Rupak Majumdar
Paulo Tabuada (Eds.)

LNCS 5469

Hybrid Systems: Computation and Control

12th International Conference, HSCC 2009
San Francisco, CA, USA, April 2009
Proceedings

Springer

Lecture Notes in Computer Science 5469

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Rupak Majumdar Paulo Tabuada (Eds.)

Hybrid Systems: Computation and Control

12th International Conference, HSCC 2009
San Francisco, CA, USA, April 13-15, 2009
Proceedings

Springer

Volume Editors

Rupak Majumdar
University of California, Department of Computer Science
4531E Boelter Hall, Los Angeles, CA 90095, USA
E-mail: rupak@cs.ucla.edu

Paulo Tabuada
University of California, Department of Electrical Engineering
66-147F Engineering IV, Los Angeles, CA 90095, USA
E-mail: tabuada@ee.ucla.edu

Library of Congress Control Number: Applied for

CR Subject Classification (1998): C.3, C.1.3, F.3, D.2, F.1.2, J.2, I.6

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743
ISBN-10 3-642-00601-9 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-00601-2 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12634446 06/3180 5 4 3 2 1 0

Preface

This volume contains the proceedings of the 12th International Conference on Hybrid Systems Computation and Control (HSCC 2009) held in San Francisco, California during April 13-15, 2009. The annual conference on hybrid systems focuses on research in embedded, reactive systems involving the interplay between discrete switching and continuous dynamics. HSCC is a forum for academic and industrial researchers and practitioners to exchange information on the latest advancements, both practical and theoretical, in the design, analysis, control, optimization, and implementation of hybrid systems.

HSCC 2009 was the 12th in a series of successful meetings. Previous versions were held in Berkeley (1998), Nijmegen (1999), Pittsburgh (2000), Rome (2001), Palo Alto (2002), Prague (2003), Philadelphia (2004), Zurich (2005), Santa Barbara (2006), Pisa (2007), and St. Louis (2008).

HSCC 2009 was part of the 2nd Cyber-Physical Systems Week (CPSWeek), which consisted of the co-location of HSCC with the International Conference on Information Processing in Sensor Networks (IPSN) and the Real-Time and Embedded Technology and Applications Symposium (RTAS). Through CPSWeek, the three conferences had joint invited speakers, poster sessions, and joint social events. In addition to the workshops sponsored by CPSWeek, HSCC 2009 sponsored two workshops:

- NSV II: Second International Workshop on Numerical Software Verification
- HSCB 2009: Hybrid Systems Approaches to Computational Biology

We would like to thank the authors of submitted papers, the Program Committee members, the additional reviewers, the workshop organizers, and the HSCC Steering Committee members for their help in composing a strong program. We also thank the CPSWeek Organizing Committee, in particular Rajesh Gupta, for their strenuous work in handling the local arrangements. Finally, we would also like to thank Springer for having agreed to publish these proceedings as a volume in the *Lecture Notes in Computer Science* series and to EasyChair for hosting the management service for paper submissions, reviewing, and final generation of proceedings.

April 2009

Rupak Majumdar
Paulo Tabuada

Organization

Program Chairs

Rupak Majumdar	University of California at Los Angeles, USA
Paulo Tabuada	University of California at Los Angeles, USA

Steering Committee

Rajeev Alur	University of Pennsylvania, USA
Bruce Krogh	Carnegie Mellon University, USA
Oded Maler	VERIMAG, France
Manfred Morari	ETH, Switzerland
George Pappas	University of Pennsylvania, USA
John Rushby	SRI International, USA

Program Committee

Panos Antsaklis	University of Notre Dame, USA
Karl-Erik Årzén	Lund University, Sweden
Alexandre Bayen	University of California at Berkeley, USA
Calin Belta	Boston University, USA
Alberto Bemporad	Università di Siena, Italy
Ed Brinksma	Embedded Systems Institute, The Netherlands
Luca Carloni	Columbia University, USA
Patrick Cousot	New York University, USA
Domitilla del Vecchio	University of Michigan, USA
Stephen Edwards	Columbia University, USA
Martin Fränzle	Carl von Ossietzky Universität, Germany
Emilio Frazzoli	Massachusetts Institute of Technology, USA
Antoine Girard	Université Joseph Fourier, France
Radu Grosu	State University of New York at Stony Brook, USA
Maurice Heemels	Eindhoven University of Technology, The Netherlands
Thomas Henzinger	Ecole Polytechnique Fédérale de Lausanne, Switzerland
Jun-ichi Imura	Tokyo Institute of Technology, Japan
Karl Henrik Johansson	Royal Institute of Technology, Sweden
Eric Klavins	University of Washington, USA

Daniel Liberzon	University of Illinois at Urbana-Champaign, USA
Ian Mitchell	University of British Columbia, Canada
Luigi Palopoli	Università degli Studi di Trento, Italy
Carla Piazza	Università degli Studi di Udine, Italy
Nacim Ramdani	INRIA, France
Jan Rutten	CWI, The Netherlands
Sriram Sankaranarayanan	NEC Laboratories, USA
Olaf Stursberg	Technical University of Munich, Germany
Herbert Tanner	University of Delaware, USA
Ashish Tiwari	SRI International, USA
Stavros Tripakis	Cadence Labs, USA
Franck van Breugel	York University, Canada
Manel Velasco	Universitat Politècnica de Catalunya, Spain
Mahesh Viswanathan	University of Illinois at Urbana-Champaign, USA

External Reviewers

Abate, Alessandro
Althoff, Matthias
Anta, Adolfo
Azuma, Shun-ichi
Bartocci, Ezio
Batt, Gregory
Bernardini, Daniele
Bortolussi, Luca
Bouissou, Olivier
Bradley, Aaron
Bresolin, Davide
Callanan, Sean
Camlibel, Kanat
Campagna, Dario
Casagrande, Alberto
Chadha, Rohit
Collins, Pieter
Crouzen, Pepijn
Cucinotta, Tommaso
D'Innocenzo, Alessandro
Dimarogonas, Dimos
Dold, Johannes
Donkers, Tijs
Doyen, Laurent
Eggers, Andreas
Fainekos, Georgios
Fischione, Carlo
Fontanelli, Daniele
Frehse, Goran
Ganty, Pierre
Garcia, Eloy
Groote, Jan Friso
Gupta, Vijay
Hafner, Mike
Henriksson, Erik
Ivancic, Franjo
Kashima, Kenji
Kobayashi, Koichi
Komenda, Jan
Lazar, Mircea
Lindha, Magnus
Maler, Oded
Martí, Pau
Mazo, Manuel
McCourt, Michael
McNew, John-Michael
Mitra, Sayan
Mueller, Matthias
Napp, Nils
Niqui, Milad
Noll, Dominikus
Norman, Gethin
Oehlerking, Jens
Paschedag, Tina

Passenberg, Benjamin
Petreczky, Mihaly
Platzer, Andre
Pola, Giordano
Polderman, Jan Willem
Porreca, Riccardo
Prabhakar, Pavithra
Puppis, Gabriele
Rabi, Maben
Ramirez-Prado, Guillermo
Riganelli, Oliviero
Ripaccioli, Giulio
Rondepierre, Aude
Rungger, Matthias
Rutkowski, Michal
Sandberg, Henrik
Sanfelice, Ricardo
Sharon, Yoav
Shaw, Fayette
Smolka, Scott
Sobotka, Marion
Tanwani, Aneel
Tartamella, Chris
Tazaki, Yuichi
Teige, Tino
Thorsley, David
Verma, Rajeev
Vladimerou, Vladimeros
Wang, Xiaofeng
Wolf, Verena
Worrell, James
Wu, Po
Ye, Pei
Yu, Han
Zamani, Majid
Zavlanos, Michael
van der Schaft, Arjan

Table of Contents

Regular Papers

Short Papers

Applications of MetiTarski in the Verification of Control and Hybrid Systems

Behzad Akbarpour[1] and Lawrence C. Paulson[2]

[1] Concordia University, Montreal, Quebec, H3G 1M8, Canada
behzad@ece.concordia.ca
[2] Computer Laboratory, University of Cambridge, England
lp15@cam.ac.uk

Abstract. MetiTarski, an automatic proof procedure for inequalities on elementary functions, can be used to verify control and hybrid systems. We perform a stability analysis of control systems using Nichols plots, presenting an inverted pendulum and a magnetic disk drive reader system. Given a hybrid systems specified by a system of differential equations, we use Maple to obtain a problem involving the exponential and trigonometric functions, which MetiTarski can prove automatically.

