_i E_i \quad > \quad 0 \tag{9.49}$$

where U_i, W_i are non-negative, then $x_e = 0$ is asymptotically stable.

Example 3: Consider the hybrid automaton of Figure 9.10 with

$$A_1 = A_3 = \begin{bmatrix} -0.1 & 1 \\ -5 & -0.1 \end{bmatrix}, A_2 = A_4 = \begin{bmatrix} -0.1 & 5 \\ -1 & -0.1 \end{bmatrix} \tag{9.50}$$

Here, we may choose

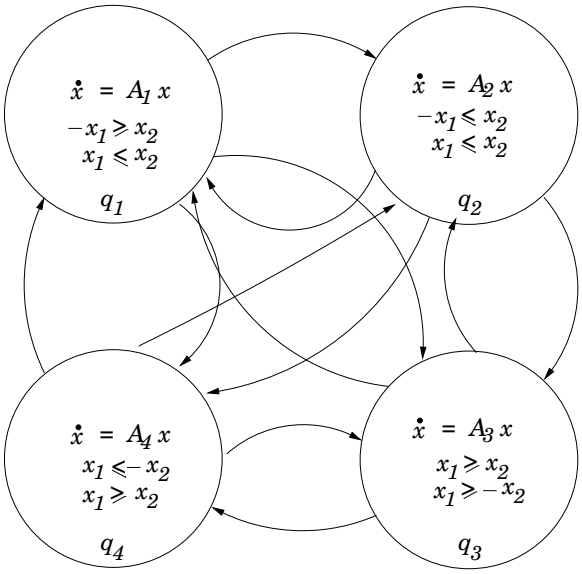

Figure 9.10. Example 3

$$E_1 = -E_3 = \begin{bmatrix} -1 & 1 \\ -1 & -1 \end{bmatrix}, E_2 = -E_4 = \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} \tag{9.51}$$

and

$$F_i = \begin{bmatrix} E_i \\ I \end{bmatrix} \forall i \in \{1, 2, 3, 4\} \tag{9.52}$$

The eigenvalues of A_i are $-1/10 \pm \sqrt{5}i$. The evolution of the continuous state is shown in Figure 9.11. We can use a Lyapunov function given by:

$$P_1 = P_3 = \begin{bmatrix} 5 & 0 \\ 0 & 1 \end{bmatrix}, P_2 = P_4 = \begin{bmatrix} 1 & 0 \\ 0 & 5 \end{bmatrix} \tag{9.53}$$

to prove asymptotic stability of the hybrid automaton.

9.3.3 Linear Matrix Inequalities

LMIs appear in many problems in systems and control theory (for example, see the reference [**?**]). For example, in the last Theorem we would like to

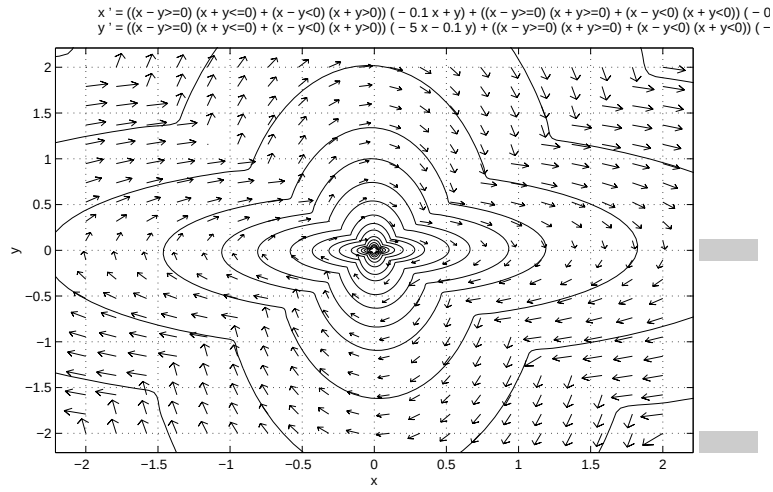

Figure 9.11. Example 3

solve

$$E_i^T M E_i \succ 0 \tag{9.54}$$

$$A_i^T P_i + P_i A_i + E_i^T U_i E_i \prec 0 \tag{9.55}$$

$$P_i - E_i^T W_i E_i \succ 0 \tag{9.56}$$

for the unknowns M, U_i, and W_i. This problem may be cast as an optimization problem, which turns out to be convex, so that it can be efficiently solved. In MATLAB, there is a toolbox called LMI Control Toolbox:

```
>> help lmilab
```

Try the demo:

```
>> help lmidem
```

and you'll see there is a graphical user interface to specify LMIs; you can enter this directly through:

```
lmiedit
```

After specifying an LMI, ie.:

$$P = P^T \succ 0 \tag{9.57}$$

$$A^T P + PA \prec 0 \tag{9.58}$$

where A is a matrix that you've already entered in MATLAB's workspace in `lmisys`, a feasible solution P is found (if it exists) by running the commands:

```
[tmin, pfeas] = feasp(lmisys)
p = dec2mat(lmisys,pfeas,p)
```

The feasibility problem is solved as a convex optimization problem, which has a global minimum. This means that if the feasibility problem has a solution, the optimization algorithm will (at least theoretically) always find it.

Chapter 10

Automated Highway Systems (John)

Chapter 11

Air Traffic Management and avionics (Claire/John)

Chapter 12

Biochemical networks (Claire/John)

Chapter 13

Research Directions and New Vistas

Appendix A

Preliminaries on Continuous and Discrete Systems

We start by providing a brief overview of some material on purely discrete and purely continuous systems that will be used in subsequent chapters. The coverage is by necessity concise. There are however many excellent textbooks that cover this material very thoroughly and in great detail. Pointers to some of the best known ones are given in the concluding section of the chapter.

A.1 Notation

We start by summarizing some fairly standard mathematical notation that will be used throughout the book.

- $\mathbb{R}^n$ denotes the n-dimensional Euclidean space. This is a finite dimensional **vector space** (also known as a linear space). If $n = 1$, we will drop the superscript and write just $\mathbb{R}$ (the set of real numbers or "the real line"). $\mathbb{R}_+$ will be used to denote the closed half space $\mathbb{R}_+ = \{x \in \mathbb{R} \mid x \geq 0\}$. We will make no distinction between vectors and real numbers in the notation (no arrows over the letters, bold fond, etc.). Both vectors and real numbers will be denoted by lower case letters.
- $|x| = \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}$ denotes the standard (Euclidean) **norm** in $\mathbb{R}^n$.
- $\mathbb{Z}$ denotes the set of integers, $\ldots, -2, -1, 0, 1, 2, \ldots$.

- $x \in A$ is a shorthand for "x belongs to a set A", e.g. $x \in \mathbb{R}^n$ means that x is an n-dimensional vector.
- Given a set X, 2^X denotes the **power set** of X, i.e. the set of all subsets of X. In other words, $A \in 2^X$ means that $A \subseteq X$. By definition, $X \in 2^X$ for all sets X.
- $\emptyset$ denotes the **empty set** (a set containing nothing). By definition $\emptyset \in 2^X$ for all sets X.

  Exercise A.1 *Consider a set containing only 3 elements, say* $Q = \{q_1, q_2, q_3\}$. *Write down all the elements of* 2^Q. *There should be* 8 *of them. Can you guess why* 2^X *is used to denote the power set of* X*?*
- $f(\cdot) : A \to B$ is a shorthand for a function mapping every element $x \in A$ to an element $f(x) \in B$. For example the function $\sin(\cdot) : \mathbb{R} \to \mathbb{R}$ maps a real number x to its sine, $\sin(x)$.
- In logic formulas
  - $\forall$ is a shorthand for "for all", as in "$\forall x \in \mathbb{R}, x^2 \geq 0$".
  - $\exists$ is a shorthand for "there exists", as in "$\exists x \in \mathbb{R}$ such that $\sin(x) = 0$".
  - $\wedge$ is a shorthand for "and", $\vee$ stands for "or", and $\neg$ stands for "not".
  - $\Rightarrow$ is a shorthand for "implies".
- Logic expressions can be used to define sets by listing properties of their elements. For example, the following expression defines a subset of $\mathbb{R}^2$

  $$\{x \in \mathbb{R}^2 \mid (x_1^2 + x_2^2 = 1) \wedge (x_1 \geq 0) \wedge (x_2 \leq 0)\},$$

  namely the part of the unit circle that falls in the 4^{th} quadrant.
- ∞ denotes "infinity".
- Given two real numbers $a \leq b$,

  $$[a, b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}$$

  denotes the **closed interval** from a to b, while

  $$[a, b) = \{x \in \mathbb{R} \mid a \leq x < b\}$$

  denotes the **right-open** interval from a to b. Notice that if $a = b$, then $[a, b] = [a, a] = \{a\}$, whereas $[a, b) = [a, a) = \emptyset$. If $a < b$, $[a, b] = [a, b) = \emptyset$. We also define $[a, \infty)$ as the set of all real numbers greater than or equal to a. Clearly, $\mathbb{R}_+ = [0, \infty)$.
- Given two sets Q and X, $Q \times X$ denotes the **product** of the two sets. This is the set of all ordered pairs (q, x) with $q \in Q$ and $x \in X$, i.e.

  $$Q \times X = \{(q, x) \mid q \in Q \text{ and } x \in X\}.$$

Notice that $\mathbb{R}^2 = \mathbb{R} \times \mathbb{R}$ and, more generally, $\mathbb{R}^n = \mathbb{R} \times \mathbb{R} \times \ldots \times \mathbb{R}$. Elements of $\mathbb{R}^n$ will therefore be denoted interchangably as standard column vectors

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

or as ordered $n-$tuples, $x = (x_1, x_2, \ldots, x_n)$.

The book assumes some familiarity with the concepts of vector space, state space, differential equations, etc. A brief review of some of these topics will be given below.

A.2 Review of Continuous Systems

All the continuous nonlinear systems considered in this book can be reduced to the standard **state space** form. It is usual to denote

- the **states** of the system by $x_i \in \mathbb{R}$, $i = 1, \ldots, n$,
- the **inputs** by $u_j \in \mathbb{R}$, $j = 1, \ldots, m$, and
- the **outputs** by $y_k \in \mathbb{R}$, $k = 1, \ldots, p$.

The number of states, n, is called the **dimension** (or **order**) of the system. The evolution of the states, inputs and outputs is governed by a set of functions

$$f_i : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R} \to \mathbb{R}, \text{ for } i = 1, \ldots, n$$
$$h_j : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R} \to \mathbb{R}, \text{ for } j = 1, \ldots, p$$

Roughly speaking, at a given time $t \in \mathbb{R}$ and for given values of all the states and inputs these functions determine in what direction the state will move, and what the output is going to be.

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, \ldots, x_n, u_1, \ldots, u_m, t) \\ &\vdots \\ \dot{x}_n &= f_n(x_1, \ldots, x_n, u_1, \ldots, u_m, t) \\ y_1 &= h_1(x_1, \ldots, x_n, u_1, \ldots, u_m, t) \\ &\vdots \\ y_p &= h_p(x_1, \ldots, x_n, u_1, \ldots, u_m, t) \end{aligned}$$

Exercise A.2 *What is the dimension of the pendulum example of Section 2.2.1? What are the functions f_i?*

It is usually convenient to simplify the equations somewhat by introducing vector notation. Let

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n,\ u = \begin{bmatrix} u_1 \\ \vdots \\ u_m \end{bmatrix} \in \mathbb{R}^m,\ y = \begin{bmatrix} y_1 \\ \vdots \\ y_p \end{bmatrix} \in \mathbb{R}^p,$$

and define

$$f : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R} \to \mathbb{R}^n$$
$$h : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R} \to \mathbb{R}^p$$

by

$$f(x,u,t) = \begin{bmatrix} f_1(x_1,\ldots,x_n,u_1,\ldots,u_m,t) \\ \vdots \\ f_n(x_1,\ldots,x_n,u_1,\ldots,u_m,t) \end{bmatrix},$$
$$h(x,u,t) = \begin{bmatrix} h_1(x_1,\ldots,x_n,u_1,\ldots,u_m,t) \\ \vdots \\ h_p(x_1,\ldots,x_n,u_1,\ldots,u_m,t) \end{bmatrix}.$$

Then the system equations simplify to

$$\left.\begin{aligned} \dot{x} &= f(x,u,t) \\ y &= h(x,u,t) \end{aligned}\right\} \tag{A.1}$$

Equations (A.1) are known as the **state space form** of the system. The vector space $\mathbb{R}^n$ in which the state of the system takes values is known as the **state space** of the system. If the system is of dimension 2, the state space is also referred to as the **phase plane**. The function f that determines the direction in which the state will move is known as the **vector field**.

Notice that the differential equation for x is first order, i.e. involves $\dot{x}$ but no higher derivatives of x. Sometimes the system dynamics are given to us in the form of higher order differential equations, i.e. equations involving a variable $\theta \in \mathbb{R}$ and its derivatives with respect to time up to $\frac{d^r\theta}{dt^r}$ for some integer $r \geq 1$. Such systems can be easily transformed to state space form by setting $x_1 = \theta$, $x_2 = \dot{\theta}$, $\ldots$, $x_{r-1} = \frac{d^{r-1}\theta}{dt^{r-1}}$.

Exercise A.3 *Consider the system*

$$\frac{d^r\theta}{dt^r} + g(\theta, \frac{d\theta}{dt}, \ldots, \frac{d^{r-1}\theta}{dt^{r-1}}) = 0$$

Write this system in state space form.

It may of course happen in certain examples that there are no inputs or outputs, or that there is no explicit dependence of the dynamics on time. Systems of the form

$$\dot{x} = f(x)$$

(i.e. without inputs or outputs and with no explicit dependence on time) are called **autonomous** systems.

Exercise A.4 *Is the pendulum an autonomous system?*

Exercise A.5 *Consider a non-autonomous system of the form $\dot{x} = f(x,t)$, of dimension n. Show that it can be transformed to an autonomous system of dimension $n+1$.*

A.2.1 Existence and Uniqueness of Solutions

Consider an autonomous dynamical system in state space form

$$\dot{x} = f(x)$$

and assume that at time $t = 0$ the state is equal to x_0, i.e.

$$x(0) = x_0$$

We would like to "solve" the dynamics of the system to determine how the state will evolve in the future (i.e. for $t \geq 0$). More precisely, given some $T > 0$ we would like to determine a function

$$x(\cdot) : [0,T] \to \mathbb{R}^n$$

such that

$$\begin{aligned} x(0) &= x_0 \\ \dot{x}(t) &= f(x(t)), \forall t \in [0,T]. \end{aligned}$$

Such a function $x(\cdot)$ is called a **trajectory** (or **solution**) of the system. Notice that given a candidate trajectory $x(\cdot) : [0,T] \to \mathbb{R}^n$ one needs to verify both the differential condition and the initial condition to ensure that $x(\cdot)$ is indeed a solution of the differential equation.

Exercise A.6 *Assume that instead of $x(0) = x_0$ it is required that $x(t_0) = x_0$ for some $t_0 \neq 0$. Show how one can construct solutions to the system*

$$x(t_0) = x_0,\ \dot{x} = f(x)$$

from solutions to

$$x(0) = x_0,\ \dot{x} = f(x)$$

by appropriately redefining t. Could you do this with a non-autonomous system?

Without any additional information, it is unclear whether one can find a function $x(\cdot)$ solving the differential equation. A number of things can go wrong.

Example (No solutions) Consider the one dimensional system

$$\dot{x} = -\operatorname{sgn}(x) = \begin{cases} -1 & \text{if } x \geq 0 \\ 1 & \text{if } x < 0, \end{cases}$$

with initial condition $x(0) = 0$. A solution to this differential equation does not exist for any $T \geq 0$.

Exercise A.7 *Assume that $x(0) = 1$. Show that solutions to the system exist for all $T \leq 1$ but not for $T > 1$.*

Incidentally, something similar would happen with the thermostat system of Section 2.3.2, if the thermostat insisted on switching the radiator on and off exactly at 20 degrees. ■

Example (Multiple Solutions) Consider the one dimensional system

$$\dot{x} = 3x^{2/3},\ x(0) = 0$$

All functions of the form

$$x(t) = \begin{cases} (t-a)^3 & t \geq a \\ 0 & t \leq a \end{cases}$$

for any $a \geq 0$ are solutions of this differential equation.

Exercise A.8 *Verify this.*

Notice that in this case the solution is not unique. In fact there are infinitely many solutions, one for each $a \geq 0$. ■

Example (Finite Escape Time) Consider the one dimensional system

$$\dot{x} = 1 + x^2,\ x(0) = 0$$

The function

$$x(t) = tan(t)$$

is a solution of this differential equation.

Exercise A.9 *Verify this. What happens at $t = \pi/2$?*

Notice that the solution is defined for $T < \pi/2$ but not for $T \geq \pi/2$. ■

To eliminate such pathological cases we need to impose some assumptions on f.

Definition A.1 (Lipshitz Continuity) *A function $f : \mathbb{R}^n \to \mathbb{R}^n$ is called Lipschitz continuous if there exists $\lambda > 0$ such that for all $x, \hat{x} \in \mathbb{R}^n$*

$$\|f(x) - f(\hat{x})\| < \lambda \|x - \hat{x}\|$$

λ is known as a *Lipschitz constant*. Notice that if λ is a Lipschitz constant then any other $\lambda' > \lambda$ is also a Lipschitz constant. A Lipschitz continuous function is continuous, but not necessarily differentiable. All differentiable functions with bounded derivatives are Lipschitz continuous.

Exercise A.10 *Show that for $x \in \mathbb{R}$ the function $f(x) = |x|$ that returns the absolute value of x is Lipschitz continuous. Provide a Lipschitz constant. Is f continuous? Is it differentiable?*

Theorem A.2 (Existence & Uniqueness of Solutions) *If f is Lipschitz continuous, then the differential equation*

$$\dot{x} = f(x)$$
$$x(0) = x_0$$

has a unique solution $x(\cdot) : [0, T] \to \mathbb{R}^n$ for all $T \geq 0$ and all $x_0 \in \mathbb{R}^n$.

Exercise A.11 *Three examples of dynamical systems that do not have unique solutions were given above. Why do these systems fail to meet the conditions of the theorem?*

This theorem allows us to check whether the differential equation models we develop make sense. It also allows us to spot potential problems with proposed solutions. For example, uniqueness implies that solutions can not cross.

Exercise A.12 *Why does uniqueness imply that solutions can not cross?*

A.2.2 Continuity and Simulation

Theorem A.3 (Continuity with Initial State) *Assume f is Lipschitz continuous with Lipschitz constant λ. Let $x(\cdot) : [0, T] \to \mathbb{R}^n$ and $\hat{x}(\cdot) : [0, T] \to \mathbb{R}^n$ be solutions to $\dot{x} = f(x)$ with $x(0) = x_0$ and $\hat{x}(0) = \hat{x}_0$ respectively. Then for all $t \in [0, T]$*

$$\|x(t) - \hat{x}(t)\| \leq \|x_0 - \hat{x}_0\| e^{\lambda t}$$

In other words, solutions that start close to one another remain close to one another.

This theorem provides another indication that dynamical systems with Lipschitz continuous vector fields are well behaved. For example, it provides theoretical justification for simulation algorithms. Most nonlinear differential equations are impossible to solve by hand. One can however approximate the solution on a computer, using numerical algorithms for

computing integrals (Euler, Runge-Kutta, etc.). This is a process known as **simulation**.