1 Introduction

Most research into the verification of hybrid systems involves model checking and constraint solving. In this paper, we present preliminary results involving the use of automated theorem proving. Our approach delivers proofs of its claims, which can be checked by other tools or even examined by humans. These proofs are low-level and can be very long; for example, the proof of the collision avoidance problem (see Sect. 4.1) consists of nearly 2600 text lines and 162 logical inferences, some of which refer to decision procedures. Formal verification is typically used in applications that demand high assurance. Our methodology can produce documentation of every phase of the formal analysis of the design, from differential equations to proof.

MetiTarski [1,2,3] is a new automatic theorem prover for special functions over the real numbers. It consists of a resolution theorem prover (Metis) combined with a decision procedure (QEPCAD) for the theory of real closed fields. It can prove logical statements involving the functions ln, exp, sin, cos, arctan, sqrt, etc. We have applied it to hundreds of problems mainly of mathematical origin. In this paper, we report recent experiments in which we have applied MetiTarski to standard benchmark problems about hybrid and control systems.

Our workflow typically involves using a computer algebra system (Maple) to solve a differential equation. The result is a formula over the real numbers, which we supply to MetiTarski. For most problems that we have investigated, MetiTarski returns a proof in seconds. The entry of problems is currently manual, though it is not difficult, because the output of Maple can be pasted into MetiTarski, with modest further editing to put the problem into the right form. These tasks are routine and could be automated.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 1–15, 2009.

Paper outline. Section (§2) reviews some related work. Section (§3) describes the verification of control systems. Section (§4) describes the details of our approach for the verification of hybrid systems using illustrative case studies. Section (§5) concludes the paper and provides hints for future work directions.

2 Related Work

Control systems are traditionally analysed using numerical techniques, often involving the visual inspection of plots for a number of sample inputs and different values of parameters. Then we must assume that the results of this analysis also hold for any values of the input and the parameters. This assumption can lead to incorrect conclusions. Hardy [10] proposed a formal and symbolic technique to increase the reliability of the results, removing the possibility of erroneous results due to plotting errors and uncertain parameters. She examined the underlying mathematical representation of a particular form of control system requirements: Nichols plot requirements. These requirements were reduced to their most basic form and a decision procedure was developed for use in the analysis which can be used to decide the positivity or negativity of finitely inflective functions. The resulting tool, called Nichols plot Requirements Verifier (NRV), was developed in the Maple-PVS-QEPCAD system which exploits the symbolic computation provided by the computer algebra system Maple, the formal techniques provided by the theorem prover PVS and the quantifier elimination routines provided by QEPCAD. Hardy presented two case studies to demonstrate the practical application of the NRV system. In this paper, we achieve similar results by replacing the PVS-QEPCAD combination with MetiTarski. We still use Maple for initial calculations but we replace the semi-automatic proofs by PVS with fully automatic proofs of MetiTarski.

Several techniques for model checking of hybrid systems have been proposed. The most widely investigated is bounded model checking (BMC), which computes a set of reachable states that corresponds to an over-approximation of the solution of the system equations obtained for a bounded period of time. This approach provides the algorithmic foundations for the tools that are available for computer-aided verification of hybrid systems such as Checkmate [6], d/dt [5], PHaver [9], and HyTech [11]. On the other hand, there are some hybrid system verification tools such as Stefan Ratschan's HSolver [14], which are based on constraint solving techniques. The basic idea is to decompose the state space into hyperboxes according to a rectangular grid and then use interval constraint-propagation techniques to check the flow on the boundary between neighboring grid elements. This is done via an abstraction refinement framework in order to achieve precise results.

In this paper, we present a novel approach based on automatic theorem proving for hybrid system verification. We show how our tool MetiTarski assisted with Maple can be used to prove safety properties about hybrid systems. We have selected a set of case studies in real world applications collected from standard benchmarks [15] for evaluating and comparing tools for hybrid system design and verification. Our current examples are restricted to linear systems for which we can solve the systems of ordinary differential equations (ODE) using methods

like the Laplace transform to find the closed form solutions based on elementary functions. We have been able to prove safety properties of the systems such as Room Heating and Navigation, which cannot be verified by HSolver.[1] We are planning to extend our case studies to cover nonlinear cases by finding methods of solving systems of polynomial nonlinear ordinary differential equations analytically in terms of elementary and special functions. An example of such method is the Prelle-Singer procedure [12], extensions of which are also implemented in computer algebra systems such as REDUCE (the PSODE package [13]) and Maple (the PSsolver package [8]).

3 Control Systems Verification

This section presents our methodology for using MetiTarski in the verification of control systems. Our approach can be briefly described as follows. We start from the open loop transfer function of the feedback control system in Laplace domain as a function of s ($G(s)$). Then we replace s with jw and switch to frequency domain. Then we calculate the gain and phase shift of $G(jw)$ according to Equation 1, as real valued functions over w, and plot them in the x/y plane and call it the Nichols plot. For stability, the Nichols plot of the system should lie outside an exclusion region which will be explained later. We describe this obligation as inequalities on special functions such as arctan and log over w, and prove them using MetiTarski. We use Maple to plot the Nichols plots, and also for some preliminary investigations about the intermediate expressions.

We illustrate our methodology using two moderately sized case studies, both based on examples that appear regularly in control engineering texts. In Section 3.2, an inverted pendulum system is analysed. The stability criteria are specified in terms of three intervals in which the Nichols plot of the system must not enter a given bounded region on the graph. We use MetiTarski to verify this system. We then alter the system and use MetiTarski to show that the system is now unstable. In both cases, the Nichols plot for the system lies too close to the exclusion region to be confirmed by visual inspection. In Section 3.3, a disk drive reader system is analysed with respect to stability. This system has an 'uncertain' parameter, whose value is known to lie within an interval. This type of problem is difficult to analyse using classical Nichols plot techniques as it is a three dimensional rather than two dimensional problem. The classical solution is to plot a suite of Nichols plots showing the system response for various values of the parameter. If the system meets its requirements in all of these plots the assumption is made that the system meets its requirements for all permissible values of the parameter. In this case study we provide symbolic analysis of the system for all permissible values of the parameter, generating a formal proof.

3.1 Nichols Plot Requirements

There are three main graphical analysis techniques used in the analysis of control systems in frequency or the complex plane: the Nyquist plot (complex plane),

[1] See http://hsolver.sourceforge.net/benchmarks/

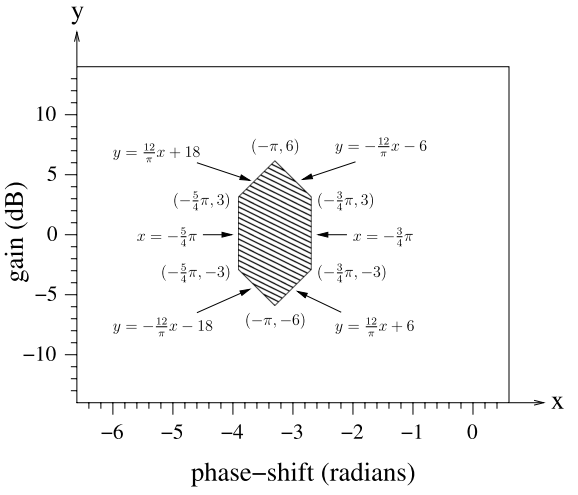


Fig. 1. Exclusion Region

Bode diagrams (frequency domain), and Nichols plots (frequency domain). We will discuss in particular analysis using Nichols plots. The *Nichols plot* [7] (also known as *Nichols chart*) plots the gain (in decibels) against the phase-shift of the output sinusoid as the frequency varies. The gain and phase-shift of a system with transfer function G can be calculated explicitly using the following formulas:

$$\begin{aligned} y &= \text{gain} = 20\log_{10}(|G(jw)|) \\ x &= \text{phase-shift} = \text{argument}(G(jw)) \\ &= \begin{cases} \arctan(\frac{\Im(G(jw))}{\Re(G(jw))}) + k\pi & \text{if } \Re(G(jw)) \neq 0 \\ \frac{\pi}{2} + k\pi & \text{if } \Re(G(jw)) = 0 \end{cases} \end{aligned} \tag{1}$$

where $\Re$ ($\Im$) denotes the real (imaginary) part of a complex number, and k is an integer. When using arctan to calculate the value of phase-shift, we may have to adjust the range of arctan, which normally is restricted to $(-\frac{\pi}{2}, \frac{\pi}{2})$ in radians. If the shift in phase at w is greater than $\frac{\pi}{2}$ then $\arctan(\frac{\Im(G(jw))}{\Re(G(jw))})$ must be adjusted by an appropriate multiple k of π to give the phase-shift as in equation 1.

Nichols plots often show *exclusion regions* that must be avoided to achieve stability and performance. In general, a system is considered stable if its Nichols plot does not enter a certain hexagonal region about the point $(-\pi, 0)$ as shown in Fig.1. This requirement can be expressed in terms of the lines bounding the region in three intervals.

1. The Nichols plot for the system must lie below the line $y = -\frac{12}{\pi}x - 18$ between the points $(-\frac{5}{4}\pi, -3)$ and $(-\pi, -6)$, or above the line $(y = \frac{12}{\pi}x + 18)$ between the points $(-\frac{5}{4}\pi, 3)$ and $(-\pi, 6)$.
2. It must lie below the line $y = -\frac{12}{\pi}x - 6$ between the points $(-\pi, -6)$ and $(-\frac{3}{4}\pi, -3)$, or above the line $y = \frac{12}{\pi}x + 6$ between $(-\pi, 6)$ and $(-\frac{3}{4}\pi, 3)$.

3. It must lie to the left of line $x = -\frac{5}{4}\pi$ between the points $(-\frac{5}{4}\pi, -3)$ and $(-\frac{5}{4}\pi, 3)$, or to the right of line $x = \frac{-3}{4}\pi$ between $(-\frac{3}{4}\pi, -3)$ and $(-\frac{3}{4}\pi, 3)$.

These conditions can be expressed as inequalities in arctan, ln, and square root.

Several different cases of curves can be identified depending on whether $y = f(x)$ is monotonic decreasing, or monotonic increasing and concave, or monotonic increasing and convex. These properties help to reduce the proofs to specific points instead of a whole range. A real-valued function f defined on an interval is *convex* if for any two points x and y in its domain and any t in $[0, 1]$, we have

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y).$$

A function f is said to be *concave* if $-f$ is convex. A twice differentiable function of one variable is convex on an interval if and only if its second derivative is non-negative there; this gives a practical test for convexity. A *point of inflection* is a point on a curve at which the curvature changes sign; at this point, the graph of the function makes a smooth transition between convexity and strict concavity. These conditions can be easily checked using Maple.

3.2 Inverted Pendulum

This section focuses on the modeling and analysis of an inverted pendulum system. An *inverted pendulum* is a pendulum that has its mass above its pivot point, which is mounted on a cart that can move horizontally (Fig. 2). Whereas a normal pendulum is stable when hanging downwards, an inverted pendulum is inherently unstable, and must be actively balanced in order to remain upright by applying a horizontal force to the cart. The inverted pendulum is a classic problem in dynamics and control theory and is widely used as benchmark for testing control algorithms.

There are two outputs of interest: the displacement of the cart x and the angle of the pendulum θ. When concerned only with the angle of the pendulum, the behaviour of the system can be represented using the following transfer function

$$G(s) = \frac{ml(K_d s^2 + K_p s + K_i)}{(MI + Mml^2 + mI)s^3 + (bI + bml^2)s^2 - (Mmgl + m^2gl)s - bmgl}$$

Table 1 shows the values for the parameters of the system chosen for this example. The value of the mass of the pendulum m is left undecided.

Analysis of an Inverted Pendulum that Meets its Requirements. Assuming that the mass of the pendulum is 0.2 kg, the open loop transfer function for the inverted pendulum system is

$$G(s) = \frac{-25(2s^2 - 7s + 2)}{11s^3 + 2s^2 - 343s - 49}$$

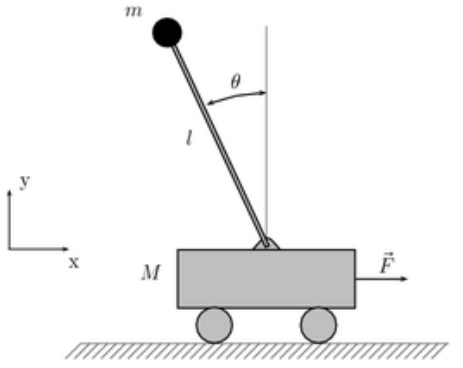


Fig. 2. Inverted Pendulum

Table 1. Values for parameters in an inverted pendulum system

Mass of cart	M	0.5 kg
Friction of the cart	b	0.1
Length to the pendulum's center of mass	l	0.3 m
Inertia of the pendulum	I	0.006 kgm^2
Gravitational acceleration	g	9.8 m/sec^2
Proportional coefficient	K_p	3.5
Integral coefficient	K_i	-1
Derivative coefficient	K_d	-1

and the gain and phase-shift can be calculated as follows:

$$y = 20\log_{10}\left(\frac{25\sqrt{\begin{array}{c}484w^{10}+35161w^8+781414w^6\\+4871449w^4+569821w^2+9604\end{array}}}{121w^6+7550w^4+117845w^2+2401}\right)$$

$$x = \begin{cases} -\arctan(\dfrac{650w^3-1029w+22w^5}{81w^4+24595w^2-98}) & \text{if } 0 \le w < 0.198 \\ -\pi & \text{if } w = 0.198 \\ -\arctan(\dfrac{650w^3-1029w+22w^5}{81w^4+24595w^2-98}) - \pi & \text{if } 0.198 < w \end{cases}$$

Next we use Maple and MetiTarski to analyse this system with respect to the exclusion region criteria and prove that it meets its requirements as follows:

1. We first calculate using Maple that the interval $[-\frac{5}{4}\pi, -\pi]$, in terms of x, corresponds to the interval $[\frac{157}{128}, \frac{129}{32}]$ in terms of w and then use MetiTarski to show that

$$\forall w.\ \frac{157}{128} \ge w \vee w \ge \frac{129}{32} \Longrightarrow -\frac{5}{4}\pi \ge x \vee x \ge -\pi.$$

Analysis using Maple shows that within this interval there is one point of inflection, which lies in the interval $[\frac{569}{256}, \frac{1139}{512}]$. The curve is convex for

$w \in [\frac{157}{128}, \frac{569}{256}]$ and concave for $w \in [\frac{1139}{512}, \frac{129}{32}]$. Then MetiTarski proves that the curve lies below $-\frac{12}{\pi}x - 18$ at $\frac{157}{128}$, $\frac{569}{256}$ and $\frac{1139}{512}$, and thus that it lies outside the exclusion region for $x \in [-\frac{5}{4}\pi, -\pi]$.

$$y < -\frac{12}{\pi}x - 18 \text{ at } \frac{157}{128}, \frac{569}{256}, \text{ and } \frac{1139}{512}$$

2. Maple calculates that the interval $[-\pi, -\frac{3}{4}\pi]$, in terms of x, corresponds to the interval $[\frac{57}{128}, \frac{629}{512}]$ in terms of w and then MetiTarski proves that

$$\forall w.\ \frac{57}{128} \geq w \vee w \geq \frac{629}{512} \Longrightarrow -\pi \geq x \vee x \geq -\frac{3}{4}\pi$$

Within this interval there are no points of inflection. The curve is convex for $w \in [\frac{57}{128}, \frac{629}{512}]$. MetiTarski proves that the curve lies below $\frac{12}{\pi}x + 6$ at $\frac{57}{128}$ and $\frac{629}{512}$, and thus it lies outside the exclusion region for $x \in [-\pi, -\frac{3}{4}\pi]$.

$$y < \frac{12}{\pi}x + 6 \text{ at } \frac{57}{128} \text{ and } \frac{629}{512}$$

3. Maple calculates that the interval $[-3, 3]$, in terms of y, corresponds to the interval $[0, \frac{101}{512}]$ in terms of w and then MetiTarski proves that

$$\forall w.\ w \geq \frac{101}{512} \Longrightarrow -3 \geq y \vee y \geq 3$$

Within this interval there are no points of inflection. The curve is convex for $w \in [0, \frac{101}{512}]$. MetiTarski proves that the curve lies above $-\frac{3}{4}\pi$ at $\frac{101}{512}$ and thus that it lies outside the exclusion region for $y \in [-3, 3]$.

$$-\frac{3}{4}\pi < x \text{ at } \frac{101}{512}$$

Analysis of an Inverted Pendulum that Fails to Meet its Requirements. Next a parameter of the inverted pendulum system is altered slightly and the system is re-analysed with respect to the same criteria. Given that the mass of the pendulum in the inverted pendulum system has the value 0.17, the open loop transfer function for the system is

$$G(s) = \frac{-4250(2s^2 - 7s + 2)}{1945s^3 + 355s^2 - 55811s - 8330}$$

and the gain and phase-shift can be calculated as follows:

$$y = 20 \log_{10} \left(\frac{425\sqrt{0.1w^{10} + 10.2w^8 + 214.0w^6 + 1290.9w^4 + 153.2w^2 + 2.7}}{37.8w^6 + 2172.3w^4 + 31207.8w^2 + 693.8} \right)$$

$$x = \begin{cases} -\arctan(\frac{105247w^3 + 3890w^5 - 169932w}{14325w^4 + 406627w^2 - 16660}) & \text{if } 0 \leq w < 0.202 \\ -\pi & \text{if } w = 0.202 \\ -\arctan(\frac{105247w^3 + 3890w^5 - 169932w}{14325w^4 + 406627w^2 - 16660}) - \pi & \text{if } 0.202 < w \end{cases}$$