Powerful computer packages, such as Matlab, make the simulation of most systems relatively straight forward. For example, the pendulum trajectories can be generated using a Matlab function

```
function [xprime] = pendulum(t,x)
xprime=[0; 0];
l = 1;
m=1;
d=1;
g=9.8;
xprime(1) = x(2);
xprime(2) = -sin(x(1))*g/l-x(2)*d/m;
```

The simulation code is then simply

```
>> x=[0.75 0];
>> [T,X]=ode45('pendulum', [0 10], x');
>> plot(T,X);
>> grid;
```

The continuity property ensures that the numerical approximation to the solution computed by the simulation algorithms and the actual solution remain close.

When one studies hybrid systems, many of these nice properties unfortunately vanish. As the non-deterministic thermostat system suggests, existence and uniqueness of solutions are much more difficult to guarantee. Continuity is usually impossible.

A.2.3 Control systems

In subsequent discussion we will also consider continuous state control systems of the form

$$\begin{cases} \dot{x} = f(x,u) \\ x \in \mathbb{R}^n \\ u \in U \subseteq \mathbb{R}^m \\ f(\cdot,\cdot) : \mathbb{R}^n \times U \to \mathbb{R}^n, \end{cases} \tag{A.2}$$

For systems of this type one would also like to characterise the existence of solutions. In other words, given an initial condition x_0, a time horizon $T \geq 0$ and an input function $u(\cdot) : [0,T] \to U$ one would like to find a function $x(\cdot) : [0,T] \to \mathbb{R}^n$ such that $x(0) = x_0$ and $\dot{x}(t) = f(x(t),u(t))$.

Even though this definition of solution is in principle good, in practice it is often inadequate. The reason is that to be able to find a solution such that the derivative $\dot{x}$ exists for all $t \in [0,T]$ we need to restrict our attention

to input functions which are least continuous. However, in many practical problems (e.g. many optimal control problems) we would also like to allow input functions which are discontinuous. In this case the derivative $\dot{x}$ will be undefined at the discontinuity points of $u(\cdot)$.

Fortunatelly, it turns out that as long as the function $u(\cdot)$ is not too wild (e.g. is discintinuous only at a finite number of points in the interval $[0, T]$), the solutions of the differential equation can be defined despite this technical difficulty, by requiring $x(\cdot)$ to meet the condition $\dot{x}(t) = f(x(t), u(t))$ for *almost every* $t \in [0, T]$ (instead for all $t \in [0, T]$). The precise meaning of "almost every" and the precise definition of the class of functions $u(\cdot)$ that are acceptable in this context can be developed using tools from measure theory. The "acceptable" functions are known as *Lebesgue measurable functions*. We will use $\mathcal{U}_{[t,t']}$ to denote the set of Lebesgue measurable functions from the interval $[t, t']$ to U.

With the right definitions in place one can show that solutions for the control system exist and are unique.

Theorem A.4 *If $f(x, u)$ is Lipschitz continuous in x and continuous in u then the control system of equation A.2 admits a unique solution $x(\cdot) : [0, T] \to \mathbb{R}^n$ for all initial conditions $x_0 \in \mathbb{R}^n$, time horizons $T \geq 0$ and measurable control functions $u(\cdot) \in \mathcal{U}_{[0,T]}$.*

Similar extensions of the continuity theorem given above for autonomous systems are also possible.

A.3 Review of discrete systems (Claire)

A.4 Bibliography and Further Reading

The material on continuous systems is thoroughly covered in any good textbook on nonlinear dynamical systems, see for example [4, 5, 6, 7]. Discrete state systems have also been studied for many years, especially in computer science. Good textbooks include [8, 9, 10]. For a very thorough coverage of the basics of control systems see [145].

Appendix B
Review of Optimal Control and Games

B.1 Optimal Control and the Calculus of Variations

There are numerous excellent books on optimal control. Commonly used books which we will draw from are Athans and Falb [**?**], Berkovitz [**?**], Bryson and Ho [**?**], Pontryagin et al [**?**], Young [**?**], Kirk [**?**], Lewis [**?**] and Fleming and Rishel[**?**]. The history of optimal control is quite well rooted in antiquity, with allusion being made to Dido, the first Queen of Carthage, who when asked to take as much land as could be covered by an ox-hide, cut the ox-hide into a tiny strip and proceeded to enclose the entire area of what came to be know as Carthage in a circle of the appropriate radius[1]. The calculus of variations is really the ancient precursor to optimal control. Iso perimetric problems of the kind that gave Dido her kingdom were treated in detail by Tonelli and later by Euler. Both Euler and Lagrange laid the foundations of mechanics in a variational setting culminating in the Euler Lagrange equations. Newton used variational methods to determine the shape of a body that minimizes drag, and Bernoulli formulated his brachistochrone problem in the seventeenth century, which attracted the attention of Newton and L'Hôpital. This intellectual heritage was revived and generalized by Bellman [**?**] in the context of dynamic programming

[1] The optimal control problem here is to enclose the maximum area using a closed curve of given length.

and by Pontryagin and his school in the so-called Pontryagin principle for optimal control ([**?**]).

Consider a nonlinear possibly time varying dynamical system described by

$$\dot{x} = f(x, u, t) \tag{B.1}$$

with state $x(t) \in \mathbb{R}^n$ and the control input $u \in \mathbb{R}^{n_i}$. Consider the problem of minimizing the performance index

$$J = \phi(x(t_f), t_f) + \int_{t_0}^{t_f} L(x(t), u(t), t)dt \tag{B.2}$$

where t_0 is the initial time, t_f the final time (free), $L(x, u, t)$ is the running cost, and $\phi(x(t_f), t_f)$ is the cost at the terminal time. The initial time t_0 is assumed to be fixed and t_f variable. Problems involving a cost only on the final and initial state are referred to as Mayer problems, those involving only the integral or running cost are called Lagrange problems and costs of the form of equation (B.2) are referred to as Bolza problems. We will also have a constraint on the final state given by

$$\psi(x(t_f), t_f) = 0 \tag{B.3}$$

where $\psi : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^p$ is a smooth map. To derive necessary conditions for the optimum, we will perform the calculus of variations on the cost function of (B.2) subject to the constraints of equations (B.1), (B.3). To this end, define the modified cost function, using the Lagrange multipliers $\lambda \in \mathbb{R}^p, p(t) \in \mathbb{R}^n$,

$$\tilde{J} = \phi(x(t_f), t_f) + \lambda^T \psi(x(t_f), t_f) + \int_{t_0}^{t_f} \left[L(x, u, t) + p^T (f(x, u, t) - \dot{x}) \right] dt \tag{B.4}$$

Defining the *Hamiltonian* $H(x, u, t)$ using what is referred to as a *Legendre transformation*

$$H(x, p, u, t) = L(x, u, t) + p^T f(x, u, t) \tag{B.5}$$

The variation of (B.4) is given by assuming independent variations in $\delta u(), \delta x(), \delta p(), \delta\lambda$, and δt_f:

$$\begin{array}{rcl} \delta\tilde{J} = & (D_1\phi + D_1\psi^T\lambda)\delta x|_{t_f} + (D_2\phi + D_2\psi^T\lambda)\delta t|_{t_f} + \psi^T\delta\lambda \\ + & (H - p^T\dot{x})\delta t|_{t_f} \\ + & \int_{t_0}^{t_f} \left[D_1 H\delta x + D_3 H\delta u - p^T\delta\dot{x} + (D_2 H^T - \dot{x})^T\delta p \right] dt \end{array} \tag{B.6}$$

The notation $D_i H$ stands for the derivative of H with respect to the i th argument. Thus, for example,

$$D_3 H(x, p, u, t) = \frac{\partial H}{\partial u} \quad D_1 H(x, p, u, t) = \frac{\partial H}{\partial x}$$

Integrating by parts for $\int p^T \delta\dot{x} dt$ yields

$$\begin{aligned} \delta\tilde{J} = & \quad (D_1\phi + D_1\psi^T\lambda - p^T)\delta x(t_f) + (D_2\phi + D_2\psi^T\lambda + H - p^T\dot{x})\delta t_f + \psi^T\delta\lambda \\ + & \quad \int_{t_0}^{t_f} \left[(D_1 H + \dot{p}^T)\delta x + D_3 H \delta u + (D_2^T H - \dot{x})^T \delta p\right] dt \end{aligned} \tag{B.7}$$

An extremum of $\tilde{J}$ is achieved when $\delta\tilde{J} = 0$ for all independent variations $\delta\lambda, \delta x, \delta u, \delta p$. These conditions are recorded in the following

Table of necessary conditions for optimality:

Table 1

Description	Equation	Variation
Final State constraint	$\psi(x(t_f), t_f) = 0$	$\delta\lambda$
State Equation	$\dot{x} = \frac{\partial H}{\partial p}^T = \text{f (x, u, t)}$	δp
Costate equation	$\dot{p} = -\frac{\partial H}{\partial x}^T$	δx
Input stationarity	$\frac{\partial H}{\partial u} = 0$	δu
Boundary conditions	$D_1\phi - p^T = -D_1\psi^T\lambda\|_{t_f}$	$\delta x(t_f)$
	$H - p^T f(x,u,t) + D_2\phi = -D_2\psi^T\lambda\|_{t_f}$	δt_f

The conditions of Table (B.1) and the boundary conditions $x(t_0) = x_0$ and the constraint on the final state $\psi(x(t_f), t_f) = 0$ constitute the necessary conditions for optimality. The end point constraint equation is referred to as the transversality condition:

$$\begin{aligned} D_1\phi - p^T = & \quad -D_1\psi^T\lambda \\ H + D_2\phi = & \quad -D_2\psi^T\lambda \end{aligned} \tag{B.8}$$

The optimality conditions may be written explicitly as

$$\begin{aligned} \dot{x} = & \quad \frac{\partial H}{\partial p}^T(x, u^*, p) \\ \dot{p} = & \quad -\frac{\partial H}{\partial x}^T(x, u^*, p) \end{aligned} \tag{B.9}$$

with the stationarity condition reading

$$\frac{\partial H}{\partial u}(x, u^*, p) = 0$$

and the endpoint constraint $\psi(x(t_f), t_f) = 0$. *The key point to the derivation of the necessary conditions of optimality is that the Legendre transformation of the Lagrangian to be minimized into a Hamiltonian converts a functional minimization problem into a static optimization problem on the function* $H(x, u, p, t)$.

The question of when these equations also constitute sufficient conditions for (local) optimality is an important one and needs to be ascertained by taking the second variation of $\tilde{J}$. This is an involved procedure but the input stationarity condition in Table (B.1) hints at the **sufficient condition for local minimality** of a given trajectory $x^*(\cdot), u^*(\cdot), p^*(\cdot)$ being a local minimum as being that the Hessian of the Hamiltonian,

$$D_2^2 H(x^*, u^*, p^*, t) \tag{B.10}$$

being positive definite along the optimal trajectory. A sufficient condition for this is to ask simply that the $n_i \times n_i$ Hessian matrix

$$D_2^2 H(x, u, p, t) \tag{B.11}$$

be positive definite. As far as conditions for **global minimality** are concerned, again the stationarity condition hints at a sufficient condition for global minimality being that

$$u^*(t) = \underset{\{\text{ min over u }\}}{\text{argmin}} \quad H(x^*(t), u, p^*(t), t) \tag{B.12}$$

Sufficient conditions for this are, for example, the convexity of the Hamiltonian $H(x, u, p, t)$ in u.

Finally, there are instances in which the Hamiltonian $H(x, u, p, t)$ is not a function of u at some values of x, p, t. These cases are referred to as *singular extremals* and need to be treated with care, since the value of u is left unspecified as far as the optimization is concerned.

B.1.1 Fixed Endpoint problems

In the instance that the final time t_f is fixed, the equations take on a simpler form, since there is no variation in δt_f. Then, the boundary condition of equation (B.8) becomes

$$p^T(t_f) = D_1\phi + D_1\psi^T\lambda|_{t_f} \tag{B.13}$$

Further, if there is no final state constraint the boundary condition simplifies even further to

$$p(t_f) = D_1\phi^T|_{t_f} \tag{B.14}$$

B.1.2 Time Invariant Systems

In the instance that $f(x, u, t)$ and the running cost $L(x, u, t)$ are not explicitly functions of time, there is no final state constraint and the final time t_f is fixed, the formulas of Table (B.1) can be rewritten as

State Equation	$\dot{x} = \frac{\partial H}{\partial p}^T = f(x, u^*)$
Costate Equation	$\dot{p} = -\frac{\partial H}{\partial x}^T = -D_1 f^T p + D_1 L^T$
Stationarity Condition	$0 = \frac{\partial H}{\partial u} = D_2 L^T + D_2 f^T p$
Transversality Conditions	$D_1\phi - p^T = -D_1\psi^T\lambda$

In addition, it may be verified that

$$\frac{dH^*}{dt} = \frac{\partial H^*}{\partial x}(x,p)\dot{x} + \frac{\partial H^*}{\partial p}\dot{p} = 0 \tag{B.15}$$

thereby establishing that $H^*(t) \equiv H^*(t_f)$.

B.1.3 Connections with Classical Mechanics

Hamilton's principle of least action states (under certain conditions [2]) that a conservative system moves so as to minimize the time integral of its "action", defined to be the difference between the kinetic and potential energy. To make this more explicit we define $q \in \mathbb{R}^n$ to be the vector of generalized coordinates of the system and denote by $U(q)$ the potential energy of the system and $T(q, \dot{q})$ the kinetic energy of the system. Then Hamilton's principle of least action asks to solve an optimal control problem for the system

$$\dot{q} = u$$

with Lagrangian

$$L(q, u) = T(q, u) - U(q)$$

The equations (B.9) in this context have $H(q, u, p) = L(q, u) + p^T u$. $u^* = u^*(p, q)$ is chosen so as to minimize the Hamiltonian H. A necessary condition for stationarity is that $u^*(p, q)$ satisfies

$$0 = \frac{\partial H}{\partial u} = \frac{\partial L}{\partial u} + p^T \tag{B.16}$$

The form of the equations (B.9) in this context is that of the familiar *Hamilton Jacobi equations*. The costate p has the interpretation of

[2] For example, there is no dissipation or no nonholonomic constraints. Holonomic or integrable constraints are dealt with by adding appropriate Lagrange multipliers. If nonholonomic constraints are dealt with in the same manner, we get equations of motion, dubbed vakonomic by Arnold [146] which do not correspond to experimentally observed motions. On the other hand, if there are only holonomic constraints, the equations of motion that we derive from Hamilton's principle of least action is equivalent to Newton's laws.

momentum.

$$\begin{array}{ll} \dot{q} = & \frac{\partial H^*}{\partial p}(p,q) \quad = u^*(p,q) \\ \dot{p} = & -\frac{\partial H^*}{\partial q}(p,q) \end{array} \tag{B.17}$$

Combining the second of these equations with (B.16) yields the familiar *Euler Lagrange equations*

$$\frac{\partial L}{\partial q} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) = 0 \tag{B.18}$$

B.2 Optimal Control and Dynamic Programming

To begin this discussion, we will embed the optimization problem which we are solving in a larger class of problems, more specifically we will consider the original cost function of equation (B.2) from an initial time $t \in [t_0, t_f]$ by considering the cost function on the interval $[t, t_f]$:

$$J(x(t), t) = \phi(x(t_f), t_f) + \int_t^{t_f} L(x(\tau), u(\tau), \tau) d\tau$$

Bellman's principle of optimality says that if we have found the optimal trajectory on the interval from $[t_0, t_f]$ by solving the optimal control problem on that interval, the resulting trajectory is also optimal on all subintervals of this interval of the form $[t, t_f]$ with $t > t_0$, provided that the initial condition at time t was obtained from running the system forward along the optimal trajectory from time t_0. The optimal value of $J(x(t), t)$ is referred to as the "cost-to go". To be able to state the following key theorem of optimal control we will need to define the "optimal Hamiltonian" to be

$$H^*(x, p, t) := H(x, u^*, p, t)$$

Theorem B.1 The Hamilton Jacobi Bellman equation

Consider, the time varying optimal control problem of (B.2) with fixed endpoint t_f and time varying dynamics. If the optimal value function, i.e. $J^*(x(t_0), t_0)$ is a smooth function of x, t, then $J^*(x, t)$ satisfies the **Hamilton Jacobi Bellman** *partial differential equation*

$$\frac{\partial J^*}{\partial t}(x, t) = -H^*(x, \frac{\partial J^*}{\partial x}(x, t), t) \tag{B.19}$$

with boundary conditions given by $J^*(x, t_f) = \phi(x, t_f)$ for all $x \in \{x : \psi(x, t_f) = 0\}$.

Proof: The proof uses the principle of optimality. This principle says that if we have found the optimal trajectory on the interval from $[t, t_f]$ by solving the optimal control problem on that interval, the resulting trajectory is also optimal on all subintervals of this interval of the form $[t_1, t_f]$ with $t_1 > t$, provided that the initial condition at time t_1 was obtained from running

the system forward along the optimal trajectory from time t. Thus, from using $t_1 = t + \Delta t$, it follows that

$$J^*(x,t) = \min_{\substack{u(\tau) \\ t \le \tau \le t+\Delta t}} \left[\int_t^{t+\Delta t} L(x,u,\tau) d\tau + J^*(x+\Delta x, t+\Delta t) \right] \tag{B.20}$$

Taking infinitesimals and letting $\Delta t \to 0$ yields that

$$-\frac{\partial J^*}{\partial t} = \min_{u(t)} \left(L + (\frac{\partial J^*}{\partial x}) f \right) \tag{B.21}$$

with the boundary condition being that the terminal cost is

$$J^*(x,t_f) = \phi(x,t_f)$$

on the surface $\psi(x) = 0$. Using the definition of the Hamiltonian in equation (B.5), it follows from equation (B.21) that the Hamilton Jacobi equation of equation (B.19) holds.

□

Remarks:

1. The preceding theorem was stated as a necessary condition for extremal solutions of the optimal control problem. As far as minimal and global solutions of the optimal control problem, the Hamilton Jacobi Bellman equations read as in equation (B.21). In this sense, the form of the Hamilton Jacobi Bellman equation in (B.21) is more general.

2. The **Eulerian conditions** of Table (B.1) are easily obtained from the Hamilton Jacobi Bellman equation by proving that $p^T(t) := \frac{\partial J^*}{\partial x}(x,t)$ satisfies the costate equations of that Table. Indeed, consider the equation (B.21). Since $u(t)$ is unconstrained, it follows that it should satisfy

$$\frac{\partial L}{\partial u}(x^*,u^*) + \frac{\partial f}{\partial u}^T p = 0 \tag{B.22}$$

Now differentiating the definition of $p(t)$ above with respect to t yields

$$\frac{dp}{dt}^T = \frac{\partial^2 J^*}{\partial t \partial x}(x^*,t) + \frac{\partial^2 J^*}{\partial x^2} f(x^*,u^*,t) \tag{B.23}$$

Differentiating the Hamilton Jacobi equation (B.21) with respect to x and using the relation (B.22) for a stationary solution yields

$$-\frac{\partial^2 J^*}{\partial t \partial x}(x^*,t) = \frac{\partial L}{\partial x} + \frac{\partial^2 J^*}{\partial x^2} f + p^T \frac{\partial f}{\partial x} \tag{B.24}$$

Using equation (B.24) in equation (B.23) yields

$$-\dot{p} = \frac{\partial f}{\partial x}^T p + \frac{\partial L}{\partial x}^T \tag{B.25}$$

establishing that p is indeed the co-state of Table 1. The boundary conditions on $p(t)$ follow from the boundary conditions on the Hamilton Jacobi Bellman equation.