We use Maple and MetiTarski to analyse this system with respect to the exclusion region criteria and prove that it fails to meet its requirements by providing a counter example as follows:

1. We first calculate using Maple that the interval $[-\frac{5}{4}\pi, -\pi]$, in terms of x, corresponds to the interval $[\frac{79}{64}, \frac{517}{128}]$ in terms of w and then use MetiTarski to show that

$$\forall w.\ \frac{79}{64} \geq w \vee w \geq \frac{517}{128} \Longrightarrow -\frac{5}{4}\pi \geq x \vee x \geq -\pi$$

Within this interval there is one point of inflection, which lies in the interval $[\frac{1059}{512}, \frac{265}{128}]$. The curve is convex for $w \in [\frac{79}{64}, \frac{1059}{512}]$ and concave for $w \in [\frac{256}{128}, \frac{517}{128}]$. MetiTarski proves that the curve lies below the line $-\frac{12}{\pi}x - 18$ at $\frac{79}{64}$ and $\frac{1059}{512}$.

$$y < -\frac{12}{\pi}x - 18 \text{ at } \frac{79}{64} \text{ and } \frac{1059}{512}$$

MetiTarski then proves that at $\frac{265}{128}$ the curve lies within the exclusion region and thus the Nichols plot fails to meet its requirements for $x \in [-\frac{5}{4}\pi, -\pi]$.

$$y \geq -\frac{12}{\pi}x - 18 \wedge y \leq \frac{12}{\pi}x + 18 \text{ at } \frac{256}{128}$$

2. Maple calculates that the interval $[-\pi, -\frac{3}{4}\pi]$, in terms of x, corresponds to the interval $[\frac{231}{512}, \frac{633}{512}]$ in terms of w and then MetiTarski proves that

$$\forall w.\ \frac{231}{512} \geq w \vee w \geq \frac{633}{512} \Longrightarrow -\pi \geq x \vee x \geq -\frac{3}{4}\pi$$

Within this interval there are no points of inflection. The curve is convex for $w \in [\frac{231}{512}, \frac{633}{512}]$. MetiTarski proves that at $\frac{57}{128}$ the curve lies within the exclusion region and thus the Nichols plot fails to meet its requirements for $x \in [-\pi, -\frac{3}{4}\pi]$.

$$y \geq \frac{12}{\pi}x + 6 \wedge y \leq -\frac{12}{\pi}x - 6 \text{ at } \frac{57}{128}$$

3. Maple calculates that the interval $[-3, 3]$, in terms of y, corresponds to the interval $[0, \frac{55}{256}]$ in terms of w and then MetiTarski proves that

$$\forall w.\ w \geq \frac{55}{256} \Longrightarrow -3 \geq y \vee y \geq 3$$

Within this interval there are no points of inflection. The curve is convex for $w \in [0, \frac{103}{512}]$ and concave for $w \in [\frac{13}{64}, \frac{55}{256}]$. MetiTarski proves that the curve lies above $-\frac{3}{4}\pi$ at $\frac{103}{512}$, $\frac{13}{64}$, and $\frac{55}{256}$, and thus that it lies outside the exclusion region for $y \in [-3, 3]$.

$$-\frac{3}{4}\pi < x \text{ at } \frac{103}{512}, \frac{13}{64}, \text{and } \frac{55}{256}$$

3.3 Magnetic Disk Drive Reader System

This section focuses on the modeling and analysis of a magnetic disk drive system [7] with respect to stability. Modern computers use magnetic disks to store data. A disk drive reader reads the data by positioning a reader head over a track on the disk. It consists of a controller (or amplifier), a motor, an arm and a read head. A metal spring (or flexure) holds the read head slightly above the disk. For a given set of parameter values, the open loop transfer function of the disk drive system is

$$G(s) = \frac{2.8 \times 10^{11} K_m}{(s+1000)s(s+20)(3s^2+30000+100000000).}$$

This system has an 'uncertain' parameter, namely the motor constant which is represented by the constant K_m and its value is known to lie within the interval $[120, 130]$. The gain and phase-shift of the system can be calculated as follows:

$$y = 20 \log_{10}\left(\frac{2.8 \times 10^{11} K_m}{\sqrt{9w^{10} + 3.09 \times 10^8 w^8 + 1.03 \times 10^{16} w^6 + 10^{22} w^4 + 4 \times 10^{24} w^2}}\right)$$

$$x = \begin{cases} -\arctan\left(\frac{-130660000w^2 + 3w^4 + 2 \times 10^{12}}{1140w(29w^2 - 90000000)}\right) - \pi & \text{if } 0 \le w < 1761.6 \\ -\pi & \text{if } w = 1761.6 \\ -\arctan\left(\frac{-130660000w^2 + 3w^4 + 2 \times 10^{12}}{1140w(29w^2 - 90000000)}\right) - 2\pi & \text{if } 1761.6 < w \end{cases}$$

Following a similar approach to the inverted pendulum, we have used Maple and MetiTarski to provide a symbolic analysis and formal proof. The system meets its requirements for all permissible parameter values. The three Nichols plot exclusion zones (recall Sect. 3.1) give rise to the following proof obligations:

$$\forall w.\ \frac{15839}{128} \ge w \vee w \ge \frac{354991}{512} \Longrightarrow -\frac{5}{4}\pi \ge x \vee x \ge -\pi$$
$$y < -\frac{12}{\pi}x - 18 \text{ at } \frac{15839}{128} \text{ and } \frac{354991}{512} \text{ for } K_m = 120 \text{ and } K_m = 130$$

$$\forall w.\ \frac{9745}{512} \ge w \vee w \ge \frac{63357}{512} \Longrightarrow -\pi \ge x \vee x \ge -\frac{3}{4}\pi$$
$$y < \frac{12}{\pi}x + 6 \text{ at } \frac{9745}{512} \text{ and } \frac{63357}{512} \text{ for } K_m = 120 \text{ and } K_m = 130$$

$$\forall w.\ \frac{1347}{128} \ge w \vee w \ge \frac{9601}{512} \Longrightarrow -3 \ge y \vee y \ge 3$$
$$-\frac{3}{4}\pi < x \text{ at } \frac{1347}{128} \text{ for } K_m = 120 \text{ and } K_m = 130$$

4 Hybrid Systems Verification

In order to examine the feasibility of verifying hybrid systems using MetiTarski, we developed the following procedure. It involves a number of manual steps, but they are essentially mechanical and could be automated.

1. Derive the hybrid automaton model of the system under investigation as an state diagram, including the number of locations with the corresponding parameters, the transition relation between different locations, and the system of differential equations governing the system in each location.
2. Starting from any particular location, we supply its system of ODEs and initial condition to Maple, and apply a Laplace transform to find an expression for the state variables of the system as an output function of time.
3. Using the transition relations, we use Maple to find the switching time from the first location to the next location. At this calculated time, we determine the values of all state variables using the time-dependent analytical expressions determined in the previous step, to find the final values of the state variables in location 1, and use them as the initial condition for the next state. We continue this procedure until we cover all reachable locations taking non-singleton initial sets of states into account.
4. Formulate the verification question as a safety property involving inequalities over the real-valued special functions.
5. Supply this first-order formula in TPTP format, including the corresponding axioms, as an input file to MetiTarski.

If MetiTarski is successful, it delivers a proof. Otherwise, it will probably run until terminated.

4.1 Collision Avoidance

We consider a cruise control system with automatic collision avoidance [16]. Let gap, v_f, v and a respectively represent the gap between the two cars, the velocity of the leading car, and the velocity and acceleration of the rear car. Then, the set of differential equations governing the system is

$$\dot{v} = a,\ \dot{a} = -3a - 3(v - v_f) + (gap - (v + 10)),\ \dot{gap} = v_f - v$$

Assuming the variable v_f is a parameter (unchanging symbolic constant), the dynamics of the system can be written as $\dot{x} = Ax + B$, where

$$x = \begin{bmatrix} v \\ v_f \\ a \\ gap \end{bmatrix} \quad A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -4 & 3 & -3 & 1 \\ -1 & 1 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 0 \\ -10 \\ 0 \end{bmatrix}$$

For the given set of initial states as $x_0 = (2, 2, -0.5, 1)^T$, the problem is to verify that rear car would never collide with the car in front, that is, always $gap > 0$.

Let X denote the Laplace transform of x ($X = \mathcal{L}\ x$), then $sX - x_0 = AX + \frac{B}{s}$, and solving for X we have $X = (sI - A)^{-1}(x_0 + \frac{B}{s})$. Using Maple we have

$$X = \begin{bmatrix} \dfrac{2s^3 + 5.5s^2 - 3s + 2}{s(s^3 + 3s^2 + 4s + 1.0)} \\ 2s^{-1} \\ \dfrac{-0.5s(22 + s)}{s^3 + 3s^2 + 4s + 1} \\ \dfrac{3s^2 + 4.5s + 12 + s^3}{s(s^3 + 3s^2 + 4s + 1)} \end{bmatrix}$$

Therefore, $gap = \mathcal{L}^{-1}\ \dfrac{3s^2 + 4.5s + 12 + s^3}{s\,(s^3 + 3s^2 + 4s + 1)}$, and using Maple for the inverse Laplace transform we have

$$gap = 12 - 14.2\mathrm{e}^{-0.318t} + 3.24\mathrm{e}^{-1.34t}\cos(1.16t) - 0.154\mathrm{e}^{-1.34t}\sin(1.16t).$$

MetiTarski proves that this expression is always greater than zero, and therefore the system is safe for the given initial conditions.

4.2 Navigation

This benchmark deals with an object (perhaps a vehicle, though the dynamics are not exactly vehicle dynamics) that moves in the $\mathbb{R}^2$ plane [9]. The desired velocity $\mathbf{v}_d$ is determined by the position of the object in an $n \times m$ grid, and the desired velocities may take values as follows:

$$\mathbf{v}_d = (v_{d1}(i), v_{d2}(i)) = (\sin(i \times \frac{\pi}{4}), \cos(i \times \frac{\pi}{4})),\ \text{for } i = 0, \ldots, 7$$

We assume that the length and the width of a cell is 1, and that the lower left corner of the grid is the origin. An example of a 3×3 grid is depicted in Fig. 3a, where the label i in each cell refers to the desired velocity. In addition, the grid contains cells labelled **A** that have to be reached and cells labelled **B** that ought to be avoided.

Given $\mathbf{v}_d$ the behavior of the actual velocity $\mathbf{v}$ is determined by the differential equation $\dot{\mathbf{v}} = C(\mathbf{v} - \mathbf{v}_d)$, where $C \in \mathbb{R}^{2\times 2}$ is assumed to have eigenvalues with strictly negative real part. This guarantees that the velocity will converge to the desired velocity. Figure 3b shows two trajectories, with $C = \begin{pmatrix} -1.2 & 0.1 \\ 0.1 & -1.2 \end{pmatrix}$. Both satisfy the property that **A** should be reached, and **B** avoided.

An instance of this benchmark is characterized by the initial condition on $\mathbf{x}$ and $\mathbf{v}$, by matrix C in the differential equation for $\mathbf{v}$ and by the map of the grid, which can be represented as $n \times m$ matrix with elements from $\{0, \ldots, 7\} \cup \{\mathbf{A}, \mathbf{B}\}$. For the example in Fig. 3 this matrix is $\begin{pmatrix} \mathbf{B} & 2 & 4 \\ 4 & 3 & 4 \\ 2 & 2 & \mathbf{A} \end{pmatrix}$.

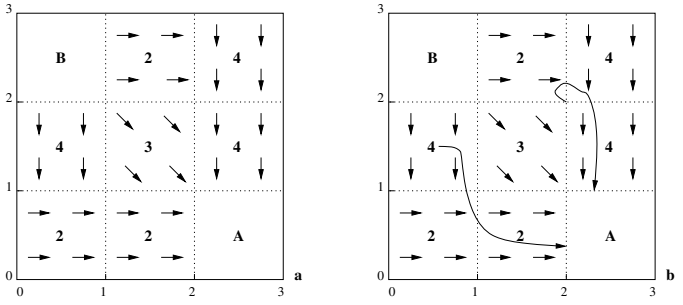


Fig. 3. a. The map determines the desired velocity of the moving object, depending on the position of the object. b. Two trajectories of objects moving in the plane. Both objects eventually reach cell **A** while avoiding **B**.

The dynamics of the 4-dimensional state vector $(x_1, x_2, v_1, v_2)^T$ are given by

$$\begin{pmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{v_1} \\ \dot{v_2} \end{pmatrix} = \begin{pmatrix} 0\,0 & 1 & 0 \\ 0\,0 & 0 & 1 \\ 0\,0 & -1.2 & 0.1 \\ 0\,0 & 0.1 & -1.2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ v_1 \\ v_2 \end{pmatrix} - \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ -1.2 & 0.1 \\ 0.1 & -1.2 \end{pmatrix} \begin{pmatrix} v_{d1}(i) \\ v_{d2}(i) \end{pmatrix}$$

The resulting time-deterministic hybrid system [4] is shown in Figure 4. The system has five locations.

1. Location ℓ_0, corresponds to cells labelled **B** that ought to be avoided.
2. Location ℓ_1, corresponds to $i = 2$ or $\mathbf{v}_d = (1, 0)$. Therefore, the differential equations of the system in this mode are

$$\dot{x_1} = v_1,\ \dot{x_2} = v_2,\ \dot{v_1} = -1.2v_1 + 0.1v_2 + 1.2,\ \dot{v_2} = 0.1v_1 - 1.2v_2 - 0.1. \quad (2)$$

3. Location ℓ_2, corresponds to $i = 3$ or $\mathbf{v}_d = (+0.707, -0.707)$. Therefore, the differential equations of the system in this mode are

$$\dot{x_1} = v_1,\ \dot{x_2} = v_2,\ \dot{v_1} = -1.2v_1 + 0.1v_2 + 0.919,\ \dot{v_2} = 0.1v_1 - 1.2v_2 - 0.919. \quad (3)$$

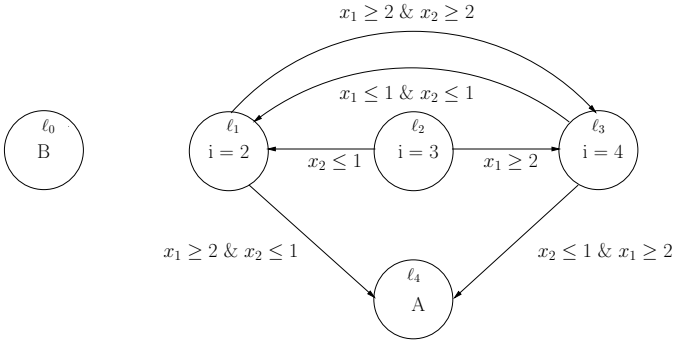


Fig. 4. The hybrid automaton model of the Navigation system

4. Location ℓ_3, corresponds to $i = 4$ or $\mathbf{v}_d = (0, -1)$. Therefore, the differential equations of the system in this mode are

$$\dot{x_1} = v_1,\ \dot{x_2} = v_2,\ \dot{v_1} = -1.2v_1 + 0.1v_2 + 0.1,\ \dot{v_2} = 0.1v_1 - 1.2v_2 - 1.2. \quad (4)$$

5. Location ℓ_4, corresponds to cells labelled **A** that have to be reached.

The transition relations between different locations are specified by logical formulas in Fig.4. Now, suppose we start from the initial states defined by $(0.5, 1.5, 0.1, 0)^T$, which means we are initially in location ℓ_3, and the differential equations governing the system are those described in equation (4). Using the Laplace transform method as described before, we can solve this system of ODEs using Maple to get the following closed form formulas for x_1 and x_2

$$x_1 = -0.5\mathrm{e}^{-1.1t} + 0.654 + 0.346\mathrm{e}^{-1.3t}$$
$$x_2 = -0.5\mathrm{e}^{-1.1t} + 2.35 - 0.346\mathrm{e}^{-1.3t} - t$$

More analysis with Maple shows that at $t = 1.12$, $x_1 = 1$. At this point we switch to location ℓ_1 with $i = 2$. We also use Maple to calculate the value of the other state variables at this time as $x_2 = 0.588$, $v_1 = 0.057$, and $v_2 = -0.735$. Therefore, the new initial states can be defined by $(1, 0.588, 0.057, -0.735)^T$, and the differential equations governing the system are those described in equation (2). Using the Laplace transform method as described before, we can solve this system of ODEs using Maple to derive formulas for x_1 and x_2:

$$x_1 = 0.742\mathrm{e}^{-1.1t} - 0.252 + 0.0974\mathrm{e}^{-1.3t} + t$$
$$x_2 = 0.736\mathrm{e}^{-1.1t} + 0.317 - 0.0538\mathrm{e}^{-1.3t}$$

We used MetiTarski to prove that in the first mode, for all values of time in the range $0 \leq t \leq 1$, we have $x_2 \leq 2$, and in the second mode, for all values of time in the range $0 \leq t$, we have $x_2 \leq 1$, and therefore, we verified that **B** cannot be reachable.