B.2.1 Constrained Input Problems

In the instance that there are no constraints on the input, the extremal solutions of the optimal control problem are found by simply extremizing the Hamiltonian and deriving the stationarity condition. Thus, if the specification is that $u(t) \in U \subset \mathbb{R}^{n_i}$ then, the optimality condition is that

$$H(x^*, u^*, p^*, t) \leq H(x^*, u, p^*, t) \quad \forall u \in U \tag{B.26}$$

If the Hamiltonian is convex in u and U is a convex set, there are no specific problems with this condition. In fact, when there is a single input and the set U is a single closed interval, there are several interesting examples of Hamiltonians for which the optimal inputs switch between the endpoints of the interval, resulting in what is referred to as **bang bang control**. However, problems can arise when U is either not convex or compact. In these cases, a concept of a **relaxed control** taking values in the convex hull of U needs to be introduced. As far as an implementation of a control $u(t) \in \text{conv}U$, but not in U, a probabilistic scheme involving switching between values of U whose convex combination u is needs to be devised.

B.2.2 Free end time problems

In the instance that the final time t_f is free, the transversality conditions are that

$$p^T(t_f) = \quad D_1\phi + D_1\psi^T\lambda \tag{B.27}$$

B.2.3 Minimum time problems

A special class of minimum time problems of especial interest is minimum time problems, where t_f is to be minimized subject to the constraints. This is accounted for by setting the Lagrangian to be 1, and the terminal state cost $\phi \equiv 0$, so that the Hamiltonian is $H(x, u, p, t) = 1 + p^T f(x, u, t)$. Note that by differentiating $H(x, u, p, t)$ with respect to time, we get

$$\frac{dH^*}{dt} = D_1H^*\dot{x} + D_2H^*\dot{u} + D_3H^*\dot{p} + \frac{\partial H^*}{\partial t} \tag{B.28}$$

Continuing the calculation using the Hamilton Jacobi equation,

$$\frac{dH^*}{dt} = (\frac{\partial H^*}{\partial x} + \dot{p})f(x, u^*, t) + \frac{\partial H^*}{\partial t} = \frac{\partial H^*}{\partial t} \tag{B.29}$$

In particular, if H^* is not an explicit function of t, it follows that $H^*(x, u, p, t) \equiv H$. Thus, for minimum time problems for which $f(x, u, t)$ and $\psi(x, t)$ are not explicitly functions of t, it follows that $H(t_f) \equiv H(t)$.

B.3 Two person Zero Sum Dynamical games

The theory of games also has a long and distinguished, if not as long a history. Borel encountered saddle points in a matrix in 1915, but it really took von Neumann to prove his fundamental theorem about the existence of mixed strategies for achieving a saddle solution in games in 1936. In a classic book, von Neumann and Morgenstern ([**?**]) laid out the connections between static games and economic behavior. Nash, von Stackelberg and others extended the work to N person non-cooperative games. This work has continued in many important new directions in economics. Differential or dynamical games showed up in the work of Isaacs in 1969 and rapidly found fertile ground in the control community where it has progressed. There are several nice books on the subject of dynamical games, our treatment is drawn heavily from Basar and Olsder ([**?**]).

Consider a so-called dynamical, two person zero sum, perfect state information game modeled by

$$\dot{x} = f(x, u, d, t) \qquad x(t_0) = x_0 \tag{B.30}$$

on a fixed duration $[t_0, t_f]$. Here $x \in \mathbb{R}^n$ models the state of the system and $u \in \mathbb{R}^{n_1}, d \in \mathbb{R}^{n_2}$ represent the actions of the two players. The game is said to be *zero sum* if one player seeks to minimize and the other to maximize the same cost function taken to be of the form

$$J = \phi_(x(t_f), t_f) + \int_{t_0}^{t_f} L(x, u, d, t)dt \tag{B.31}$$

We will assume that player 1 (u) is trying to minimize J and player 2 (d) is trying to maximize J. For simplicity we have omitted the final state constraint and also assumed the end time t_f to be fixed. These two assumptions are made for simplicity but we will discuss the t_f free case when we study pursuit evasion games. The game is said to have perfect information if both players have access to the full state $x(t)$. The solution of two person zero sum games proceeds very much along the lines of the optimal control problem by setting up the Hamiltonian

$$H(x, u, d, p, t) = L(x, u, d, t) + p^T f(x, u, d, t) \tag{B.32}$$

Rather than simply minimizing $H(x,u,d,p,t)$ the game is said to have a *saddle point solution* if the following analog of the saddle point condition for two person zero sum static games holds:

$$\min_{\mathrm{u}} \max_{\mathrm{d}} H(x,u,d,p,t) = \max_{\mathrm{d}} \min_{\mathrm{u}} H(x,u,d,p,t) \quad \text{(B.33)}$$

If the minmax is equal to the maxmin, the resulting optimal Hamiltonian is denoted $H^*(x,p,t)$ and the optimal inputs u^*, d^* are determined to be respectively,

$$u^*(t) = \operatorname*{argmin}_{\mathrm{u}} \left(\max_{\mathrm{d}} H(x,u,d,p,t) \right) \quad \text{(B.34)}$$

and

$$d^*(t) = \operatorname*{argmax}_{\mathrm{d}} \left(\min_{\mathrm{u}} H(x,u,d,p,t) \right) \quad \text{(B.35)}$$

The equations for the state and costate and the transversality conditions are given as before by

$$\begin{aligned} \dot{x} &= \tfrac{\partial H^*}{\partial p}^T(x,p) \\ \dot{p} &= -\tfrac{\partial H^*}{\partial x}^T(x,p) \end{aligned} \quad \text{(B.36)}$$

with boundary conditions $x(t_0) = x_0$ and $p^T(t_f) = D_1\phi(x_{t_f})$, and the equation is the familiar *Hamilton Jacobi equation*. As before, one can introduce the optimal cost to go $J^*(x(t),t)$ and we have the following analog of Theorem (B.1):

Theorem B.2 The Hamilton Jacobi Isaacs equation

Consider, the two person zero sum differential game problem of (B.31) with fixed endpoint t_f. If the optimal value function, i.e. $J^*(x(t_0),t_0)$ is a smooth function of x,t, then $J^*(x,t)$ satisfies the **Hamilton Jacobi Isaacs** *partial differential equation*

$$\frac{\partial J^*}{\partial t}(x,t) = -H^*(x, \frac{\partial J^*}{\partial x}(x,t), t) \quad \text{(B.37)}$$

with boundary conditions given by $J^*(x,t_f) = \phi(x)$ for all x.

Remarks

1. We have dealt with saddle solutions for unconstrained input signals u,d thus far in the development. If the inputs are constrained to lie in sets U,D respectively the saddle solutions can be guaranteed to exist if

$$\min_{u \in U} \max_{d \in D} H(x,u,d,p,t) = \max_{d \in D} \min_{u \in U} H(x,u,d,p,t) \quad \text{(B.38)}$$

   Again, if the input sets are not convex, relaxed controls may be needed to achieve the min-max.

2. The sort of remarks that were made about free endpoint optimal control problems can also be made of games.

3. In our problem formulation for games, we did not include explicit terminal state constraints of the form $\psi(x(t_f), t_f) = 0$. These can be easily included, and we will study this situation in greater detail under the heading of pursuit evasion games.

4. The key point in the theory of dynamical games is that the Legendre transformation of the Lagrangian cost function into the Hamiltonian function converts the solution of the "dynamic" game into a "static" game, where one needs to find a saddle point of the Hamiltonian function $H(x, u, d, p, t)$. This is very much in the spirit of the calculus of variations and optimal control.

B.4 N person Dynamical Games

When there are N persons playing a game, many new and interesting new possibilities arise. There is a scenario in which the N agents are non-cooperative, and another in which they cooperate in teams. Of course, if the information available to each of them is different, this makes the solution even more interesting. In this section, we will assume that each of the agents has access to the full state of the system. In this section, we will only discuss non-cooperative solution concepts: first the Nash solution concept and then briefly the Stackelberg solution concept. Cooperative games with total cooperation are simply optimal control problems. If there is cooperation among teams, this can be viewed as a noncooperative game between the teams. When however there are side payments between teams the scope of the problem increases quite considerably.

B.4.1 Non-cooperative Nash solutions

When there are N players each able to influence the process by controls $u_i \in \mathbb{R}^{n_i}, i = 1, \ldots, N$, modeled by

$$\dot{x} = f(x, u_1, \ldots, u_N, t) \tag{B.39}$$

and each cost functional (to be minimized) is of the form

$$J_i(u_1(\cdot), \ldots, u_N(\cdot)) = \phi_i(x(t_f), t_f) + \int_{t_0}^{t_f} L_i(x, u_1, \ldots, u_N, t)dt \tag{B.40}$$

different solution concepts need to be invoked. The simplest non-cooperative solution strategy is a so-called *non-cooperative Nash equilibrium*. A set of controls $u_i^*, i = 1, \ldots, N$ is said to be a Nash strategy if for

each player modifying that strategy, and assuming that the others play their Nash strategies, results in an increase in his payoff, that is for $i = 1, \ldots, N$

$$J_i(u_1^*, \ldots, u_i, \ldots, u_N^*) \geq J_i(u_1^*, \ldots, u_i^*, \ldots, u_N^*) \qquad \forall u_i(\cdot) \tag{B.41}$$

It is important to note that Nash equilibria may not be unique. It is also easy to see that for 2 person zero sum games, a Nash equilibrium is a saddle solution.

As in the previous section on saddle solutions, we can write Hamilton Jacobi equations for Nash equilibria by defining Hamiltonians $H_i(x, u_1, \ldots, u_N, p, t)$ according to

$$H_i(x, u_1, \ldots, u_N, p, t) = L_i(x, u_1, \ldots, u_N) + p^T f(x, u_1, \ldots, u_N, t) \tag{B.42}$$

The conditions for a Nash equilibrium of equation (B.41) are there exist $u_i^*(x, p, t)$ such that

$$H_i(x, u_1^*, \ldots, u_i, \ldots, u_N^*, p, t) \geq H_i(x, u_1^*, \ldots, u_i^*, \ldots, u_N^*, p, t) \tag{B.43}$$

Then, we have N sets of Hamilton Jacobi equations for the N players satisfying the Hamilton Jacobi equations with $H_i^* = H_i^*(x, u_1^*, \ldots, u_N^*, p_i, t)$. Note that we have changed the costate variables to p_i to account for different Hamiltonians and boundary conditions.

$$\begin{aligned} \dot{x} &= \frac{\partial H_i^*}{\partial p_i}^T \\ \dot{p}_i &= -\frac{\partial H_i^*}{\partial x}^T \end{aligned} \tag{B.44}$$

with transversality conditions $p_i^T(t_f) = -D_1\phi_i(x(t_f), t_f)$.

B.4.2 Noncooperative Stackelberg Solutions

Stackelberg or hierarchical equilibria refer to noncooperative solutions, where one or more of the players act as leaders. We will illustrate the solution concept for a two person game where player 1 is the leader and player 2 the follower. Once player 1 announces a strategy $u_1^o(\cdot)$, if player 2 is rational he choose his strategy so as to minimize his cost J_2 with the dynamics

$$\dot{x} = f(x, u_1^o, u_2, t) \tag{B.45}$$

and

$$J_2(u_2) = \phi_2(x(t_f), t_f) + \int_{t_0}^{t_f} L_2(x, u_1^o(t), u_2, t)dt$$

Thus, $u_2^*(u_1^o)$ is chosen to minimize $H_2(x, u_1^o, u_2, p_2, t)$, where p_2 satisfies the differential equation

$$\dot{p}_2 = -\frac{\partial H_2}{\partial x}^T (x, u_1^o, u_2(u_1^o), p, t) \quad p_2(t_f) = D_1^T \phi_2(x(t_f), t_f)$$

In turn the leader chooses his strategy to be that u_1^* which minimizes J_1 subject to the assumption that player 2 will rationally play $u_2^*(u_1^*)$. Thus, the system of equations that he has to solve to minimize J_1 subject to

$$\begin{array}{rll} \dot{x} = & f(x, u_1, u_2, t) & x(t_0) = x_0 \\ \dot{p}_2 = & -\frac{\partial H_2}{\partial x}^T(x, u_1^o, u_2(u_1^o), p, t) & p_2(t_f) = D_1^T \phi_2(x(t_f), t_f) \\ 0 = & D_3 H_2(x, u_1, u_2, p_2, t) & \end{array} \tag{B.46}$$

The last equation in (B.46) is the stationarity condition for minimizing H_2. The optimization problem of the system in (B.46) is not a standard optimal control in $\mathbb{R}^{2n}$ because there is an equality to be satisfied. Thus, Lagrange multipliers (co-states) taking values in $\mathbb{R}^{2n+n_2}$ for $t \in [t_0, t_f]$ are needed.

Appendix C
Hybrid System Viability (John)

C.1 Reachability with Inputs

In addition to discrete and continuous state variables, many hybrid systems also contain input variables U, possibly divided into discrete (U_D) and continuous (U_C). Depending on the application, inputs may influence

1. Continuous evolution, through the vector field
$$f(\cdot,\cdot,\cdot) : Q \times X \times U \to \mathbb{R}^n.$$
2. When discrete transitions take place, through the domain
$$\mathrm{D}om(\cdot) : Q \to 2^{X \times U}.$$
3. The destination of discrete transitions, through the reset map
$$R(\cdot,\cdot,\cdot) : E \times X \times U \to 2^X.$$

One can pause a number of interesting problems for hybrid systems with inputs, that make no sense for autonomous hybrid systems. For example, one can study stabilisation, optimal control, reachability specifications, etc. As before we will restrict our attention to reachability specifications. Depending on what the inputs are supposed to represent, reachability questions for hybrid systems with inputs can take a number of forms:

1. **Viability**. Roughly speaking, this involves answering the question "Does there exist a choice for the inputs u such that the executions of the hybrid system remain in a given set?". In this case one can think

of the inputs as controls, that can be used to steer the executions of the system.

2. **Invariance**. This involves answering the question "Do the executions of the system remain in a given set for all choices of u?". In this case one can think of the inputs as uncontrollable disturbances that can steer the system outside the desired set.

3. **Gaming**. In this case some of the input variables play the role of controls, while others play the role of disturbances. The relevant question in this case is "Does there exist a choice for the controls, such that despite the action of the disturbances the execution of the system remain is a given set?"

Working with hybrid systems with inputs is considerably more complicated. Even defining the precise semantics of an execution of the system is far from straight forward. Additional complications arise when one considers gaming problems, since one has to introduce assumptions about information exchange among the players, appropriate notions of strategy, etc. Here we will restrict our attention to a special case of the general problem for which precise mathematical answers can be formulated for these questions. In particular, we will study questions of viability and invariance for a class of hybrid systems known as *impulse differential inclusions*. The proofs are rather technical and are included in the notes only for the sake of completeness. For more details please refer to [45].

C.2 Impulse Differential Inclusions

Definition C.1 (Impulse Differential Inclusion) *An impulse differential inclusion is a collection $H = (X, F, R, J)$, consisting of a finite dimensional vector space X, a set valued map $F : X \to 2^X$, regarded as a* differential inclusion $\dot{x} \in F(x)$*, a set valued map $R : X \to 2^X$, regarded as a* reset map*, and a set $J \subseteq X$, regarded as a* forced transition set*.*

We call $x \in X$ the *state* of the impulse differential inclusion. Subsequently, $I = X \setminus J$ will be used to denote the complement of J. Comparing Definition C.1 with Definition 3.1, we see that the set I plays a similar role for the impulse differential inclusion that D*om* played for hybrid automata. The differential inclusion, $F(\cdot)$, plays a similar role to the vector field, $f(\cdot, \cdot)$. Finally, the domain of the reset map R plays the same role as the guards G of a hybrid automaton, and the image of the reset map R plays the same role as the reset relation (also denoted by R) of the hybrid automaton. Note that the initial states are not explicitly mentioned, and that there are no discrete states. Discrete states, q, can be introduced into the model, by embedding them in $\mathbb{R}$ (e.q., $q = 1, 2, 3, \ldots$) and introducing a trivial differential inclusion $\dot{q} \in \{0\}$ to capture their continuous evolution. This

is somewhat cumbersome to do, so we will state the results without this additional complication.

Impulse differential inclusions are extensions of differential inclusions and discrete time systems over finite dimensional vector spaces. A differential inclusion,

$$\dot{x} \in F(x),$$

over a finite dimensional vector space X can be thought of as an impulse differential inclusion, (X, F, R, J), with $R(x) = \emptyset$ for all $x \in X$ and $J = \emptyset$. Likewise, a discrete time system,

$$x_{k+1} \in R(x_k),$$

can be thought of as an impulse differential inclusion, $H = (X, F, R, J)$, with $F(x) = \{0\}$ for all $x \in X$ and $J = X$. In the control literature, differential inclusions and discrete time systems are frequently used to model continuous and discrete control systems. The continuous control system

$$\dot{x} = f(x, u), \quad u \in U(x)$$

with $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ and $U : \mathbb{R}^n \to 2^{\mathbb{R}^m}$ can be though of as a differential inclusion

$$\dot{x} \in F(x) = \{f(x, u) \mid u \in U(x)\}.$$

Likewise, the discrete time control system

$$x_{k+1} = r(x_k, u_k), \quad u_k \in U(x_k)$$

with $x_k \in \mathbb{R}^n$, $u_k \in \mathbb{R}^m$, $r : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ and $U : \mathbb{R}^n \to 2^{\mathbb{R}^m}$ can be though of as

$$x_{k+1} \in R(x_k) = \{r(x_k, u) \mid u \in U(x_k)\}.$$

Therefore, impulse differential inclusions can be thought of as hybrid control systems, with controls both on the continuous evolutions and on the discrete transitions.

Impulse differential inclusions can be used to describe hybrid phenomena. As for the executions of a hybrid automaton, the state of an impulse differential inclusion evolves over hybrid time sets.

Definition C.2 (Run of an Impulse Differential Inclusion) *a run of an impulse differential inclusion, $H = (X, F, R, J)$, is a hybrid trajectory, (τ, x), consisting of a hybrid time set $\tau = \{I_i\}_{i=0}^N$ and a sequence of maps $x = \{x_i\}_{i=0}^N$, $x_i(\cdot) : I_i \to X$, that satisfies:*

- Discrete Evolution*: for all i, $x_{i+1}(\tau_{i+1}) \in R(x_i(\tau_i'))$*
- Continuous Evolution*: if $\tau_i < \tau_i'$, $x_i(\cdot)$ is a solution to the differential inclusion $\dot{x} \in F(x)$ over the interval $[\tau_i, \tau_i']$ starting at $x_i(\tau_i)$, with $x_i(t) \notin J$ for all $t \in [\tau_i, \tau_i'[$.*

Recall that a *solution* to the differential inclusion $\dot{x} \in F(x)$ over an interval $[0,T]$ starting at $x_0 \in X$ is an absolutely continuous function $x : [0,T] \to X$, such that $x(0) = x_0$ and almost everywhere $\dot{x}(t) \in F(x(t))$. As for hybrid automata, runs of impulse differential inclusions can be classified into finite, infinite, Zeno, etc. (cf. Definition 3.6).