We have similarly verified safety properties of other hybrid system case studies such as the Room Heating and Mutant systems.

5 Conclusions

Our experiments demonstrate that problems arising in real-world applications can be tackled using a suitable automatic theorem prover. Table 2 shows the problems and runtimes for three categories of case studies: inverted pendulum, disk drive reader, and hybrid systems. The runtimes were measured on a 2.66 GHz Mac Pro running Poly/ML.

As can be seen from Table 2, there are different versions of the IPM and DDR problems which are related to the three intervals specifying the stability criteria of the Nichols plot of the systems. In each interval, we have proved several problems to guarantee that it meets or fails to meet its requirements.

Table 2. Problems with Runtimes in Seconds

IPM-1-1	8.4
IPM-1-2	0.2
IPM-1-3	0.4
IPM-1-5-w	0.4
IPM-2-1	0.1
IPM-2-2	5.3
IPM-2-3	0.4
IPM-2-5-w	0.4
IPM-3-1	0.1
IPM-3-2	0.2
IPF-1-1	29.4
IPF-1-2	0.2
IPF-1-3	0.6
IPF-1-5-w	2.7
IPF-2-1	0.2
IPF-2-2	23.3
IPF-2-3	0.6
IPF-3-1	0.1
IPF-3-2	0.2
Inverted Pendulum	

DDR-1-1	0.8
DDR-1-2	6.8
DDR-1-3	0.2
DDR-1-5	0.8
DDR-1-6-w	0.3
DDR-1-7-w	0.4
DDR-1-8-w	0.3
DDR-2-1	1.0
DDR-2-2	0.2
DDR-2-5-w	0.4
DDR-2-6-w	0.4
DDR-2-7-w	0.4
DDR-2-8-w	0.4
DDR-3-1	0.1
DDR-3-2	0.1
Disk Drive Reader	

Collision Avoidance	5.1
Room Heating	0.8
Navigation-1	0.2
Navigation-2	0.4
Mutant-1	0.1
Mutant-2	12.9
Mutant-3	67.9
Hybrid Systems	

Different versions of a hybrid systems problem correspond to different modes of operation for the corresponding system.

The formulas to be proved are complicated, containing many occurrences of special functions. On the other hand, and in contrast to our earlier problems from the world of mathematics, they often have great margins of error. Therefore, they can be tackled even if we use fairly crude approximations, which in turn makes proofs less taxing than they would be otherwise.

We still need to investigate how well our work scales to larger and nonlinear problems. There will clearly still be a place for the competitive approaches based on model checking and constraint solving. Nevertheless, a theorem proving approach is a suitable alternative, particularly when we require proofs and not merely claims of correctness.

Acknowledgements. The research was supported by the Engineering and Physical Sciences Research Council [grant number EP/C013409/1]. Ruth (Hardy) Letham, Hanne Gottliebsen, and Ursula Martin helped with the control systems problems. Stefan Ratschan, Zhikun She, and Mohamed Zaki helped with the hybrid systems problems. Figure 2 is obtained from `http://en.wikipedia.org/wiki/Inverted_pendulum`, and other figures are obtained from the corresponding references noted in each section.

References

1. Akbarpour, B., Paulson, L.: Towards Automatic Proofs of Inequalities Involving Elementary Functions. In: Pragmatics of Decision Procedures in Automated Reasoning (PDPAR), pp. 27–37 (2006)
2. Akbarpour, B., Paulson, L.: Extending a Resolution Prover for Inequalities on Elementary Functions. In: Dershowitz, N., Voronkov, A. (eds.) LPAR 2007. LNCS, vol. 4790, pp. 47–61. Springer, Heidelberg (2007)

3. Akbarpour, B., Paulson, L.: Metitarski: An Automatic Prover for the Elementary Functions. In: Autexier, S., Campbell, J., Rubio, J., Sorge, V., Suzuki, M., Wiedijk, F. (eds.) AISC 2008, Calculemus 2008, and MKM 2008. LNCS, vol. 5144, pp. 217–231. Springer, Heidelberg (2008)
4. Alur, R., Courcoubetis, C., Halbwaches, N., Henzinger, T.A., Ho, P.-H., Nicollin, X., Olibero, A., Sifakis, J., Yovine, S.: The Algorithmic Analysis of Hybrid Systems. Theoretical Computer Science 138, 3–34 (1995)
5. Asarin, E., Dang, T., Maler, O.: The d/dt Tool for Verification of Hybrid Systems. In: Brinksma, E., Larsen, K.G. (eds.) CAV 2002. LNCS, vol. 2404, pp. 365–370. Springer, Heidelberg (2002)
6. Chutianan, A., Krogh, B.H.: Computational Techniques for Hybrid System Verification. IEEE Transactions on Automatic Control 48(1), 64–75 (2003)
7. Dorf, R.C., Bishop, R.H.: Modern Control Systems. Prentice-Hall, Englewood Cliffs (2001)
8. Duarte, L., Duarte, S., da Mota, L., Skea, J.: An Extension of the Prelle-Singer Method and a Maple Implementation. Computer Physics Communications 144(1), 46–62 (2002)
9. Frehse, G.: PHAVer: Algorithmic Verification of Hybrid Systems Past HyTech. In: Morari, M., Thiele, L. (eds.) HSCC 2005. LNCS, vol. 3414, pp. 258–273. Springer, Heidelberg (2005)
10. Hardy, R.: Formal Methods for Control Engineering: A Validated Decision Procedure for Nichols Plot Analysis. PhD thesis, St. Andrews University (2006)
11. Henzinger, T.A., Ho, P.H., Wong-Ti, H.: HyTech: A Model Checker for Hybrid Systems. Software Tools for Technology Transfer 1(1-2), 110–122 (1997)
12. Prelle, M.S.M.: Elementary First Integrals of Differential Equations. Transactions of the American Mathematical Society 279(1), 215–229 (1983)
13. Man, Y.: Computing closed form solutions of first order odes using the prelle-singer procedure. J. Symb. Comput. 16(5), 423–443 (1993)
14. Ratschan, S., She, Z.: Safety Verification of Hybrid Systems by Constraint Propagation-Based Abstraction Refinement. ACM Transactions on Embedded Computing Systems 6(1) (2007)
15. Ratschan, S., She, Z.: Benchmarks for Safety Verification of Hybrid Systems, June 13 (2008), http://hsolver.sourceforge.net/benchmarks
16. Tiwari, A.: Approximate Reachability for Linear Systems. In: Maler, O., Pnueli, A. (eds.) HSCC 2003. LNCS, vol. 2623, pp. 514–525. Springer, Heidelberg (2003)

Three-Dimensional Kneed Bipedal Walking: A Hybrid Geometric Approach

Aaron D. Ames, Ryan W. Sinnet, and Eric D.B. Wendel

Department of Mechanical Engineering
Texas A&M University, College Station, TX
aames@tamu.edu, {rsinnet,ericdbw}@neo.tamu.edu

Abstract. A 3D biped with knees and a hip is naturally modeled as a nontrivial hybrid system; impacts occur when the knee strikes and when the foot impacts the ground causing a switch in the dynamics governing the system. Through a variant of geometric reduction—termed *functional Routhian reduction*—we can reduce the dynamics on each domain of this hybrid system to obtain a planar equivalent biped. Using preexisting techniques for obtaining walking gaits for 2D bipeds, and utilizing the decoupling effect afforded by the reduction process, we design control strategies that result in stable walking gaits for the 3D biped. That is, the main result of this paper is a control law that results in 3D bipedal walking obtained through stable walking gaits for the equivalent 2D biped.

1 Introduction

Adding knees to a bipedal robot is important from both a practical and theoretical perspective: knees allow for an increase in energy efficiency and for the ability to navigate rough terrain more robustly. Yet adding knees significantly adds to the complexity of analyzing and controlling the biped [4], [13]. To see this, note that bipedal robots are naturally modeled as hybrid systems; when the foot impacts the ground, there is an instantaneous change in the velocity of the system. Adding locking knees to the robot results in an even more complex hybrid model since at knee lock there is another instantaneous change in the velocity of the system. Moreover, this necessarily results in two sets of dynamical equations: one where the knee is unlocked and one where the knee is locked. Kneed walking, therefore, provides significant novel challenges, especially when coupled with the desire for three-dimensional bipedal walking.

Three-dimensional (3D) bipedal walking provides interesting challenges not found in its two-dimensional (2D) counterpart. In this case one must not only achieve stable forward motion, but simultaneously stabilize the walker upright during this motion. In addition, while 2D bipedal walking has been well-studied (see [6], [7], [12], [18] and [14] to name a few), the results in 3D bipedal walking are relatively limited (see [5], [9] and [8] for some results in 3D walking) and there have yet to be results on obtaining walking for 3D bipedal robots with knee locking. Coupling the study of 3D bipeds with the study of locking knees,

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 16–30, 2009.

therefore, forms a challenge that will test our understanding of the underlying mechanisms of walking.

Fundamental to understanding 3D walking—with or without knees—is understanding the interplay between the lateral and sagittal dynamics. That is, we must mathematically quantify how to "decouple" the dynamics of a 3D biped into its sagittal and lateral components; this is done by exploiting inherent symmetries in walkers through the use of geometric reduction. Specifically, we consider a form of geometric reduction termed *functional Routhian reduction* (first introduced in [3] and generalized in [2]). As with classical reduction, this form of reduction utilizes symmetries in a system, in the form of "cyclic" variables, to reduce the dimensionality of the system. Unlike classical reduction, this is done by setting the conserved quantities equal to an arbitrary function of the "cyclic" variables rather than a constant, i.e., there is a *functional conserved quantity*. This allows us to "control" the decoupling effect of geometric reduction through this function, a fact that will be instrumental in the construction of our control law.

The main result of this paper is a control law that results in stable walking for a 3D biped with knees and a hip, which is achieved by combining three control laws. The first control law affects the sagittal dynamics of the biped by shaping the potential energy so that the 2D biped, obtained by constraining the 3D biped to the sagittal plane, has stable walking gaits. The second control law shapes the total energy of the 3D biped so that functional Routhian reduction can be applied—the reduced system is exactly the 2D system after applying the first control law—thus decoupling the sagittal and lateral dynamics, while allowing us to affect the lateral dynamics through our specific choice of the functional conserved quantity, for *certain initial conditions*. Finally, the third control law stabilizes to the surface of initial conditions for which the decoupling afforded by the second control law is valid. We verify numerically that the combined control law results in stable walking, i.e., a locally exponentially stable periodic orbit.

2 Bipedal Model

Hybrid systems are systems that display both continuous and discrete behavior and so bipedal walkers are naturally modeled by systems of this form; the continuous component consists of the dynamics dictated by Lagrangians modeling mechanical systems in different domains, and the discrete component consists of the impact equations which instantaneously change the velocity of the system when the knees lock or when the foot contacts the ground. This section, therefore, introduces the basic terminology of hybrid systems and introduces the hybrid model of the biped considered in this paper.

Definition 1. *A* hybrid control system *is a tuple*

$$\mathscr{HC} = (\Gamma, D, U, G, R, FG),$$

where

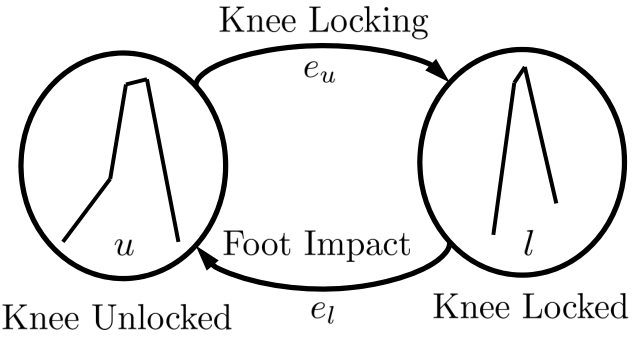


Fig. 1. A graphical representation of the domains of the hybrid control system $\mathscr{HC}_{3D}$

- $\Gamma = (V, E)$ *is an oriented graph, i.e.,* V *and* E *are a set of vertices and edges, respectively, and there exists a source function* $\mathrm{sor} : E \to V$ *and a target function* $\mathrm{tar} : E \to V$ *which associates to an edge its source and target, respectively.*
- $D = \{D_v\}_{v \in V}$ *is a set of domains, where* $D_v \subseteq \mathbb{R}^{n_v}$ *is a smooth submanifold of* $\mathbb{R}^{n_v}$*,*
- $U = \{U_v\}_{v \in V}$*, where* $U_v \subset \mathbb{R}^{k_v}$ *is a set of admissible controls,*
- $G = \{G_e\}_{e \in E}$ *is a set of guards, where* $G_e \subseteq D_{\mathrm{sor}(e)}$*,*
- $R = \{R_e\}_{e \in E}$ *is a set of reset maps, where* $R_e : G_e \to D_{\mathrm{tar}(e)}$ *is a smooth map,*
- $FG = \{(f_v, g_v)\}_{v \in E}$*, where* (f_v, g_v) *is a control system on* D_v*, i.e.,* $\dot{x} = f_v(x) + g_v(x)u$ *for* $x \in D_v$ *and* $u \in U_v$*.*

A hybrid system $\mathscr{H} = (\Gamma, D, G, R, F)$ *is a hybrid control system with* $U = \{0\}$*, in which case* $F = \{f_v\}_{v \in E}$*.*

Solutions to hybrid systems, or *hybrid flows* or *hybrid executions*, are defined in the traditional manner (see [10]). A solution to a hybrid system is k-periodic if it returns to the same point after passing through the domain in which it is contained k times (in the process it may pass through an arbitrary number of other domains of the hybrid system). One can consider the local exponential stability of k-periodic solutions in the obvious way (see [2] for this definition in the case of a hybrid system with one domain). One can associate to a k-periodic solution of a hybrid system a Poincare map, and the stability of the k-periodic solution can be determined by considering the stability of the Poincare map. Finally, the stability can be determined numerically using approximations of the Jacobian of the Poincare map (see [14] and [15]). This is how we will determine that the periodic orbit for the 3D biped produced in this paper is stable.

3D biped model. The model of interest is a controlled bipedal robot with a hip, knees and splayed legs that walks on flat ground in three dimensions (see Figure 2), from which we will explicitly construct the hybrid control system:

$$\mathscr{HC}_{3D} = (\Gamma_{3D}, D_{3D}, U_{3D}, G_{3D}, R_{3D}, FG_{3D}).$$

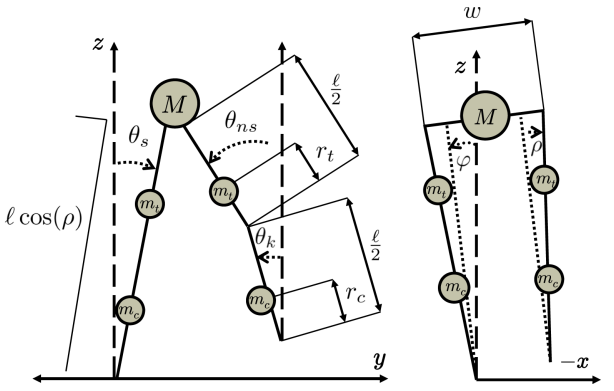


Fig. 2. The sagittal and lateral planes of a three-dimensional bipedal robot

In particular, $\Gamma_{3\mathrm{D}} = (\{u, l\}, \{e_u = (u, l), e_l = (l, u)\})$. That is, there are two domains u, l and two edges e_u, e_l (see Figure 1). In the first domain the biped's non-stance knee is unlocked and in the second domain the biped's knee is locked. Transitions occur from domain u to domain l when the knee locks, and from l to u when the foot strikes the ground. Note that the discrete structure of this model enforces temporal ordering to events (kneelock and footstrike) as motivated by the two-dimensional biped with knees considered in [4]. We will now construct the rest of the hybrid system $\mathscr{HC}_{3\mathrm{D}}$ beginning on the level of Lagrangians and constraint functions (see [1,2,3]).

Associated with each domain, there is a configuration space $Q_u^{3\mathrm{D}} = \mathbb{T}^3 \times \mathbb{S}^1$ and $Q_l^{3\mathrm{D}} = \mathbb{T}^2 \times \mathbb{S}^1$ associated with the knee being unlocked and locked, respectively. The coordinates on $Q_u^{3\mathrm{D}}$ are given by $q_u = (\theta_u^T, \varphi)^T$, with $\theta_u = (\theta_\mathrm{s}, \theta_\mathrm{ns}, \theta_\mathrm{k})^T$ the vector of sagittal-plane variables with the knee unlocked, where θ_s is the angle of the stance leg from vertical, θ_ns is the angle of the non-stance leg from vertical and θ_k is the angle of the knee from vertical (see Figure 2), and φ is the lean (or roll) from vertical. Similarly, the coordinates on $Q_l^{3\mathrm{D}}$ are given by $q_l = (\theta_l^T, \varphi)^T$, where $\theta_l = (\theta_\mathrm{s}, \theta_\mathrm{ns})^T$ is again the vector of sagittal-plane variables with the knee locked. Note that the hip width w, leg length ℓ, and leg splay angle ρ are held constant.