Definition C.2 dictates that, along a run the state can evolve continuously according to the differential inclusion $\dot{x} \in F(x)$ until the set J is reached. Moreover, whenever $R(x) \neq \emptyset$, a discrete transition from state x to some state in $R(x)$ may take place. In other words R *enables* discrete transitions (transitions may happen when $R(x) \neq \emptyset$ but do not have to), while J *forces* discrete transitions (transitions must happen when $x \in J$).

Notice that if at a state $x \in X$ a transition must happen ($x \in J$) but is not able to ($R(x) = \emptyset$) the system *blocks*, in the sense that there does not exist a run of the impulse differential inclusion starting at x (other than the trivial run ($[0,0], x$)). This is similar to the blocking situations (lack of runs) that we encountered for hybrid automata. To prevent such behaviour we introduce the following assumption.

Assumption C.3 *An impulse differential inclusion (X, F, R, J) is said to satisfy* Assumption C.3 *if $J \subseteq R^{-1}(X)$ and, if J is open (hence $I = X \setminus J$ is closed), $F(x) \cap T_I(x) \neq \emptyset$, for all $x \in I \setminus R^{-1}(X)$.*

As an example, consider again the bouncing ball system. The vertical motion of the ball can be captured by an impulse differential inclusion, $H_B = (X_B, F_B, R_B, J_B)$ with two state variables, the height of the ball, x_1 and its velocity in the vertical direction, x_2. Therefore, $X_B = \mathbb{R}^2$ and

$$\begin{aligned} F_B(x_1, x_2) =& (x_2, -g) \\ R_B(x_1, x_2) =& \begin{cases} (x_1, -cx_2) & \text{if } x_1 \leq 0 \text{ and } x_2 \leq 0 \\ \emptyset & \text{otherwise} \end{cases} \\ J_B =& \{x \in X_B \mid x_1 \leq 0 \text{ and } x_2 \leq 0\}, \end{aligned}$$

where g represents the acceleration due to gravity and $c^2 \in [0,1]$ the fraction of energy lost with each bounce.

C.3 Viability and Invariance Definitions

For impulse differential inclusions, reachability questions can be characterised by viability constraints.

Definition C.4 (Viable Run) *a run, (τ, x) of an impulse differential inclusion, $H = (X, F, R, J)$, is called viable in a set $K \subseteq X$ if $x(t) \in K$ for all $I_i \in \tau$ and all $t \in I_i$.*

Notice that the definition of a viable run requires the state to remain in the set K throughout, along continuous evolution up until and including the

state before discrete transitions, as well as after discrete transitions. Based on the notion of a viable run, one can define two different classes of sets.

Definition C.5 (Viable and Invariant Set) *A set $K \subseteq X$ is called viable under an impulse differential inclusion, $H = (X, F, R, J)$, if for all $x_0 \in K$ there exists an infinite run starting at x_0 that is viable in K. K is called invariant under the impulse differential inclusion, if for all $x_0 \in K$ all runs starting at x_0 are viable in K.*

In the cases where an impulse differential inclusion fails to satisfy a given viability or invariance requirement, one would like to establish sets of initial conditions (if any) for which the requirement will be satisfied. This notion can be characterised in terms of viability and invariance kernels.

Definition C.6 (Viability and Invariance Kernel) *The viability kernel, $Viab_H(K)$ of a set $K \subseteq X$ under an impulse differential inclusion $H = (X, F, R, J)$ is the set of states $x_0 \in X$ for which there exists an infinite run viable in K. The invariance kernel, $Inv_H(K)$ of $K \subseteq X$ under $H = (X, F, R, J)$ is the set of states $x_0 \in X$ for which all runs are viable in K.*

Notice that by definition $\mathrm{Viab}_H(K) \subseteq K$ and $\mathrm{Inv}_H(K) \subseteq K$, but in general the two sets are incomparable.

To state viability and invariance results for impulse differential inclusions we need to introduce some technical definitions from set valued analysis. For more details see [114, 147].

For a set valued map $R : X \to 2^Y$ and a set $K \subseteq Y$ we use $R^{-1}(K)$ to denote the *inverse image* of K under R and $R^{\ominus 1}(K)$ to denote the *extended core* of K under R, defined by

$$R^{-1}(K) = \{x \in X \mid R(x) \cap K \neq \emptyset\}, \quad \text{and}$$
$$R^{\ominus 1}(K) = \{x \in X \mid R(x) \subseteq K\} \cup \{x \in X \mid R(x) = \emptyset\}.$$

Notice that $R^{-1}(Y)$ is the set of $x \in X$ such that $R(x) \neq \emptyset$. We call the set $R^{-1}(Y)$ the *domain* of R and the set $\{(x, y) \in X \times Y \mid y \in R(x)\}$ the *graph* of R.

A set valued map $R : X \to 2^X$ is called *upper semicontinuous at $x \in X$* if for every $\epsilon > 0$ there exists $\delta > 0$ such that

$$\forall\, x' \in B(x, \delta), \;\; R(x') \subseteq B(R(x), \epsilon).$$

R is called *lower semicontinuous at $x \in X$* if for all $x' \in R(x)$ and for all sequences x_n converging to x, there exists a sequence $x'_n \in R(x_n)$ converging to x'. R is called *upper semicontinuous* (respectively *lower semicontinuous*) if it is upper semicontinuous (respectively lower semicontinuous) at all $x \in X$. It should be noted that, unlike single valued functions, these two notions of continuity are not equivalent for set valued maps. It can be shown that if R is upper semicontinuous with closed domain and $K \subseteq X$ is

a closed set, then $R^{-1}(K)$ is closed, whereas if R is lower semicontinuous and $U \subseteq X$ is an open set, then $R^{-1}(U)$ is open. Notice that the last statement also implies that if R is lower semicontinuous and $K \subseteq X$ is closed, $R^{\ominus 1}(K)$ is closed, since $R^{\ominus 1}(K) = X \setminus R^{-1}(X \setminus K)$.

For a closed subset, $K \subseteq X$, of a finite dimensional vector space, and a point $x \in K$, we use $T_K(x)$ to denote the *contingent cone* to K at x, i.e. the set of $v \in X$ such that there exists a sequence of real numbers $h_n > 0$ converging to 0 and a sequence of $v_n \in X$ converging to v satisfying

$$\forall\, n \geq 0,\ \ x + h_n v_n \ \in\ K.$$

Notice that, if x is in the interior of K, $T_K(x) = X$.

Subsequently we will be dealing with differential inclusions of the form $\dot{x} \in F(x)$, where $F : X \to 2^X$. To ensure existence of solutions we will need to impose some standard regularity assumptions on the map F, for example require F to be Marchaud and/or Lipschitz. We say that a map $F : X \to 2^X$ is *Marchaud* if and only if

1. the graph and the domain of F are nonempty and closed;
2. for all $x \in X$, $F(x)$ is convex, compact and nonempty; and,
3. the growth of F is linear, that is there exists $c > 0$ such that for all $x \in X$

$$\sup\{\|v\| \mid v \in F(x)\} \ \leq\ c(\|x\| + 1).$$

We say F is *Lipschitz* if and only if there exists a constant $\lambda > 0$ (known as the *Lipschitz constant*) such that for all $x, x' \in X$

$$F(x) \subseteq F(x') + \lambda \|x - x'\| B(0, 1).$$

C.4 Viability Conditions

The viability conditions for impulse differential inclusions involve the notion of "viability with target". Viability with target provides conditions under which solutions of $\dot{x} \in F(x)$ that remain in a set K until they reach a target set C exist.

Lemma C.7 *Consider a Marchaud map $F : X \to 2^X$ and two closed sets $K \subseteq X$ and $C \subseteq X$. For all $x_0 \in K$, there exists a solution of $\dot{x} \in F(x)$ starting at x_0 which is either*

1. *defined over $[0, \infty[$ with $x(t) \in K$ for all $t \geq 0$, or*
2. *defined over $[0, T]$ for some $T \geq 0$, with $x(T) \in C$ and $x(t) \in K$ for all $t \in [0, T]$,*

if and only if for all $x \in K \setminus C$, $F(x) \cap T_K(x) \neq \emptyset$.

Proof:

Necessity: Consider $x_0 \in K \backslash C$ and $x(\cdot)$ a trajectory starting from x_0 which stays in K on some interval $[0, \sigma]$ (and which does not reach C in this time interval). By application of Proposition 3.4.1 of [114], we obtain

$$F(x_0) \cap T_K(x_0) \neq \emptyset.$$

Sufficiency: Let $x_0 \in K \backslash C$. Because C is closed, some $r > 0$ exists such that $\overline{B(x_0, r)} \cap C \neq \emptyset$. In the set $B_K(x_0, r) := K \cap B(x_0, r)$, one can imitate the proof of Proposition 3.4.2 of [114] and obtain the existence of $T > 0$ and of a solution to $\dot{x} \in F(x)$ starting at x_0 which remains in $B_K(x_0, r)$ on $[0, T]$.

Using an argument (Zorn's Lemma) classical in differential equation theory, it is possible to extend $x(\cdot)$ to a maximal trajectory - again denoted $x(\cdot)$ - on some $[0, \tilde{T}]$ viable in K and such that $C \cap [0, \hat{T}) = \emptyset$. Either $\hat{T} = +\infty$ and the proof is complete, or $\hat{T} < +\infty$ and then $x(\hat{T}) \in C$ (if not one could extend a little the trajectory to a viable one, this would be a contradiction with the maximality of $x(\cdot)$). ∎

The conditions characterising viable sets depend on whether the set J is open or closed. In the case where J is closed we have the following.

Theorem C.8 (Viability Conditions, J Closed) *Consider an impulse differential inclusion $H = (X, F, R, J)$ such that F is Marchaud, R is upper semicontinuous with closed domain and J is closed. A closed set $K \subseteq X$ is viable under H if and only if*

1. $K \cap J \subseteq R^{-1}(K)$, *and*
2. $\forall\, x \in K \backslash R^{-1}(K)$, $F(x) \cap T_K(x) \neq \emptyset$

In words, the conditions of the theorem require that for any state $x \in K$, whenever a discrete transition has to take place ($x \in K \cap J$), a transition back into K is possible ($R(x) \cap K \neq \emptyset$), and whenever a discrete transition to another point in K is not possible ($R(x) \cap K = \emptyset$) continuous evolution that remains in K has to be possible (encoded by the local viability condition $F(x) \cap T_K(x) \neq \emptyset$).

Proof: Notice that, since R is upper semicontinuous with closed domain and K is closed, $R^{-1}(K)$ is also closed.

Necessity: Assume that K is viable under (X, F, R, J) and consider an arbitrary $x_0 \in K$. To show the first condition is necessary assume $x_0 \in K \cap J$. Then continuous evolution is impossible at x_0. Assume, for the sake of contradiction, that $x_0 \notin R^{-1}(K)$. Then either $R(x) = \emptyset$ (in which case the system blocks and no infinite runs start at x_0) or all runs starting at x_0 leave K through a discrete transition to some $x_1 \in R(x_0)$. In either case, the assumption that K is viable is contradicted. To show the second condition is necessary, assume $x_0 \in K \setminus R^{-1}(K)$. Since an infinite run viable in K starts at x_0, there exists a solution to the differential inclusion $\dot{x} \in F(x)$ starting at x_0 which is either

1. defined on $[0, \infty[$ with $x(t) \in K \setminus J$ for all $t \geq 0$; or,
2. defined on $[0, t']$ with $x(t') \in R^{-1}(K)$ and $x(t) \in K \setminus J$ for all $t \in [0, t'[$.

This implies, in particular, that there is a solution to the differential inclusion $\dot{x} \in F(x)$ starting at x_0 which is either

1. defined on $[0, \infty[$ with $x(t) \in K$ for all $t \geq 0$; or,
2. defined on $[0, t']$ with $x(t') \in R^{-1}(K)$ and $x(t) \in K$ for all $t \in [0, t']$.

By the necessary part of Lemma C.7, this implies that for all $x_0 \in K \setminus R^{-1}(K)$, $F(x) \cap T_K(x) \neq \emptyset$.

Sufficiency: Assume the conditions of the theorem are satisfied and consider an arbitrary $x_0 \in K$. We construct an infinite run of (X, F, R, J) starting at x_0 and viable in K by induction. We distinguish two cases, $x_0 \in K \setminus R^{-1}(K)$ and $x_0 \in K \cap R^{-1}(K)$. In the first case, by the sufficient part of Lemma C.7, there exists a solution to the differential inclusion $\dot{x} \in F(x)$ starting at x_0 which is either

1. defined on $[0, \infty[$ with $x(t) \in K$ for all $t \geq 0$; or,
2. defined on $[0, t']$ with $x(t') \in R^{-1}(K)$ and $x(t) \in K$ for all $t \in [0, t']$.

Notice that, since by the first assumption of the theorem, $K \cap J \subseteq R^{-1}(K)$ there must also be a solution to the differential inclusion $\dot{x} \in F(x)$ starting at x_0 which is either

1. defined on $[0, \infty[$ with $x(t) \in K \setminus J$ for all $t \geq 0$; or,
2. defined on $[0, t']$ with $x(t') \in R^{-1}(K)$ and $x(t) \in K \setminus J$ for all $t \in [0, t'[$

(i.e. either the solution stays in K forever and never reaches J, or the solution stays in K and reaches $R^{-1}(K)$ by the time it reaches J). In the former case, consider the infinite run $([0, \infty[, x)$; this is clearly a run of (X, F, R, J), viable in K. In the latter case, let $\tau_0 = 0$, $\tau_0' = t'$, and $\tau_1 = \tau_0'$. Since $x(\tau_0') \in R^{-1}(K)$, $x(\tau_1)$ can be chosen such that $x(\tau_1) \in K$. Notice that this argument also covers the case where $x_0 \in K \cap R^{-1}(K)$, with $x(\tau_0')$ playing the role of x_0. An infinite run viable in K can now be constructed inductively, by substituting x_0 by $x(\tau_1)$ and repeating the process. ∎

Similar conditions characterise viability when the set J is open, or, in other words, the set $I = X \setminus J$ is closed.

Theorem C.9 (Viability Conditions, J Open) *Consider an impulse differential inclusion $H = (X, F, R, J)$ such that F is Marchaud, R is upper semicontinuous with closed domain and J is open. A closed set $K \subseteq X$ is viable under H if and only if*

1. $K \cap J \subseteq R^{-1}(K)$, *and*
2. $\forall\, x \in (K \cap I) \setminus R^{-1}(K), \;\; F(x) \cap T_{K \cap I}(x) \neq \emptyset$

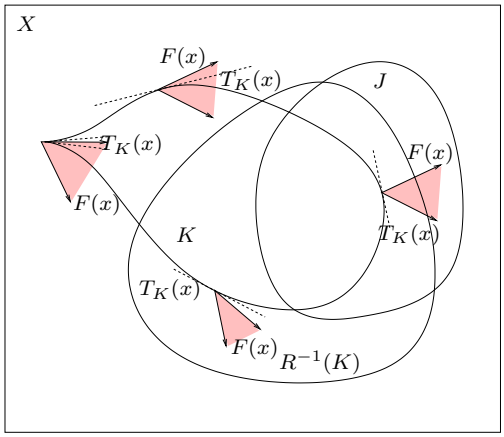

Figure C.1. K viable under $H = (X, F, R, J)$

Figure C.1 suggests how the conditions of Theorems C.8 and C.9 can be interpreted pictorially.

Notice that Assumption C.3 does not need to be added explicitly to Theorems C.8 and C.9, since the part of it that is essential to guarantee the existence of a run viable in K is implied by the conditions of the theorems. Conditions that guarantee the existence of runs for impulse differential inclusions can be deduced as a corollary of Theorems C.8 and C.9.

Corollary C.10 *Consider an impulse differential inclusion $H = (X, F, R, J)$ such that F is Marchaud, and R is upper semicontinuous with closed domain and J is either open or closed. Every finite run of H can be extended to an infinite run if and only if H satisfies Assumption C.3.*

C.5 Invariance Conditions

The conditions for invariance make use of the notion of "invariance with target" for continuous differential inclusions. Invariance with target involves conditions ensuring that all solutions of $\dot{x} \in F(x)$ remain in a set K until they reach a target set, C.

Lemma C.11 *Consider a Marchaud and Lipschitz map $F : X \to 2^X$ and two closed sets K and C. All solutions of $\dot{x} \in F(x)$ starting at some $x_0 \in K$ are either*

1. *defined over $[0, \infty[$ with $x(t) \in K$ for all $t \geq 0$, or*
2. *defined over $[0, T]$ with $x(T) \in C$ and $x(t) \in K$ for all $t \in [0, T]$,*

if and only if for all $x \in K \setminus C$, $F(x) \subseteq T_K(x)$.

Proof:

Necessity: Assume that all solutions starting in K stay in K until they reach C. Consider $x_0 \in K \setminus C$ and $v_0 \in F(x_0)$. Then (see for example [114] Corollary 5.3.2) there exists a trajectory $x(\cdot)$ of $\dot{x} \in F(x)$ starting at x_0 such that $\frac{d}{dt}x(0) = v_0$. Since x is a solution to $\dot{x} \in F(x)$ it remains in K until it reaches C. But $x_0 \in K \setminus C$ and C is closed, therefore, there exists $\alpha > 0$ such that $x(t) \in K$ for all $t \in [0, \alpha]$. Since x is absolutely continuous, for all $t \in [0, \alpha[$ where $\frac{d}{dt}x(t)$ is defined, $\frac{d}{dt}x(t) \in T_K(x(t))$ (see for example [114]). In particular, for $t = 0$, $v_0 = \frac{d}{dt}x(0) \in T_K(x_0)$. Hence, for all $x_0 \in K \setminus C$ and for all $v_0 \in F(x_0)$, $v_0 \in T_K(x_0)$, or, in other words, $F(x_0) \subseteq T_K(x_0)$.

Sufficiency: Let λ be the Lipschitz constant of F. Consider $x_0 \in K$ and a solution $x(\cdot)$ of $\dot{x} \in F(x)$ starting at x_0, and show that x remains in K until it reaches C. If $x_0 \in C$ then there is nothing to prove. If $x_0 \in K \setminus C$ consider

$$\theta = \sup\{t \mid \forall\, t' \in [0, t[,\ x(t') \in K \setminus C\}.$$

If $\theta = \infty$ or $x(\theta) \in C$ we are done. We show that $x(\theta) \in K \setminus C$ leads to a contradiction. Indeed, consider $\alpha > 0$ such that $B(x(\theta), \alpha) \cap C = \emptyset$ (which exists since $x(\theta) \notin C$ and C is closed), and $\theta' > \theta$, such that for all $t \in [\theta, \theta']$, $x(t) \in B(x(\theta), \alpha)$ (which exists since x is continuous). For $t \in [\theta, \theta']$, let $\Pi_K(x(t))$ denote a point of $B(x(\theta), \alpha) \cap K$ such that

$$d(x(t), K) = d(x(t), \Pi_K(x(t)))$$

(a projection of $x(t)$ onto K). Then (see for example [114], Lemma 5.1.2) for almost every $t \in [\theta, \theta']$,

$$\begin{aligned}
\frac{d}{dt}d(x(t), K) \leq & d\left(\frac{d}{dt}x(t), T_K(\Pi_K(x(t)))\right) \\
\leq & d\left(\frac{d}{dt}x(t), F(\Pi_K(x(t)))\right) \\
\leq & d\left(\frac{d}{dt}x(t), F(x(t))\right) + \lambda d(x(t), \Pi_K(x(t))) \\
\leq & 0 + d(x(t), K)
\end{aligned}$$

since x is a solution to $\dot{x} \in F(x)$ and by definition of Π. By the Gronwall lemma, $d(x(t), K) = 0$ for all $t \in [\theta, \theta']$, which contradicts the definition of θ. Summarising, if $F(x) \subseteq T_K(x)$ for all $x \in K \setminus C$, then all solutions starting in K either stay for ever in $K \setminus C$ or reach C before they leave K.

■

Lemma C.11 allows us to prove the following invariance theorem for impulse differential inclusions.

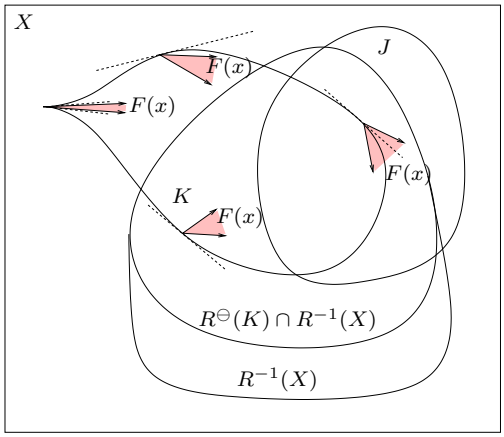

Figure C.2. K invariant under (X, F, R, J)

Theorem C.12 (Invariance Conditions) *Consider an impulse differential inclusion $H = (X, F, R, J)$ such that F is Marchaud and Lipschitz and J is closed. A closed set $K \subseteq X$ is invariant under H if and only if*

1. $R(K) \subseteq K$, *and*
2. $\forall\, x \in K \setminus J, \;\; F(x) \subseteq T_K(x)$.

In words, the conditions of the theorem require that for all $x \in K$, if a discrete transition is possible ($x \in R^{-1}(X)$), then all states after the transition are also in K ($R(x) \subseteq K$), whereas if continuous evolution is possible ($x \notin J$) then all possible solutions of the differential inclusion $\dot{x} \in F(x)$ remain in K (characterised here by the invariance condition $F(x) \subseteq T_K(x)$). Figure C.2 suggests how the conditions of Theorem C.12 can be interpreted pictorially.

Proof:

Necessity: Assume that K is invariant under (X, F, R, J). If the first condition is violated, then there exists $x_0 \in K$ and $x_1 \in R(x_0)$ with $x_1 \notin K$. Therefore, there exists a run starting at x_0 that leaves K through a discrete transition to some x_1 and the assumption that K is invariant is contradicted. To show the second condition is necessary, notice that since all runs of (X, F, R, J) starting in K are viable in K, then all solutions to $\dot{x} \in F(x)$ starting in K are either

1. defined on $[0, \infty[$ with $x(t) \in K \setminus J$ for all $t \geq 0$; or,
2. defined on $[0, t']$ with $x(t') \in J$ and $x(t) \in K$ for all $t \in [0, t']$.

Otherwise, there would exist a solution of $\dot{x} \in F(x)$ which leaves K before reaching J. This solution would be a run of (X, F, R, J) that is not viable in K, which would contradict the assumption that K is invariant. By the

necessary part of Lemma C.11, 1 and 2 imply that for all $x_0 \in K \setminus J$, $F(x) \subseteq T_K(x)$.

Sufficiency: Assume the conditions of the theorem are satisfied and consider an arbitrary $x_0 \in K$ and an arbitrary run, (τ, x), of (X, F, R, J) starting at x_0. Notice that $x(\tau_0) = x_0 \in K$ by assumption. Assume $x(\tau_i) \in K$ and show $x(t) \in K$ until τ_{i+1}; the claim then follows by induction. If $t = \tau_i$ we are done. If $\tau_i \prec t \preceq \tau_i'$, then $x(\tau_i) \in K \setminus J$ since continuous evolution is possible from $x(\tau_i)$. By the second condition of the theorem and the sufficient part of Lemma C.11, all solutions to the differential inclusion $\dot{x} \in F(x)$ starting at $x(\tau_i)$ are either

1. defined on $[0, \infty[$ with $x(t) \in K$ for all $t \geq 0$; or,
2. defined on $[0, t']$ with $x(t') \in J$ and $x(t) \in K$ for all $t \in [0, t']$.

In the first case, the run is viable in K and we are done. In the second case, $\tau_i' \preceq t'$ and therefore for all $t \in [\tau_i, \tau_i']$, $x(t) \in K$. If $x(\tau_i') \in R^{-1}(K)$, $x(\tau_{i+1}) \in R(x(\tau_i)) \subseteq K$ by the first condition of the theorem. If, on the other hand, $x(\tau_i') \in J$, but $R(x(\tau_i')) = \emptyset$, then the execution blocks at τ_i, and therefore is viable in K. ∎

Notice that no assumptions need to be imposed on R. Strictly speaking, Theorem C.12 remains true even without Assumption C.3; if the impulse differential inclusion has no runs for certain initial conditions in K, then, vacuously, all runs that start at these initial conditions are viable in K. In practice, it may be prudent to impose Assumption C.3, to ensure the results are meaningful.

C.6 Viability Kernels

If K is not viable under an impulse differential inclusion H, one would like to characterise the largest subset of K which is viable under H. This set turns out to be the viability kernel of K under the impulse differential inclusion. The viability kernel of an impulse differential inclusion can be characterised in terms of the notion of the viability kernel with target for a continuous differential inclusion. For a differential inclusion $\dot{x} \in F(x)$, the viability kernel of a set K with target C, $\mathrm{Viab}_F(K, C)$, is defined as the set of states for which there exists a solution to the differential inclusion that remains in K either forever, or until it reaches C.

Lemma C.13 *Consider a Marchaud map $F : X \to 2^X$ and two closed subsets of X, K and C. $Viab_F(K, C)$ is the largest closed subset of K satisfying the conditions of Lemma C.7.*

Proof: Let $D \subseteq K$ a closed set satisfying assumptions of Lemma C.7. Clearly $D \subseteq \mathrm{Viab}_F(K, C)$.

We claim that $\mathrm{Viab}_F(K,C)$ is closed. Consider a sequence $x_n \in \mathrm{Viab}_F(K,C)$ converging to some $x \in X$. Since K is closed, $x \in K$. We show that $x \in \mathrm{Viab}_F(K,C)$. If $x \in C$, the proof is done. Else, there exists an $r > 0$ with $K \cap B(x,r) \neq \emptyset$. For n large enough $x_n \in B(x, \frac{r}{2})$. For any such n, consider $x_n(\cdot)$ a solution to the differential inclusion starting from x_n, viable in K until it reaches C. Such a solution exists, since $x_n \in \mathrm{Viab}_F(K,C)$.

The graph of the solution map of the differential inclusion restricted to the compact set

$$\{x\} \cup \{x_n,\ n > 0\}.$$

is compact (Theorem 3.5.2 in [114]). Hence, there exists a subsequence to $x_n(\cdot)$ - again denoted $x_n(\cdot)$ - converging to a solution $x(\cdot)$ of the differential inclusion starting at x uniformly on compact intervals.

Let $\sigma > 0$ such that $x[0,\sigma) \cap C = \emptyset$. Such a σ exists since $x \notin C$, C is closed, and $x(\cdot)$ is continuous. Fix $0 \leq t < \sigma$. For n large enough, $x_n[0,t] \cap C = \emptyset$ because C is closed and $x_n(\cdot)$ converges uniformly to $x(\cdot)$ on $[0,t]$. Since $x_n[0,t]$ is contained in K so is $x[0,t]$. Because σ and t are arbitrary, we can deduce that $x(\cdot)$ is viable in K until it reaches C. So $x \in \mathrm{Viab}_F(K,C)$, and therefore $\mathrm{Viab}_F(K,C)$ is closed.

It remains to prove that $\mathrm{Viab}_F(K,C)$ satisfies conditions of Lemma C.7 (i.e., that it is itself viable with target C). Let $x_0 \in \mathrm{Viab}_F(K,C)$. By the very definition of the viability kernel some trajectory $x(\cdot)$ starting from x exists which is viable in K until it reaches C. Suppose by contradiction that some $s > 0$ exists such $x(s) \notin \mathrm{Viab}_F(K,C)$ and $x[0,s] \cap C = \emptyset$. Then any trajectory starting from $x(s)$ leaves K before reaching C. But $t \mapsto x(s+t)$ is such a trajectory which is viable in K until it reaches C, a contradiction. ■

Exercise C.1 *Show that* $K \cap C \subseteq Viab_F(K,C) \subseteq K$.

Using this notion, one can give an alternative characterisation of the sets that are viable under an impulse differential inclusion, as fixed points of an appropriate operator. For an impulse differential inclusion $H = (X,F,R,J)$, consider the operator $\mathrm{Pre}_H^{\exists} : 2^X \to 2^X$ defined by

$$\mathrm{Pre}_H^{\exists}(K) = \mathrm{Viab}_F(K \cap I, R^{-1}(K)) \cup (K \cap R^{-1}(K))$$

Recall that $I = X \setminus J$.

Lemma C.14 *Consider an impulse differential inclusion* $H = (X,F,R,J)$ *such that* F *is Marchaud,* R *is upper semicontinuous with closed domain, and* J *is open. A closed set* $K \subseteq X$ *is viable under* H *if and only if it is a fixed point of the operator* $\mathrm{Pre}_H^{\exists}$.

Proof:

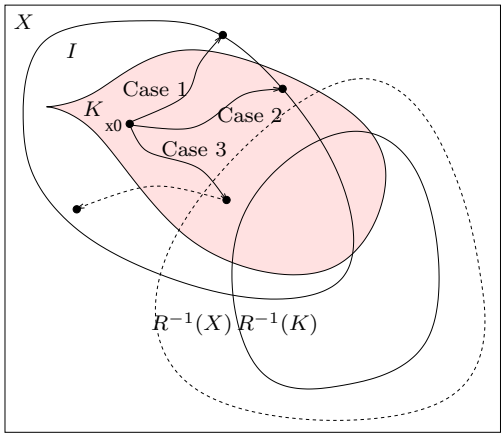

Figure C.3. Three possible evolutions for $x_0 \notin \mathrm{Viab}_F(K \cap I, R^{-1}(K)) \cup (K \cap R^{-1}(K))$.

Necessity: We first show that for every closed set K viable under $H = (X, F, R, J)$, $\mathrm{Pre}^{\exists}_H(K) = K$. $\mathrm{Pre}^{\exists}_H(K)$ is clearly a subset of K, since $\mathrm{Viab}_F(K \cap I, R^{-1}(K)) \subseteq K \cap I \subseteq K$. Conversely, consider an arbitrary $x_0 \in K$. Assume, for the sake of contradiction, that $x_0 \notin \mathrm{Viab}_F(K \cap I, R^{-1}(K)) \cup (K \cap R^{-1}(K))$. Consider an arbitrary infinite run (τ, x) viable in K and starting at x_0. Then $x(\tau_0) \notin R^{-1}(K)$ and $x(\tau_0) \notin \mathrm{Viab}_F(K \cap I, R^{-1}(K))$. If $\tau_0 = \tau'_0$, x starts by a discrete transition to some $x(\tau_1) \in R(x(\tau_0))$. Since $x(\tau_0) \notin R^{-1}(K)$, $x(\tau_1) \notin K$, which contradicts the assumption that (τ, x) is viable in K. If $\tau_0 < \tau'_0$, then (τ, x) starts by continuous evolution. Since $x_0 = x(\tau_0) \notin \mathrm{Viab}_F(K \cap I, R^{-1}(K))$, the run either

1. leaves K (at time $t \prec \tau'_0$) before it reaches $R^{-1}(K)$, or
2. leaves I (at time τ'_0) before it reaches $R^{-1}(K)$, or
3. takes a transition from some $x(\tau'_0) \in K \cap I \setminus R^{-1}(K)$

(see Figure C.3). The first case contradicts the assumption that (τ, x) is viable in K. In the remaining cases, $x(\tau'_0) \notin R^{-1}(K)$ and since $x(\tau_1) \in R(x(\tau'_0))$, we have $x(\tau_1) \notin K$. This also contradicts the assumption that (τ, x) is viable in K.

Sufficiency: Next, we show that every closed set K such that $K = \mathrm{Pre}^{\exists}_H(K)$ is viable. Consider an arbitrary $x_0 \in K$; we construct by induction an infinite run, (τ, x) that starts at x_0 and is viable in K. By assumption $x_0 = x(\tau_0) \in K$. Assume that we have constructed a run viable in K defined over a finite sequence $[\tau_0, \tau'_0], [\tau_1, \tau'_1], \ldots, [\tau_i, \tau_i]$. Since K is a fixed point of $\mathrm{Pre}^{\exists}_H$ and the run is viable in K, $x(\tau_i) \in \mathrm{Viab}_F(K \cap I, R^{-1}(K)) \cup (K \cap R^{-1}(K))$. If $x(\tau_i) \in K \cap R^{-1}(K)$, let $\tau'_i = \tau_i$

Algorithm 5 (Viability Kernel Approximation)

initialization: $K_0 = K$, $i = 0$
repeat
$K_{i+1} = Pre_H^{\exists}(K_i)$
$i = i + 1$
until $K_i = K_{i-1}$

Table C.1. Viability kernel approximation algorithm

and chose $x(\tau_{i+1}) \in R(x(\tau_i')) \cap K$. If $x(\tau_i) \in \text{Viab}_F(K \cap I, R^{-1}(K))$, then there exists a solution to the differential inclusion $\dot{x} \in F(x)$ which is either:

1. defined over $[0, \infty[$ with $x(t) \in K \cap I$ for all $t \geq 0$; or,
2. defined over $[0, t']$ with $x(t') \in R^{-1}(K)$ and $x(t) \in K \cap I$ for all $t \in [0, t']$.

In the former case, set $\tau_i' = \infty$ and the construction of the infinite run is complete. In the latter case, let $\tau_i' = \tau_i + t'$ and choose $x(\tau_{i+1}) \in R(x(\tau_i')) \cap K$. The claim follows by induction. ∎

Theorem C.15 (Viability Kernel) *Consider an impulse differential inclusion $H = (X, F, R, J)$ such that F is Marchaud, R is upper semicontinuous with closed domain and compact images, and J is open. The viability kernel of a closed set $K \subseteq X$ under H is the largest closed subset of K viable under H, that is, the largest closed fixed point of $Pre_H^{\exists}$ contained in K.*

The proof of Theorem C.15 makes use of the sequnce of sets generated by the algorithm given in Table C.1. The algorithm generates a sequence of nested, closed sets K_i that "converge" to the viability kernel. In addition to being useful in the proof of the theorem, the algorithm can therefore also be used to provide progressively better estimates of the viability kernel. This is, of course, provided one can compute $Pre^{\exists}(K_i)$ at each step. Numerical algorithms for approximating this computation have been developed see, for example, [148].

Proof: Let $K_\infty = \bigcap_{i=0}^{\infty} K_i$. We show that:

1. For every viable set $L \subseteq K$, $L \subseteq \text{Viab}_H(K)$.
2. K_∞ is closed.
3. $\text{Viab}_H(K) \subseteq K_\infty$.
4. $K_\infty \subseteq \text{Viab}_H(K)$.
5. $\text{Viab}_H(K)$ is viable.

Step 1: Every set $L \subseteq K$ which viable under $H = (X, F, R, J)$ must be contained in $\text{Viab}_H(K)$, since for all $x_0 \in L$ there exists an infinite run starting at x_0 that stays in L, and therefore in K.

Step 2: Since $\text{Viab}_F(K_i \cap I, R^{-1}(K_i)) \subseteq K_i \cap I \subseteq K_i$, $K_{i+1} \subseteq K_i$ for all i. Since K is closed, K_0 is closed. Moreover, if K_i is closed, then $R^{-1}(K_i)$ is closed (since R is upper semicontinuous with closed domain), and $\text{Viab}_F(K_i \cap I, R^{-1}(K_i))$ is closed (by Lemma C.13, since I and $R^{-1}(K_i)$ are closed), and, therefore, K_{i+1} is closed. By induction, K_i form a sequence of nested closed sets, and therefore K_∞ is closed (possibly the empty set).

Step 3: Consider a point $x_0 \in \text{Viab}_H(K)$ and show that $x_0 \in K_\infty$. Assume, for the sake of contradiction, that $x_0 \notin K_\infty$. Then there exists $N \geq 0$ such that $x_0 \notin K_N$. If $N = 0$, then $x_0 \notin K_0 = K$, therefore all runs starting at x_0 that are not viable in K (trivially). This contradicts the assumption that $x_0 \in \text{Viab}_H(K)$. If $N > 0$, we show that for all infinite runs (τ, x) starting at x_0 (which exist since $x_0 \in \text{Viab}_H(K)$), there exists a $t \preceq \tau_1$ such that[1] $x(t) \notin K_{N-1}$. The claim then follows by induction. Indeed, since $x_0 \notin K_N$ we must have have $x_0 \notin \text{Viab}_F(K_{N-1} \cap I, R^{-1}(K_{N-1})) \cup (K_{N-1} \cap R^{-1}(K_{N-1}))$. If $\tau_0 < \tau_0'$, then (τ, x) starts by continuous evolution. Since $x_0 = x(\tau_0) \notin \text{Viab}_F(K_{N-1} \cap I, R^{-1}(K_{N-1}))$, then all solutions to $\dot{x} \in F(x)$ either

1. leave K_{N-1} (at some $t \preceq \tau_0'$) before they reach $R^{-1}(K_{N-1})$, or
2. leave I (at time τ_0') before they reach $R^{-1}(K_{N-1})$, or
3. take a transition from some $x(\tau_0') \in (K_{N-1} \cap I) \setminus R^{-1}(K_{N-1})$.

(refer to Figure C.3). In the first case we are done. In the remaining cases, $x(\tau_0') \notin R^{-1}(K_{N-1})$ and since $x(\tau_1) \in R(x(\tau_0'))$, we have $x(\tau_1) \notin K_{N-1}$. The last argument also subsumes the case $\tau_0 = \tau_0'$, since $x_0 \notin K_{N-1} \cap R^{-1}(K_{N-1})$.

Step 4: Consider an arbitrary $x_0 \in K_\infty$. To show that $x_0 \in \text{Viab}_H(K)$, we construct an infinite run $(\tau, x) \in \mathcal{R}_H^\infty(x_0)$ viable in K. More specifically, since $x_0 \in K_k$ for all k, there exists a sequence of runs $(\tau^{(k)}, x^{(k)}) \in \mathcal{R}_H(x_0)$, which remain in K for at least k jumps. We will show that the sequence $(\tau^{(k)}, x^{(k)})$ has a cluster point $(\bar{\tau}, \bar{x}) \in \mathcal{R}_H^\infty(x_0)$, which is an infinite run of (X, F, R, J), starting at x_0, viable in K.

Let $[\tau_i^{(k)}, \tau_i^{(k)'}]$ (or $[\tau_i^{(k)}, \tau_i^{(k)'}[$ if i is the last interval) denote the sequence of intervals $\tau^{(k)}$. Recall that, without loss of generality, we can assume that $\tau_0^{(k)} = 0$ for all k. Let $\bar{\tau}_0 = 0$ and define

$$\bar{\tau}_0' = \lim \inf_{k \to \infty} \tau_0^{(k)'}.$$

[1] If $\tau = [\tau_0, \infty)$, $t \preceq \tau_1$ is replaced by $t \prec \tau_0' = \infty$.

Then there exists a subsequence of $\tau_0^{(k)'}$, denoted by $\tau_0^{\sigma(k)'}$, such that

$$\lim_{k\to\infty} \tau_0^{\sigma(k)'} = \bar{\tau}_0'.$$

We distinguish three cases:

1. $\bar{\tau}_0' = +\infty$;
2. $\bar{\tau}_0' \in]0, +\infty[$; and,
3. $\bar{\tau}_0' = 0$.

Case 1 will lead to a run $(\bar{\tau}, \bar{x}) \in \mathcal{R}_H^\infty(x_0)$ that is viable in K and makes no jumps. Case 2 will lead to a run $(\bar{\tau}, \bar{x}) \in \mathcal{R}_H^\infty(x_0)$ that is viable in K, whose first jump comes after an interval of continuous evolution. Finally, Case 3 will lead a run $(\bar{\tau}, \bar{x}) \in \mathcal{R}_H^\infty(x_0)$ viable in K, that takes its first jump immediately.

Case 1: Consider a sequence $y^{\sigma(k)}(\cdot)$ of solutions to the differential inclusion

$$\dot{x} \in F(x), \quad x(0) = x_0, \tag{C.1}$$

that coincide with $x^{\sigma(k)}$ on $[0, \tau_0^{\sigma(k)'}[$. Because the set of solutions of (C.1) is compact (see Theorem 3.5.2 of [114]), there exists a subsequence $y^{\phi(k)}(\cdot)$ of the sequence $y^{\sigma(k)}(\cdot)$ that converges to a solution $\bar{y}(\cdot)$ of (C.1). Moreover, since $\lim_{k\to\infty} \tau_0^{\sigma(k)'} = +\infty$, the sequence $y^{\phi(k)}(\cdot)$ (and hence the sequence $x^{\phi(k)}(\cdot)$) converges to $\bar{y}(\cdot)$ uniformly over $[0, T]$, for all $T > 0$.

Now, $(\tau^{\phi(k)}, x^{\phi(k)})$ is a run of (X, F, R, J) viable in K for at least k jumps. Therefore, $x^{\phi(k)}(t) \in K \cap I$ for all $t \in [0, \tau_0^{\phi(k)'}[$, and hence, for sufficiently large k, $x^{\phi(k)}(t) \in K \cap I$ for all $t \in [0, T]$. Since $K \cap I$ is closed, $\bar{y}(t) \in K \cap I$ for all $t \in [0, T]$. Since T is arbitrary, $([0, \infty[, \bar{y})$ is an infinite run of (X, F, R, J) (with no jumps) starting at x_0 and viable in K. The proof is complete.

Case 2: We can restrict attention to $k \geq 1$. As for case 1, define the sequence $y^{\sigma(k)}(\cdot)$ of solutions of (C.1) that coincide with $x^{\sigma(k)}$ on $[0, \tau_0^{\sigma(k)'}[$ and the subsequence $y^{\phi(k)}(\cdot)$ converging (uniformly over compact intervals) to a solution $\bar{y}(\cdot)$ of (C.1). As before, $(\tau^{\phi(k)}, x^{\phi(k)})$ is a run of (X, F, R, J) viable in K for at least $k > 0$ jumps. Therefore, $x^{\phi(k)}(t) \in K \cap I$ for all $t \in [0, \tau_0^{\phi(k)'}[$. Since $K \cap I$ is closed, $\bar{y}(t) \in K \cap I$ for all $t \in [0, \bar{\tau}_0']$. Therefore, $([\bar{\tau}_0, \bar{\tau}_0'], \bar{y})$ is a finite run of (X, F, R, J) (with no jumps) starting at x_0 and viable in K.

Since $y^{\phi(k)}(\cdot)$ converges to $\bar{y}(\cdot)$ and $\tau_0^{\phi(k)'}$ converges to $\bar{\tau}_0'$, $x^{\phi(k)}(\tau_0^{\phi(k)'})$ converges to $\bar{y}(\bar{\tau}_0')$. Recall that $(\tau^{\phi(k)}, x^{\phi(k)})$ is a run of (X, F, R, J) viable in K for at least $k > 0$ jumps, therefore $x^{\phi(k)}(\tau_1^{\phi(k)}) \in R(x^{\phi(k)}(\tau_0^{\phi(k)'})) \cap K$. Since R is upper semicontinuous with closed domain and compact images, there exists a subsequence of $x^{\phi(k)}(\tau_1^{\phi(k)})$ converging to some point $\bar{y}_1 \in R(\bar{y}(\bar{\tau}_0')) \cap K$. Therefore, $([0, \bar{\tau}_0'][\bar{\tau}_1, \bar{\tau}_1'], \bar{y})$ with $\bar{\tau}_1 = \bar{\tau}_1' = \bar{\tau}_0'$ and $\bar{y}(\bar{\tau}_1) = \bar{y}_1$

defined as above is a finite run of (X, F, R, J) (with one jump) starting at x_0 and viable in K.

Case 3: The second part of the argument for Case 2 shows that, since $x^{\phi(k)}(\tau_0^{\sigma(k)'})$ converge to x_0, there exists $\bar{y}_1 \in R(x_0) \cap K$. Therefore, $([0, \bar{\tau}_0'][\bar{\tau}_1, \bar{\tau}_1'], \bar{y})$ with $\bar{\tau}_0' = \bar{\tau}_1 = \bar{\tau}_1' = 0$, $\bar{y}(\tau_0') = x_0$ and $\bar{y}(\bar{\tau}_1) = \bar{y}_1$ is a finite run of (X, F, R, J) (with one instantaneous jump) starting at x_0 and viable in K.

To complete the proof for Cases 2 and 3, we repeat the argument starting at $\bar{y}(\bar{\tau}_1)$ (discarding the initial part of the sequences accordingly). We generate $\bar{\tau}_1' = \liminf_{k\to\infty} \tau_1^{(k)'}$ and construct a run of (X, F, R, J) viable in K, defined either over $[0, \bar{\tau}_0'][\bar{\tau}_1, \bar{\tau}_1'[$ (if $\bar{\tau}_1' = +\infty$, in which case the proof is complete) or over $[0, \bar{\tau}_0'][\bar{\tau}_1, \bar{\tau}_1'][\bar{\tau}_2, \bar{\tau}_2']$ with $\bar{\tau}_2 = \bar{\tau}_2' = \bar{\tau}_1'$ (if $\bar{\tau}_1'$ is finite). The claim follows by induction.

Step 5: Finally, we show $\mathrm{Viab}_H(K)$ is viable by showing that it is a fixed point of $\mathrm{Pre}_H^{\exists}$. Recall that $\mathrm{Pre}_H^{\exists}(\mathrm{Viab}_H(K)) \subseteq \mathrm{Viab}_H(K)$. Consider an arbitrary $x_0 \in \mathrm{Viab}_H(K)$ and assume, for the sake of contradiction, that $x_0 \notin \mathrm{Pre}_H^{\exists}(\mathrm{Viab}_H(K))$. Consider an arbitrary infinite run (τ, x) viable in K and starting at x_0 (which exists since $x_0 \in \mathrm{Viab}_H(K)$). If $\tau_0 = \tau_0'$, x starts by a discrete transition to some $x(\tau_1) \in R(x_0)$. Since $x_0 \notin R^{-1}(\mathrm{Viab}_H(K))$, $x(\tau_1) \notin \mathrm{Viab}_H(K)$. If $\tau_0 < \tau_0'$, then (τ, x) starts by continuous evolution. Since $x_0 \notin \mathrm{Viab}_F(\mathrm{Viab}_H(K) \cap I, R^{-1}(\mathrm{Viab}_H(K)))$, the execution either

1. leaves $\mathrm{Viab}_H(K)$ (at time $t \prec \tau_0'$) before it reaches $R^{-1}(\mathrm{Viab}_H(K))$; or,
2. leaves I (at time τ_0') before it reaches $R^{-1}(\mathrm{Viab}_H(K))$; or,
3. takes a transition from some $x(\tau_0') \in \mathrm{Viab}_H(K) \cap I \setminus R^{-1}(\mathrm{Viab}_H(K))$

(see Figure C.3). In all cases, (τ, x) either blocks or leaves $\mathrm{Viab}_H(K)$ at some $t \in \tau$ with $t \preceq \tau_1$. But if $x(t) \notin \mathrm{Viab}_H(K)$ there is no infinite run of $H = (X, F, R, J)$ starting at $x(t)$ and viable in K. Therefore, (τ, x) either blocks or is not viable in K. This contradicts the assumption that $x_0 \in \mathrm{Viab}_H(K)$. ∎

It should be stressed that the conditions of Theorem C.15 ensure that for all initial conditions in the viability kernel infinite runs of the impulse differential inclusion exist, but do not ensure that these runs will extend over an infinite time horizon; all runs starting at certain initial conditions in the viability kernel may turn out to be Zeno.

C.7 Invariance Kernels

If K is not invariant under an impulse differential inclusion H, one would like to characterise the largest subset of K which is invariant under H. This turns out to be the invariance kernel of K under the impulse differential

inclusion. The invariance kernel can be characterised using the notion of the invariance kernel with target for continuous differential inclusions. For a differential inclusion $\dot{x} \in F(x)$, the invariance kernel of a set K with target C, $\mathrm{Inv}_F(K, C)$ is defined as the set of states for which all solutions to the differential inclusion remain in K either for ever, of until they reach C.

Lemma C.16 *Consider a Marchaud and Lipschitz map $F : X \to 2^X$ and two closed subsets of X, K and C. $Inv_F(K, C)$ is the largest closed subset of K satisfying the conditions of Lemma C.11.*

Proof: By definition, $\mathrm{Inv}_F(K, C)$ is the set of $x_0 \in K$ such that for all solutions $x(\cdot)$ of $\dot{x} \in F(x)$ starting at x_0 either

1. $x(t) \in K$ for all $t \geq 0$, or,
2. there exists $t' \geq 0$ such that $x(t') \in C$ and $x(t) \in K$ for all $t \in [0, t']$.

Therefore, $\mathrm{Inv}_F(K, C)$ satisfies the conditions of Lemma C.11. Moreover, every subset $L \subseteq K$ which satisfies the conditions of Lemma C.11 must be contained in $\mathrm{Inv}_F(K, C)$, since all runs starting in L stay in L (and therefore in K) until they reach C.

It remains to show that $\mathrm{Inv}_F(K, C)$ is closed. Consider a sequence $x_n \in \mathrm{Inv}_F(K, C)$ converging to x_0 and show that $x_0 \in \mathrm{Inv}_F(K, C)$. Since by definition $\mathrm{Inv}_F(K, C) \subseteq K$ and K is assumed to be closed, $x_0 \in K$. If $x_0 \in K \cap C$ there is nothing to prove, since by definition $K \cap C \subseteq \mathrm{Inv}_F(K, C)$. If $x_0 \in K \setminus C$, let $x(\cdot)$ be any solution of $\dot{x} \in F(x)$ starting at x_0. Let

$$\theta = \sup\{t \mid \forall\, t' \in [0, t[,\ x(t') \in K \setminus C\}.$$

If $\theta = \infty$ or if $x(\theta) \in C$, then $x_0 \in \mathrm{Inv}_F(K, C)$, and the proof is complete.

Let λ be the Lipschitz constant of F, and assume, for the sake of contradiction, that $\theta < \infty$ and $x(\theta) \in K \setminus C$. Then, by the definition of θ and the assumption that K and C are closed, there exists $\theta' > \theta$ such that $x(\theta') \notin K$ and for all $t \in [\theta, \theta']$, $x(t) \notin C$. Choose ϵ such that

$$d(x(\theta'), K) > \epsilon e^{\lambda \theta'} \tag{C.2}$$

(possible since K is closed and $x(\theta') \notin K$) and for all $t \in [0, \theta']$

$$\left\{x(t) + \epsilon B(0, 1) e^{\lambda t}\right\} \cap C = \emptyset \tag{C.3}$$

(possible since C is closed and for all $t \in [0, \theta']$, $x(t) \notin C$).

Since $x_n \to x_0$ there exists n large enough such that $\|x_n - x_0\| < \epsilon$. By Filippovs theorem (see for example [114], Theorem 5.3.1) there exists a solution $x_n(\cdot)$ of $\dot{x} \in F(x)$ starting at x_n such that for all $t \in [0, \theta']$

$$\|x_n(t) - x(t)\| \leq \|x_n - x_0\| e^{\lambda t},$$

or, in other words, for all $t \in [0, \theta']$

$$x_n(t) \in B(x(t), \|x_n - x_0\| e^{\lambda t}) \subseteq B(x(t), \epsilon e^{\lambda t}).$$

Therefore, by equation (C.3), for all $t \in [0, \theta']$, $x_n(t) \notin C$, while, by equation (C.2) $x_n(\theta') \notin K$. This contradicts the assumption that $x_n \in \mathrm{Inv}_F(K, C)$. Hence, every converging sequence has its limit in $\mathrm{Inv}_F(K, C)$, and therefore $\mathrm{Inv}_F(K, C)$ is closed. ∎

Exercise C.2 *Show that* $K \cap C \subseteq Inv_F(K, C) \subseteq K$.

Using the notion of invariance kernel with target, one can give an alternative characterisation of the sets that are invariant under an impulse differential inclusion, as fixed points of an operator. Given an impulse differential inclusion $H = (X, F, R, J)$, consider the operator $\mathrm{Pre}^{\forall}_H : 2^X \to 2^X$ defined by

$$\mathrm{Pre}^{\forall}_H(K) = \mathrm{Inv}_F(K, J) \cap R^{\ominus 1}(K)$$

Lemma C.17 *Consider an impulse differential inclusion* $H = (X, F, R, J)$ *such that* F *is Marchaud and Lipschitz,* R *is lower semicontinuous, and* J *is closed. A closed set* $K \subseteq X$ *is invariant under* H *if and only if it is a fixed point of the operator* $\mathrm{Pre}^{\forall}_H$.

Proof:

Necessity: We first show that for every closed, invariant set K, $K = \mathrm{Pre}^{\forall}_H(K)$. Clearly $\mathrm{Pre}^{\forall}_H(K) \subseteq K$, since $\mathrm{Inv}_F(K, J) \subseteq K$. Conversely, consider an arbitrary point $x_0 \in K$ and show that $x_0 \in \mathrm{Inv}_F(K, J) \cap R^{\ominus 1}(K)$. Assume, for the sake of contradiction that this is not the case. Then, either $x_0 \notin \mathrm{Inv}_F(K, J)$, or $x_0 \notin R^{\ominus 1}(K)$. If $x_0 \notin R^{\ominus 1}(K)$, there exists $x_1 \in R(x_0)$ such that $x_1 \notin K$; in other words, there exists a run of the impulse differential inclusion starting at x_0 that leaves K by a discrete transition. This contradicts the assumption that K is invariant. If, on the other hand, $x_0 \notin \mathrm{Inv}_F(K, J)$ then, in particular, $x_0 \notin J \cap K$ (since $J \cap K \subseteq \mathrm{Inv}_F(K, J)$); but $x_0 \in K$, so we must have $x_0 \notin J$, and therefore continuous evolution starting at x_0 is possible. Since $x_0 \notin \mathrm{Inv}_F(K, J)$, there exists a solution to $\dot{x} \in F(x)$ starting at x_0 that leaves K before reaching J. This solution is a run (X, F, R, J) that starts in K but is not viable in K. This also contradicts the assumption that K is invariant.

Sufficiency: Next, we show that every closed set K such that $K = \mathrm{Pre}^{\forall}_H(K)$ is invariant. Consider an arbitrary run (τ, x) starting at some $x_0 \in K$. We show that (τ, x) is viable in K by induction. Assume that we have shown that $x(t) \in K$ for all $t \in [\tau_0, \tau_0'], [\tau_1, \tau_1'], \ldots, [\tau_i, \tau_i]$. Then, since $K = \mathrm{Pre}^{\forall}_H(K)$, $x(\tau_i) \in \mathrm{Inv}_F(K, J) \cap R^{\ominus 1}(K)$. If $\tau_i = \tau_i'$ the system takes a discrete transition to some $x(\tau_{i+1}) \in R(x(\tau_i')) \subseteq K$, since $x(\tau_i') = x(\tau_i) \in R^{\ominus 1}(K)$. If $\tau_i < \tau_i'$ the run progresses by continuous evolution. Since $x(\tau_i) \in \mathrm{Inv}_F(K, J)$, then either

1. $\tau_i' = \infty$ and $x(t) \in K$ for all $t \geq \tau_i$; or,
2. $\tau_i' < \infty$, $x(\tau_i') \in J$ and $x(t) \in K$ for all $t \in [\tau_i, \tau_i']$.

Algorithm 6 (Invariance Kernel Approximation)

initialisation: $K_0 = K$, $i = 0$
repeat
$K_{i+1} = \mathrm{P}re_H^{\forall}(K_i)$
$i = i + 1$
until $K_i = K_{i-1}$

Table C.2. Invariance kernel approximation algorithm

Notice that $x(\tau_i') \in K = \mathrm{P}re_H^{\forall}(K)$, and, in particular, $x(\tau_i') \in R^{\ominus 1}(K)$. Therefore, $x(\tau_{i+1}) \in R(x(\tau_i')) \subseteq K$. The claim follows by induction.

Notice that in the last argument $R(x(\tau_i'))$ may, in fact, be empty. In this case the run "blocks", in the sense that there exist no infinite runs starting at $x(\tau_i')$. The conclusion that all runs starting at x_0 are viable in K is still true however. To preclude this somewhat unrealistic situation, one can add Assumption C.3 to the lemma and subsequent Theorem C.18. ∎

Theorem C.18 (Invariance Kernel) *Consider an impulse differential inclusion $H = (X, F, R, J)$ such that F is Marchaud and Lipschitz, R is lower semicontinuous and J is closed. The invariance kernel of a closed set $K \subseteq X$ under H is the largest closed subset of K invariant under H, that is, the largest, closed fixed point of $\mathrm{P}re_H^{\forall}$ contained in K.*

Again the proof makes use of the sequence of sets generated by the algorithm given in Table C.2. At each step, the algorithm computes the set of states for which all solution of the differential inclusion $\dot{x} \in F(x)$ stay in the K_i until they reach J. K_{i+1} is then the subset of those states for which if a transition is possible, the state after the transition is also in K_i.

Proof: Let $K_\infty = \bigcap_{i=0}^{\infty} K_i$. We show that

1. For every invariant set $L \subseteq K$, $L \subseteq \mathrm{Inv}_H(K)$.
2. K_∞ is closed.
3. $\mathrm{Inv}_H(K) \subseteq K_\infty$.
4. $K_\infty = \mathrm{P}re_H^{\forall}(K_\infty)$.

Step 2, step 4 and Lemma C.17 imply that K_∞ is invariant. Therefore, by step 1, $K_\infty \subseteq \mathrm{Inv}_H(K)$, and, by step 3, $K_\infty = \mathrm{Inv}_H(K)$.

Step 1: Every set $L \subseteq K$ which invariant under (X, F, R, J) must be contained in $\mathrm{Inv}_H(K)$, since all runs starting in L stay in L, and therefore in K.

Step 2: Clearly, for all i, $K_{i+1} \subseteq \mathrm{Inv}_F(K_i, J) \subseteq K_i$. Since K is closed, K_0 is closed. Moreover, if K_i is closed, then $\mathrm{Inv}_F(K_i, J)$ is closed (by Lemma C.16, since J is closed), $R^{\ominus 1}(K_i)$ is closed (since R is lower semicontinuous), and, therefore, K_{i+1} is closed. By induction, the K_i form a

sequence of nested closed sets, and therefore K_∞ is closed (or the empty set).

Step 3: Consider a point $x_0 \in \mathrm{Inv}_H(K)$ and show that $x_0 \in K_\infty$. Assume, for the sake of contradiction, that $x_0 \notin K_\infty$. Then there exists $N \geq 0$ such that $x \notin K_N$. If $N = 0$, then $x_0 \notin K_0 = K$, therefore there exists a (trivial) run starting at x_0 that is not viable in K. This contradicts the assumption that $x_0 \in \mathrm{Inv}_H(K)$. If $N > 0$, we show that there exists a run stating at x_0 that after at most one discrete transition finds itself outside K_{N-1}. The claim then follows by induction. Indeed, since $x_0 \notin K_N$ we must either have $x_0 \notin \mathrm{Inv}_F(K_{N-1}, J)$, or $x_0 \notin R^{\ominus 1}(K_{N-1})$. If $x_0 \notin R^{\ominus 1}(K_{N-1})$, there exists $x_1 \in R(x_0)$ such that $x_1 \notin K_{N-1}$, i.e., there exists a run starting at x_0 that transitions outside K_{N-1}. If, on the other hand, $x_0 \notin \mathrm{Inv}_F(K_{N-1}, J)$, then $x_0 \notin J \cap K_{N-1}$. Therefore either $x_0 \notin K_{N-1}$ (and the proof is complete), or $x_0 \notin J$ and continuous evolution is possible. In the latter case, since $x_0 \notin \mathrm{Inv}_F(K_{N-1}, J)$, by Lemma C.16 there exists a solution to $\dot{x} \in F(x)$ starting at x_0 that leaves K_{N-1} before reaching J. This solution is a run of (X, F, R, J) that leaves K_{N-1}.

Step 4: Recall that $\mathrm{Pre}^{\forall}_H(K_\infty) \subseteq K_\infty$. Consider an arbitrary $x_0 \in K_\infty$ and show that $x_0 \in \mathrm{Pre}^{\forall}_H(K_\infty)$. Assume, for the sake of contradiction, that $x_0 \notin \mathrm{Inv}_F(K_\infty, J) \cap R^{\ominus 1}(K_\infty)$. Then there exists a run (τ, x) starting at x_0 and a $t \preceq \tau_1$ such that[2] $x(t) \notin K_\infty$, or, in other words, there exists a run (τ, x), a $t \preceq \tau_1$ and a $N \geq 0$ such that $x(t) \notin K_N$. To see this notice that either $x(\tau_0) \notin R^{\ominus 1}(K_\infty)$ (in which case we can take $\tau_0' = \tau_0$, $x(\tau_1) \notin K_\infty$ and $t = \tau_1$) or $x(\tau_0) \notin \mathrm{Inv}_F(K_\infty, J)$ (in which case there exists a solution to $\dot{x} \in F(x)$ that leaves K before reaching J). The same argument, however, also shows that $x(\tau_0) = x_0 \notin K_{N+1}$, which contradicts the assumption that $x_0 \in K_\infty$. ∎

C.8 The Bouncing Ball Example

It is easy to check that F_B is both Marchaud and Lipschitz and that R_B is upper and lower semicontinuous and has closed domain. Moreover, H_B also satisfies Assumption C.3, since $R^{-1}(X) = J$. Therefore, we can immediately draw the following conclusion.

Proposition C.19 *Infinite runs exist for all $x_0 \in X_T$.*

The proposition suggests that the impulse differential inclusion H_B does not deadlock. However, it is easy to show that for all $x_0 \in X_T$ all infinite runs are Zeno.

[2] If $\tau = [\tau_0, \tau_0']$ or $\tau = [\tau_0, \tau_0'[$, $t \preceq \tau_1$ should be replaced by $t \preceq \tau_0$ or, respectively, $t \prec \tau_0$.

Despite the Zeno behaviour, H_B is in many ways a reasonable model of the bouncing ball system. For example, one can show that the ball never falls below the surface on which it bounces, and that the system dissipates energy.

Proposition C.20 *The set $K = \{x \in X_T \mid x_1 \geq 0\}$ is viable and invariant. For all $C > 0$ the set $L = \{x \in X_T \mid gx_1 + x_2^2/2 \leq C\}$ is invariant.*

For the first part, notice that $K \cap J_B = \{x \in X_T \mid x_1 = 0 \text{ and } x_2 \leq 0\}$. Since R_B does not affect x_1, $K \cap J_B \subseteq R^{-1}(K)$ and $R(K) \subseteq K$. Moreover, $K \setminus J_B = \{x \in X_T \mid x_1 > 0 \text{ or } x_1 = 0 \text{ and } x_2 > 0\}$. For x such that $x_1 > 0$, $F_B(x) \subseteq T_K(x) = X_B$. For x such that $x_1 = 0$ and $x_2 > 0$, $F_B(x) \subseteq \{v \in X \mid v_1 > 0\} = T_K(x)$. Therefore, K is viable by Theorem C.8 and invariant by Theorem C.12.

For the second part, R leaves x_1 unchanged and maps x_2 to αx_2. Therefore $R(L) \subseteq L$ since $\alpha \in [0, 1]$. Moreover

$$\begin{aligned} L \setminus J = &\{x \in X_T \mid x_1 > 0 \text{ or } x_2 > 0\} \\ &\cap \{x \in X_T \mid gx_1 + x_2^2/2 \leq C\} \end{aligned}$$

For $x \in L \setminus J$ such that $gx_1 + x_2^2/2 < C$, $F_B(x) \subseteq T_K(x) = X_B$. For $x \in L \setminus J$ such that $gx_1 + x_2^2/2 = C$,

$$\begin{aligned} T_K(x) &= \{v \in X_B \mid v_1 g + v_2 x_2 \leq 0\} \\ &\supseteq \{v \in X_B \mid v_1 g + v_2 x_2 = 0\} \supseteq F_B(x) \end{aligned}$$

The claim follows by Theorem C.12.

C.9 Bibliography and Further Reading

Reachability with inputs has also been studied in the context of optimal control and differential games [110, 105] and using ellipsoidal calculus [149, 150].

Other classes of control problems that have been studied for hybrid systems include supervisory control [151, 100, 152] and optimal control [46, 153].

Because the analytical study of the resulting equations is rarely possible computational tools have been developed to approximate the solutions numerically [21, 22, 154, 155, 138, 156, 139, 148].

C.10 problems

Problem C.1 *A thermostat system can be more accurately modelled by an impulse differential inclusion, $H_T = (X_T, F_T, R_T, J_T)$ with two state vari-*

ables, $x = (x_1, x_2)$: the current room temperature x_1 and the steady state toward which the temperature is converging x_2 (which of course depends on whether the heater is on or off). Let $X_T = \mathbb{R}^2$, and

$$\begin{aligned} F_T(x_1, x_2) =& ([a(x_1 - x_2), b(x_1 - x_2)], 0) \\ R_T(x_1, x_2) =& \begin{cases} (x_1, 30 - x_2) \\ \quad \textit{if } (x_1 \geq 21 \textit{ and } x_2 \geq 20) \\ \quad \textit{or } (x_1 \leq 19 \textit{ and } x_2 \leq 20) \\ \emptyset \quad \textit{otherwise} \end{cases} \\ J_T =& \{x \in X_T \mid (x_1 \geq 22 \textit{ and } x_2 \geq 20) \\ & \qquad \textit{or } (x_1 \leq 18 \textit{ and } x_2 \leq 20)\}, \end{aligned}$$

with $a \leq b < 0$. Show that

1. *Infinite executions exist for all $x_0 \in X_T$.*
2. *The set $K = \{x \in X_T \mid x_2 \in \{0, 30\}\}$ is invariant.*
3. *The set $L = \{x \in X_T \mid x_1 \in [19, 21]\}$ is viable. Is L invariant?*
4. *The set $M = \{x \in X_T \mid x_1 \in [18, 22]\}$ is invariant.*

References

[1] P. Varaiya, "Smart cars on smart roads: problems of control," *IEEE Transactions on Automatic Control*, vol. AC-38, no. 2, pp. 195–207, 1993.

[2] J. Lygeros, D.N. Godbole, and S.S. Sastry, "Verified hybrid controllers for automated vehicles," *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 522–539, April 1998.

[3] C.J. Tomlin, G.J. Pappas, and S.S. Sastry, "Conflict resolution for air traffic management: A case study in multi-agent hybrid systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 509–521, 1998.

[4] M. Vidyasagar, *Nonlinear Systems Analysis*, Prentice Hall, second edition, 1992.

[5] S. Wiggins, *Introduction to Applied Nonlinear Dynamical Systems and Chaos*, Springer-Verlag, third edition, 1997.

[6] H.B. Khalil, *Nonlinear Systems*, Prentice Hall, third edition, 2001.

[7] S.S. Sastry, *Nonlinear Systems: Analysis, Stability and Control*, Springer-Verlag, New York, 1999.

[8] J.E. Hopcroft, R. Motwani, and J.D. Ullman, *Introduction to automata theory, languages and computation*, Addison-Wesley Publishing, second edition, 2000.

[9] H.R. Lewis and C. Papadimitriou, *Elements of the Theory of Computation*, Prentice Hall, second edition, 1997.

[10] C.G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, Kluwer Academic Publishers, 1999.

[11] R. Alur and D. Dill, "A theory of timed automata," *Theoretical Computer Science*, vol. 126, pp. 183–235, 1994.

[12] R. Alur, C. Courcoubetis, T. A. Henzinger, and P. H. Ho, "Hybrid automaton: An algorithmic approach to the specification and verification of hybrid systems," in *Hybrid Systems*, Robert L. Grossman, Anil Nerode, Anders P. Ravn, and Hans Rischel, Eds., number 736 in LNCS, pp. 209–229. Springer-Verlag, Berlin, 1993.

[13] N. Lynch, R. Segala, F. Vaandrager, and H.B. Weinberg, "Hybrid I/O automata," in *Hybrid Systems III*, number 1066 in LNCS, pp. 496–510. Springer-Verlag, Berlin, 1996.

[14] O. Maler, A. Pnueli, and J. Sifakis, "On the synthesis of discrete controllers for timed systems," in *Theoretical Aspects of Computer Science*, Berlin, 1995, number 900 in LNCS, pp. 229–242, Springer-Verlag.

[15] H. Wong-Toi, "The synthesis of controllers for linear hybrid automata," in *IEEE Conference on Decision and Control*, San Diego, California, December 10–12 1997, pp. 4607–4613.

[16] T. A. Henzinger, P. H. Ho, and H. Wong Toi, "A user guide to HYTECH," in *TACAS 95: Tools and Algorithms for the Construction and Analysis of Systems*, E. Brinksma, W. Cleaveland, K. Larsen, T. Margaria, and B. Steffen, Eds., Berlin, 1995, number 1019 in LNCS, pp. 41–71, Springer-Verlag.

[17] R.W. Brockett, "Hybrid models for motion control systems," in *Perspectives in Control*, H.L. Trentelman and J.C. Willems, Eds., Boston, MA, 1993, Birkhäuser.

[18] A. Nerode and W. Kohn, "Models for hybrid systems: Automata, topologies, controllability, observability," in *Hybrid Systems*, Robert L. Grossman, Anil Nerode, Anders P. Ravn, and Hans Rischel, Eds., number 736 in LNCS, pp. 317–356. Springer-Verlag, Berlin, 1993.

[19] M. Lemmon, J. A. Stiver, and P. J. Antsaklis, "Event identification and intelligent hybrid control," in *Hybrid Systems*, Robert L. Grossman, Anil Nerode, Anders P. Ravn, and Hans Rischel, Eds., number 736 in LNCS, pp. 268–296. Springer-Verlag, Berlin, 1993.

[20] M. Heymann, F. Lin, and G. Meyer, "Control synthesis for a class of hybrid systems subject to configuration-based safety constraints," in *Hybrid and Real Time Systems*, number 1201 in LNCS, pp. 376–391. Springer-Verlag, Berlin, 1997.

[21] T. Dang and O. Maler, "Reachability analysis via face lifting," in *Hybrid Systems: Computation and Control*, S. Sastry and T.A. Henzinger, Eds., number 1386 in LNCS, pp. 96–109. Springer-Verlag, Berlin, 1998.

[22] M.R. Greenstreet and I. Mitchell, "Integrating projections," in *Hybrid Systems: Computation and Control*, S. Sastry and T.A. Henzinger, Eds., number 1386 in LNCS, pp. 159–174. Springer-Verlag, Berlin, 1998.

[23] A.S. Matveev and A.V. Savkin, *Qualitative theory of hybrid dynamical systems*, Birkhauser, Boston, MA, 2000.

[24] A.J. van der Schaft and H. Schumacher, *An Introduction to Hybrid Dynamical Systems*, Number 251 in Lecture Notes in Control and Information Sciences. Springer-Verlag, 1999.

[25] A. Pnueli and J. Sifakis Editors, "Hybrid systems," *Theoretical Computer Science*, vol. 138, no. 1, 1995.

[26] P.J. Antsaklis and A. Nerode, Editors, "Special issue on hybrid control systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 4, April 1998.

[27] J.M. Schumacher, A.S. Morse, C.C. Pandelides, and S. Sastry, Editors, "Special issue on hybrid systems," *Automatica*, vol. 35, no. 3, March 1999.

[28] A. Savkin, Editor, "Special issue on hybrid systems," *Systems and Control Letters*, vol. 38, no. 3, October 1999.

[29] P.J. Antsaklis, Editor, "Special issue on hybrid systems: Theory and applications," *Proceedings of the IEEE*, vol. 88, no. 7, July 2000.

[30] A. Balluchi, L. Benvenuti, M.D. Di Benedetto, C. Pinello, and A.L. Sangiovanni-Vincentelli, "Automotive engine control and hybrid systems: Challenges and opportunities," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 888–912, July 2000.

[31] B. Lennartsson, M. Tittus, B. Egardt, and S. Pettersson, "Hybrid systems in process control," *Control Systems Magazine*, vol. 16, no. 5, pp. 45–56, 1996.

[32] S. Engell, S. Kowalewski, C. Schulz, and O. Stursberg, "Continuous-discrete interactions in chemical processing plants," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 1050–1068, July 2000.

[33] R. Horowitz and P. Varaiya, "Control design of an automated highway system," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 913–925, July 2000.

[34] D.L. Pepyne and C.G. Cassandras, "Optimal control of hybrid systems in manufacturing," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 1108–1123, July 2000.

[35] R. May, *Stability and Complexity of Model Ecosystems*, Princeton University Press, Princeton, NJ, 1973.

[36] C.J. Tomlin, J. Lygeros, and S.S. Sastry, "Synthesizing controllers for nonlinear hybrid systems," in *Hybrid Systems: Computation and Control*, S. Sastry and T.A. Henzinger, Eds., number 1386 in LNCS, pp. 360–373. Springer-Verlag, Berlin, 1998.

[37] F.H. Clarke, Yu.S. Ledyaev, R.J. Stern, and P.R. Wolenski, *Nonsmooth analysis and control theory*, Springer-Verlag, New York, 1998.

[38] J. Lygeros, K.H. Johansson, S.N. Simić, J. Zhang, and S.S. Sastry, "Dynamical properties of hybrid automata," *IEEE Transactions on Automatic Control*, vol. 48, no. 1, pp. 2–17, January 2003.

[39] K.H. Johansson, M. Egerstedt, J. Lygeros, and S.S. Sastry, "On the regularization of Zeno hybrid automata," *Systems and Control Letters*, vol. 38, no. 3, pp. 141–150, 1999.

[40] A.S. Morse, "Control using logic based switching," in *Trends in Control*, A. Isidori, Ed., pp. 69–114. Springer Verlag, 1995.

[41] A.J. van der Schaft and H. Schumacher, "Complementarity modeling of hybrid systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 483–490, 1998.

[42] W. P. M. Heemels, B. De Schutter, and A. Bemporad, "Equivalence of hybrid dynamical models," *Automatica*, vol. 37, no. 7, pp. 1085–1091, 2001.

[43] M. Johansson, *Piecewise linear control systems*, Ph.D. thesis, Department of Automatic Control, Lund Institute of Technology, Sweden, March 1999.

[44] R. Alur, T.A. Henzinger, G. Lafferriere, and G.J. Pappas, "Discrete abstractions of hybrid systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 971–984, July 2000.

[45] J.-P. Aubin, J. Lygeros, M. Quincampoix, S.S. Sastry, and N. Seube, "Impulse differential inclusions: A viability approach to hybrid systems," *IEEE Transactions on Automatic Control*, vol. 47, no. 1, pp. 2–20, January 2002.

[46] M.S. Branicky, V.S. Borkar, and S.K. Mitter, "A unified framework for hybrid control: Model and optimal control theory," *IEEE Transactions on Automatic Control*, vol. 43, no. 1, pp. 31–45, 1998.

[47] M. Andersson, *Object-Oriented Modeling and Simulation of Hybrid Systems*, Ph.D. thesis, Lund Institute of Technology, Lund, Sweden, December 1994.

[48] S. E. Mattsson, M. Otter, and H. Elmqvist, "Modelica hybrid modeling and efficient simulation," in *IEEE Conference on Decision and Control*, Phoenix, AZ, 1999.

[49] S. E. Mattsson, M. Andersson, and K. J. Åström, "Object-oriented modelling and simulation," in *CAD for Control Systems*, D. A. Linkens, Ed., chapter 2, pp. 31–69. Marcel Dekker Inc., New York, 1993.

[50] M. Anderson, D. Bruck, S. E. Mattsson, and T. Schonthal, "Omsim- an integrated interactive environment for object-oriented modeling and simulation," in *IEEE/IFAC joint symposium on computer aided control system design*, 1994, pp. 285–290.

[51] A. Deshpande, A. Gollu, and L. Semenzato, "The SHIFT programming language for dynamic networks of hybrid automata," *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 584–587, Apr. 1998.

[52] J. Imura and A. J. van der Schaft, "Characterization of well-posedness of piecewise linear systems," *IEEE Transactions on Automatic Control*, vol. 45, no. 9, pp. 1600–1619, September 2000.

[53] M. Lemmon, "On the existence of solutions to controlled hybrid automata," in *Hybrid Systems: Computation and Control*, Nancy Lynch and Bruce H. Krogh, Eds., number 1790 in LNCS, pp. 229–242. Springer-Verlag, Berlin, 2000.

[54] M. Heemels, *Linear Complementarity Systems: a Study in Hybrid Dynamics*, Ph.D. thesis, Technische Universiteit Eindhoven, 1999.

[55] J. Lygeros, K.H. Johansson, S.S. Sastry, and M. Egerstedt, "On the existence of executions of hybrid automata," in *IEEE Conference on Decision and Control*, Phoenix, Arizona, December 7–10, 1999, pp. 2249–2254.

[56] L. Tavernini, "Differential automata and their simulators," *Nonlinear Analysis, Theory, Methods and Applications*, vol. 11(6), pp. 665–683, 1987.

[57] M. Broucke, "Regularity of solutions and homotopy equivalence for hybrid systems," in *IEEE Conference on Decision and Control*, Tampa, FL, 1998.

[58] V.S. Borkar, *Probability theory: an advanced course*, Springer-Verlag, New York, 1995.

[59] R. Alur and D. L. Dill, "Automata for modeling real-time systems," in *Proceedings of ICALP '90*, vol. 443 of *Lecture Notes in Computer Science*, pp. 322–335. Springer-Verlag, Berlin, 1990.

[60] B. Bérard, P. Gastin, and A. Petit, "On the power of non observable actions in timed automata," in *Actes du STACS '96*, Lecture Notes in Computer Science 1046, pp. 257–268. Springer-Verlag, Berlin, 1996.

[61] R. Alur and T.A. Henzinger, "Modularity for timed and hybrid systems," in *Proceedings of the Eighth International Conference on Concurrency Theory (CONCUR 1997)*, Berlin, 1997, number 1243 in LNCS, pp. 74–88, Springer-Verlag.

[62] K.H. Johansson, J. Lygeros, S.S. Sastry, and M. Egerstedt, "Simulation of Zeno hybrid automata," in *IEEE Conference on Decision and Control*, Phoenix, Arizona, December 7–10, 1999, pp. 3538–3543.

[63] M. Heemels, H. Schumacher, and S. Weiland, "Well-posedness of linear complementarity systems," in *Proc. 38th IEEE Conference on Decision and Control*, Phoenix, AZ, 1999.

[64] S. Simic, K.H. Johansson, S.S. Sastry, and J. Lygeros, "Towards a geometric theory of hybrid systems," in *Hybrid Systems: Computation and Control*, Nancy Lynch and Bruce H. Krogh, Eds., number 1790 in LNCS, pp. 421–436. Springer-Verlag, Berlin, 2000.

[65] J. Zhang, K.H. Johansson, J. Lygeros, and S.S. Sastry, "Zeno hybrid systems," *International Journal of Robust and Nonlinear Control*, vol. 11, pp. 435–451, 2001.

[66] S. E. Mattsson, "On object-oriented modeling of relays and sliding mode behaviour," in *Proc. 13th IFAC World Congress*, San Francisco, CA, 1996, vol. F, pp. 259–264.

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

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

[69] S.J. Hogan, "On the dynamics of a rigid-block motion under harmonic forcing," *Proceedings of the Royal Society, A*, vol. 425, pp. 441–476, 1989.

[70] H. Ye, A. Michel, and L. Hou, "Stability theory for hybrid dynamical systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 461–474, 1998.

[71] M.S. Branicky, "Multiple Lyapunov functions and other analysis tools for switched and hybrid systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 475–482, 1998.

[72] A.N. Michel and B. Hu, "Towards a stability theory for hybrid dynamical systems," *Automatica*, vol. 35, pp. 371–384, 1999.

[73] M. Johansson and A. Rantzer, "Computation of piecewise quadratic lyapunov functions for hybrid systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 555–559, 1998.

[74] J. Hespanha, "Uniform stability of switched linear systems: Extensions of lasalle's invariance principle," *IEEE Transactions on Automatic Control*, vol. 49, no. 4, pp. 470–482, April 2004.

[75] B. Hu, X. Xu, P.J. Antsaklis, and A.N. Michel, "Robust stabilizing control laws for a class of second order switched systems," *Systems & Control Letters*, vol. 38, no. 3, pp. 197–207, 1999.

[76] X. Xu and P.J. Antsaklis, "Stabilization of second order LTI switched systems," *International Journal of Control*, vol. 73, no. 14, pp. 1261–1279, September 2000.

[77] R. De Carlo, M. Branicky, S. Pettersson, and B. Lennarston, "Perspectives and results on the stability and stabilizability of hybrid systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 1069–1082, July 2000.

[78] M. Wicks, P. Peleties, and R. De Carlo, "Switched controller synthesis for the quadratic stabilization of a pair of unstable linear systems," *European Journal of Control*, vol. 4, pp. 140–147, 1998.

[79] D. Liberzon and A.S. Morse, "Basic problems in stability and design of switched systems," *IEEE Control Systems Magazine*, vol. 19, pp. 59–70, October 1999.

[80] D. Liberzon, J.P. Hespanha, and A.S. Morse, "Stability of switched systems: A Lie algebraic approach," *Systems and Control Letters*, vol. 37, no. 3, pp. 117–122, 1999.

[81] A. V. Savkin, E. Skafidas, and R.J. Evans, "Robust output feedback stabilizability via controller switching," *Automatica*, vol. 35, no. 1, pp. 69–74, 1999.

[82] D. Liberzon, *Switching in Systems and Control*, Birkhauser, Boston, 2003.

[83] Z. Manna and A. Pnueli, *The Temporal Logic of Reactive and Concurrent Systems: Specification*, Springer-Verlag, Berlin, 1992.

[84] Z. Manna and A. Pnueli, *Temporal Verification of Reactive Systems: Safety*, Springer-Verlag, New York, 1995.

[85] N. Lynch, *Distributed Algorithms*, Morgan Kaufmann, 1996.

[86] M.S. Branicky, E. Dolginova, and N. Lynch, "A toolbox for proving and maintaining hybrid specifications," in *Hybrid Systems IV*, A. Nerode P. Antsaklis, W. Kohn and S. Sastry, Eds., number 1273 in LNCS, pp. 18–30. Springer-Verlag, Berlin, 1997.

[87] Zohar Manna and Henny Sipma, "Deductive verification of hybrid systems using STeP," in *Hybrid Systems: Computation and Control*, S. Sastry and T.A. Henzinger, Eds., number 1386 in LNCS, pp. 305–318. Springer-Verlag, Berlin, 1998.

[88] C. Heitmeyer and N. Lynch, "The generalized railroad crossing: A case study in formal verification of real-time systems," in *Proc. ICCC Real-Time Systems Symposium*, San Juan, Puerto Rico, 1994.

[89] H.B. Weinberg, Nancy Lynch, and Norman Delisle, "Verification of automated vehicle protection systems," in *Hybrid Systems III*, number 1066 in LNCS, pp. 101–113. Springer-Verlag, Berlin, 1996.

[90] E. Dolginova and N. Lynch, "Safety verification for automated platoon maneuvers: a case study," in *Proceedings of HART97*, Oded Maler, Ed., number 1201 in LNCS, pp. 154–170. Springer-Verlag, Berlin, 1997.

[91] J. Lygeros and N. Lynch, "Strings of vehicles: Modeling and safety conditions," in *Hybrid Systems: Computation and Control*, S. Sastry and T.A. Henzinger, Eds., number 1386 in LNCS, pp. 273–288. Springer-Verlag, Berlin, 1998.

[92] C. Livadas and N. Lynch, "Formal verification of safety-critical hybrid systems," in *Hybrid Systems: Computation and Control*, S. Sastry and T.A. Henzinger, Eds., number 1386 in LNCS, pp. 253–272. Springer-Verlag, Berlin, 1998.

[93] C. Livadas, J. Lygeros, and N.A. Lynch, "High-level modeling and analysis of the traffic alert and collision avoidance system (TCAS)," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 926–948, July 2000.

[94] Zohar Manna and the STeP group, "STeP: The Stanford Temporal Prover," Tech. Rep. STAN-CS-TR-94-1518, Computer Science Department, Stanford University, July 1994.

[95] R. Alur, C. Courcoubetis, N. Halbawachs, T. A. Henzinger, P. H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine, "The algorithmic analysis of hybrid systems," *Theoretical Computer Science*, vol. 138, pp. 3–34, 1995.

[96] T. Henzinger, P. Kopke, A. Puri, and P. Varaiya, "What's decidable about hybrid automata," in 27^{th} *Annual Symposium on the Theory of Computing, STOC'95*. 1995, pp. 373–382, ACM Press.

[97] G. Lafferriere, G.J. Pappas, and S.S. Sastry, "O-minimal hybrid systems," *Mathematics of Control, Signals, and Systems*, vol. 13, no. 1, pp. 1–21, March 2000.

[98] A. Puri, P. Varaiya, and V. Borkar, "ϵ-approximation of differential inclusions," in *IEEE Conference on Decision and Control*, New Orleans, LA, 1995, pp. 2892–2897.

[99] M. Heymann, F. Lin, and G. Meyer, "Synthesis and viability of minimally interventive legal controllers for hybrid systems," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 8, no. 2, pp. 105–135, June 1998.

[100] E. Asarin, O. Bournez, T. Dang, O. Maler, and A. Pnueli, "Effective synthesis of switching controllers for linear systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 1011–1025, July 2000.

[101] R. Alur and R.P. Kurshan, "Timing analysis in COSPAN," in *Hybrid Systems III*, number 1066 in LNCS, pp. 220–231. Springer-Verlag, Berlin, 1996.

[102] C. Daws, A. Olivero, S. Trypakis, and S. Yovine, "The tool KRONOS," in *Hybrid Systems III*, R. Alur, T. Henzinger, and E. Sontag, Eds., number 1066 in LNCS, pp. 208–219. Springer-Verlag, Berlin, 1996.

[103] J. Bengtsson, K.G. Larsen, F. Larsson, P. Petterson, and W. Yi, "UPAAL: A tool suit for automatic verification of real-time systems," in *Hybrid Systems III*, number 1066 in LNCS, pp. 232–243. Springer-Verlag, Berlin, 1996.

[104] R. Alur, T. Dang, J. Esposito, R. Fierro, Y. Hur, F. Ivancic, V. Kumar, I. Lee, P. Mishra, G. Pappas, and O. Sokolsky, "Hierarchical hybrid modeling of embedded systems," in *First Workshop on Embedded Software*, T.A. Henzinger and C.M. Kirsch, Eds., number 2211 in LNCS, pp. 14–31. Springer-Verlag, Berlin, 2001.

[105] C.J. Tomlin, J. Lygeros, and S.S. Sastry, "A game theoretic approach to controller design for hybrid systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 949–969, July 2000.

[106] A. Church, "Logic, arithmetic, and automata," in *Proceedings of the International Congress of Mathematicians*, pp. 23–35. 1962.

[107] J.R. Büchi and L.H. Landweber, "Solving sequential conditions by finite-state operators," in *Proceedings of the American Mathematical Society*, 1969, pp. 295–311.

[108] M. O. Rabin, "Automata on infinite objects and Church's problem," in *Regional Conference Series in Mathematics*, 1972.

[109] P. J. G. Ramadge and W. M. Wonham, "The control of discrete event systems," *Proceedings of the IEEE*, vol. Vol.77, no. 1, pp. 81–98, 1989.

[110] J. Lygeros, C.J. Tomlin, and S.S. Sastry, "Controllers for reachability specifications for hybrid systems," *Automatica*, vol. 35, no. 3, pp. 349–370, March 1999.

[111] A.E. Bryson and Y.-C. Ho, *Applied Optimal Control*, Hemisphere Publishing Corporation, 1975.

[112] L. C. Young, *Optimal Control Theory*, Chelsea, second edition, 1980.

[113] T. Başar and G. J. Olsder, *Dynamic Non-cooperative Game Theory*, Academic Press, second edition, 1995.

[114] J.-P. Aubin, *Viability Theory*, Birkhäuser, Boston, MA, 1991.

[115] M.G. Crandall and P.-L. Lions, "Viscosity solutions of Hamilton–Jacobi equations," *Transactions of the American Mathematical Society*, vol. 277, no. 1, pp. 1–42, May 1983.

[116] L.C. Evans, *Partial differential equations*, American Mathematical Society, Providence, R.I., 1998.

[117] John Lygeros, "On reachability and minimum cost optimal control," *Automatica*, vol. 40, no. 6, pp. 917–927, 2004.

[118] I. Mitchell, A.M. Bayen, and C.J. Tomlin, "Computing reachable sets for continuous dynamic games using level set methods," Preprint.

[119] E.N. Barron and H. Ishii, "The Bellman equation for minimizing the maximum cost," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 13, no. 9, pp. 1067–1090, 1989.

[120] I.J. Fialho and T.T. Georgiou, "Worst case analysis of nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 44, no. 6, pp. 1180–1197, June 1999.

[121] M. Quincampoix and O.-S. Serea, "A viability approach for optimal control with infimum cost," 2002, Preprint.

[122] W.H. Flemming and H.M. Soner, *Controlled Markov Processes and Viscosity Solutions*, Springer-Verlag, New York, 1993.

[123] M. Bardi and I. Capuzzo-Dolcetta, *Optimal Control and Viscosity Solutions of Hamilton–Jacobi–Bellman Equations*, Birkhäuser, Boston, MA, 1997.

[124] P. Cardaliaguet, M. Quincampoix, and P. Saint-Pierre, "Pursuit differential games with state constraints," *SIAM Journal of Control and Optimization*, vol. 39, no. 5, pp. 1615–1632, 2001.

[125] E.N. Barron, "Differential games with maximum cost," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 14, no. 11, pp. 971–989, 1990.

[126] L.C. Evans and P.E. Souganidis, "Differential games and representation formulas for solutions of hamilton–jacobi–isaacs equations," *Indiana University Mathematics Journal*, vol. 33, no. 5, pp. 773–797, 1984.

[127] S.C. Di Marco and R.L.V. Gonzalez, "Supersolution and subsolutions techniques in a minimax optimal control problem with infinite horizon," *Indian Journal of Pure and Applied Mathematics*, vol. 29, no. 10, pp. 1083–1098, October 1998.

[128] S.C. Di Marco and R.L.V. Gonzalez, "Relaxation of minimax optimal control problem with infinite horizon," *Journal of Optimization Theory and Applications*, vol. 101, no. 2, pp. 285–306, May 1999.

[129] J.-P. Aubin and H. Frankowska, "The viability kernel algorithm for computing value functions of infinite horizon optimal control probolems," *J. Math. Anal. Appl.*, vol. 201, pp. 555–576, 1996.

[130] P. Cardaliaguet, M. Quincampoix, and P. Saint-Pierre, "Numerical schemes for discontinuous value functions of optimal control," *Set Valued Analysis*, vol. 8, pp. 111–126, 2000.

[131] J.-R. Abrial, E. Börger, and H. Langmaack, "The steam-boiler case study project, an introduction," in *Formal Methods for Industrial Applications: Specifying and Programming the Steam Boiler Control*, J.-R. Abrial, E. Börger, and H. Langmaack, Eds., number 1165 in LNCS. Springer-Verlag, Berlin, 1996.

[132] J.-R. Abrial, "The steam-boiler control specification problem," in *Formal Methods for Industrial Applications: Specifying and Programming the Steam Boiler Control*, J.-R. Abrial, E. Börger, and H. Langmaack, Eds., number 1165 in LNCS. Springer-Verlag, Berlin, 1996.

[133] Tomas A. Henzinger and Howard Wong-Toi, "Using HYTECH to synthesize control parameters for a steam boiler," in *Formal Methods for Industrial Applications: Specifying and Programming the Steam Boiler Control*, J.-R. Abrial, E. Börger, and H. Langmaack, Eds., number 1165 in LNCS, pp. 265–282. Springer-Verlag, Berlin, 1996.

[134] A. Puri, *Theory of Hybrid Systems and Discrete Event Systems*, Ph.D. thesis, Department of Electrical Engineering, University of California, Berkeley, 1995.

[135] J. Lygeros, C.J. Tomlin, and S.S. Sastry, "Multi-objective hybrid controller synthesis," Tech. Rep. UCB/ERL M96/59, Electronic Research Laboratory, University of California Berkeley, 1997.

[136] A. Nuic, "User manual for the base of aircraft data (BADA) revision 3.3," Tech. Rep. EEC Note no. 20/00, Eurocontrol Experimental Centre, December 2000.

[137] C.J. Tomlin, J. Lygeros, and S.S. Sastry, "Aerodynamic envelope protection using hybrid control," in *American Control Conference*, Philadelphia, Pennsylvania, June 24–26, 1998, pp. 1793–1796.

[138] I. Mitchell and C.J. Tomlin, "Level set methods for computation in hybrid systems," in *Hybrid Systems: Computation and Control*, Nancy Lynch and Bruce H. Krogh, Eds., number 1790 in LNCS, pp. 310–323. Springer-Verlag, Berlin, 2000.

[139] I. Mitchell, A.M. Bayen, and C.J. Tomlin, "Validating a Hamilton–Jacobi approximation to hybrid system reachable sets," in *Hybrid Systems: Computation and Control*, M. Di Benedetto and A. Sangiovanni-Vincentelli, Eds., number 2034 in LNCS, pp. 418–432. Springer-Verlag, Berlin, 2001.

[140] S. Osher and J.A. Sethian, "Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton–Jacobi formulations," *Journal of Computational Physics*, vol. 79, pp. 12–49, 1988.

[141] J. A. Sethian, *Level Set Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision, and Materials Science*, Cambridge University Press, New York, 1996.

[142] M. Oishi and C. Tomlin, "Switched nonlinear control of a VSTOL aircraft," in *IEEE Conference on Decision and Control*, Phoenix, Arizona, December 7–10, 1999, pp. –.

[143] M. Oishi, C. Tomlin, V. Gopal, and D. Godbole, "Addressing multiobjective control: Safety and performance through constrained optimization," in *Hybrid Systems: Computation and Control*, M. Di Benedetto and A. Sangiovanni-Vincentelli, Eds., number 2034 in LNCS, pp. 459–472. Springer-Verlag, Berlin, 2001.

[144] M.S. Branicky, *Studies in Hybrid Systems: Modeling, Analysis, and Control*, Ph.D. thesis, Massachusetts Institute of Technology, 1995.

[145] E.D. Sontag, *Mathematical Control Theory*, Springer-Verlag, New York, 1990.

[146] V.I. Arnold, *Mathematical Methods of Classical Mechanics*, Springer-Verlag, second edition, 1989.

[147] J.-P. Aubin and H. Frankowska, *Set Valued Analysis*, Birkhäuser, Boston, MA, 1990.

[148] P. Saint-Pierre, "Approximation of viability kernels and capture basins for hybrid systems," in *European Control Conference*, Porto, September 4-7, 2001, pp. 2776–2783.

[149] A.B. Kurzhanski and P. Varaiya, "Ellipsoidal techniques for reachability analysis," in *Hybrid Systems: Computation and Control*, Nancy Lynch and Bruce H. Krogh, Eds., number 1790 in LNCS, pp. 202–214. Springer-Verlag, Berlin, 2000.

[150] A. B. Kurzhanski and P. Varaiya, "On reachability under uncertainty," *SIAM Journal of Control and Optimization*, vol. 41, no. 1, pp. 181–216, 2002.

[151] X.D. Koutsoukos, P.J. Antsaklis, J.A. Stiver, and M.D. Lemmon, "Supervisory control of hybrid systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 1026–1049, July 2000.

[152] J.M. Davoren and A. Nerode, "Logics for hybrid systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 985–1010, July 2000.

[153] G. Grammel, "Maximum principle for a hybrid system via singular pertrubations," *SIAM Journal of Control and Optimization*, vol. 37, no. 4, pp. 1162–1175, 1999.

[154] A. Chutinam and B. Krogh, "Verification of polyhedral-invariant hybrid automata using polygonal flow pipe approximations," in *Hybrid Systems: Computation and Control*, Frits W. Vaandrager and Jan H. van Schuppen, Eds., number 1569 in LNCS, pp. 76–90. Springer-Verlag, Berlin, 1999.

[155] O. Botchkarev and S. Tripakis, "Verification of hybrid systems with linear differential inclusions using ellipsoidal approximations," in *Hybrid Systems: Computation and Control*, Nancy Lynch and Bruce H. Krogh, Eds., number 1790 in LNCS, pp. 73–88. Springer-Verlag, Berlin, 2000.

[156] E. Asarin, O. Bournez, T. Dang, and O. Maler, "Approximate reachability analysis of piecewise-linear dynamical systems," in *Hybrid Systems: Computation and Control*, Nancy Lynch and Bruce H. Krogh, Eds., number 1790 in LNCS. Springer-Verlag, Berlin, 2000.