Each domain and guard are constructed from constraint functions. For the knee unlocked domain, the unilateral constraint is given by:

$$H_u^{3\mathrm{D}}(q_u) = \theta_\mathrm{k} - \theta_\mathrm{ns},$$

which is positive when the knee is unlocked and zero at kneestrike. For the knee locked domain, the unilateral constraint is given by:

$$H_l^{3\mathrm{D}}(q_l) = \ell\cos(\rho)\left(\cos(\theta_s) - \cos(\theta_{ns})\right)\cos(\varphi) + (w - 2\sin(\rho))\sin(\varphi),$$

which gives the height of the non-stance foot above the ground. Thus the domains for the hybrid system $D_{3\mathrm{D}} = \{D_u^{3\mathrm{D}}, D_l^{3\mathrm{D}}\}$ are obtained by requiring that the constraint functions be positive, i.e., for $i \in \{u, l\}$,

$$D_i^{3\mathrm{D}} = \left\{ \begin{pmatrix} q_i \\ \dot{q}_i \end{pmatrix} \in TQ_i^{3\mathrm{D}} : H_i^{3\mathrm{D}}(q_i) \geq 0 \right\}.$$

We put no restrictions on the set of admissible controls except that they can only directly affect the angular accelerations. Therefore, $U_{3\mathrm{D}} = \{U_u^{3\mathrm{D}}, U_l^{3\mathrm{D}}\}$ with $U_u^{3\mathrm{D}} = \mathbb{R}^4$ and $U_l^{3\mathrm{D}} = \mathbb{R}^3$.

The set of guards is given by $G_{3\mathrm{D}} = \{G_{e_u}^{3\mathrm{D}}, G_{e_l}^{3\mathrm{D}}\}$ where $G_{e_u}^{3\mathrm{D}}$ is the set of states where the leg is locking and $G_{e_l}^{3\mathrm{D}}$ is the set of states in which the height of the swing foot is zero and infinitesimally decreasing. That is, for $i \in \{u, l\}$,

$$G_{e_i}^{3\mathrm{D}} = \left\{ \begin{pmatrix} q_i \\ \dot{q}_i \end{pmatrix} \in TQ_i^{3\mathrm{D}} : H_i^{3\mathrm{D}}(q_i) = 0, \quad dH_i^{3\mathrm{D}}(q_i)\dot{q}_i < 0 \right\},$$

with $dH_i^{3\mathrm{D}}(q_i) = \left(\frac{\partial H_i^{3\mathrm{D}}}{\partial q_i}(q_i) \right)^T$.

The set of reset maps is given by $R_{3\mathrm{D}} = \{R_{e_u}^{3\mathrm{D}}, R_{e_l}^{3\mathrm{D}}\}$. The reset map $R_{e_u}^{3\mathrm{D}}$ is given by

$$R_{e_u}^{3\mathrm{D}}(q_u, \dot{q}_u) = \begin{pmatrix} q_l \\ P(q_u, \dot{q}_u)_1 \\ P(q_u, \dot{q}_u)_2 \\ P(q_u, \dot{q}_u)_4 \end{pmatrix}$$

where

$$P(q_u, \dot{q}_u) = \dot{q}_u - \frac{dH_u^{3\mathrm{D}}(q_u)\dot{q}_u}{dH_u^{3\mathrm{D}}(q_u) M_u^{3\mathrm{D}}(q_u)^{-1} dH_u^{3\mathrm{D}}(q_u)^T} M_u^{3\mathrm{D}}(q_u)^{-1} dH_u^{3\mathrm{D}}(q_u)^T$$

with $M_u^{3\mathrm{D}}(q_u)$ the inertia matrix given in (1). This reset map models a perfectly plastic impact at the knee.

The reset map $R_{e_l}^{3\mathrm{D}}$ similarly models a perfectly plastic impact at the foot. This is obtained through the same process outlined in [3] and [7] (see [4] for a nice explanation of the computation of the impact equations for a 2D kneed walker) but space constraints prevent the inclusion of this equation. Also, note that the signs of w and ρ are flipped during impact to model the change in stance leg.

Finally, the dynamics for $\mathscr{HC}_{3\mathrm{D}}$ are obtained from the Euler-Lagrange equations for the two mechanical systems in each domain. Specifically, the Lagrangian describing each system is given by, for $i \in \{u, l\}$,

$$L_i^{3\mathrm{D}}(q_i, \dot{q}_i) = \frac{1}{2}\dot{q}_i^T M_i^{3\mathrm{D}}(q_i)\dot{q}_i - V_i^{3\mathrm{D}}(q_i),$$

where $M_i^{3\mathrm{D}}(q_i)$ is the inertial matrix and $V_i^{3\mathrm{D}}(q_i)$ is the potential energy (these are large matrices and so space constraints prevent the inclusion of them in this paper), where $M_i^{3\mathrm{D}}(q_i)$ can be expressed in block matrix form as follows:

$$M_i^{3\mathrm{D}}(q_i) = \begin{pmatrix} M_i^{\theta}(\theta_i) & M_i^{\varphi,\theta}(\theta_i)^T \\ M_i^{\varphi,\theta}(\theta_i) & m_i^{\varphi}(\theta_i) \end{pmatrix}, \tag{1}$$

where $M_i^{3D}(q_i) \in \mathbb{R}^{n_i \times n_i}$, $M_i^{\theta}(\theta_i) \in \mathbb{R}^{(n_i-1)\times(n_i-1)}$, $M_i^{\varphi,\theta}(\theta_i) \in \mathbb{R}^{1\times(n_i-1)}$ and $m_i^{\varphi}(\theta_i) \in \mathbb{R}$ where $n_u = 4$ and $n_l = 3$. The reason for this block matrix representation will become clear when the control laws are introduced.

Using the controlled Euler-Lagrange equations, the dynamics for the walker are given by:

$$M_i^{3D}(q_i)\ddot{q}_i + C_i^{3D}(q_i, \dot{q}_i)\dot{q}_i + N_i^{3D}(q_i) = B_i^{3D} v_i,$$

where $v_i \in U_i^{3D}$, $C_i^{3D}(q_i, \dot{q}_i)$ is the Coriolis matrix, $N_i^{3D} = \frac{\partial V_i^{3D}}{\partial q_i}(q_i)$, and

$$B_l^{3D} = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \qquad B_u^{3D} = \begin{pmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

which converts the torque from relative coordinates to absolute.

Thus for $FG_{3D} = \{FG_u^{3D}, FG_l^{3D}\}$, we have for $i \in \{u, l\}$

$$f_i^{3D}(q_i, \dot{q}_i) = \begin{pmatrix} \dot{q}_i \\ M_i^{3D}(q)^{-1}\left(-C_i^{3D}(q_i, \dot{q}_i)\dot{q}_i - N_i^{3D}(q_i)\right) \end{pmatrix},$$
$$g_i^{3D}(q_i, \dot{q}_i) = \begin{pmatrix} 0_{n_i \times n_i} \\ M_i^{3D}(q_i)^{-1} B_i^{3D} \end{pmatrix},$$

where $0_{n_i \times n_i}$ is a $n_i \times n_i$ matrix of zeros.

3 Control Law Construction

This section presents the control law for the 3D biped with a knee and hip, the construction of which is motivated by the control law for the 3D biped (without a knee) successfully utilized in [2]. In particular, the control law is obtained by combining three control laws on *each* domain, u and l, of the hybrid system. The first control law acts on the sagittal dynamics of the walker on each domain in a way analogous to the controlled symmetries control law used for 2D walkers, the second control law transforms the Lagrangians of the 3D walker into almost-cyclic Lagrangians so that we can utilize *functional Routhian reduction* (see [2]), and the third control law utilizes zero dynamics techniques to stabilize to the set of initial conditions where the decoupling effect afforded by functional Routhian reduction is in effect. The result of combining these control laws is a control law on each domain of the hybrid system that results in stable walking; the specific attributes of this walking will be discussed in the next section.

Reduced dynamics controller. The first control law affects the dynamics of the 3D biped's sagittal plane by shaping the potential energy of the Lagrangian describing these dynamics on each domain of the hybrid system as motivated by the controlled symmetries method of [17]. The end result is a hybrid system modeling the 2D dynamics of the biped that walks on flat ground.

We can view the 2D sagittal restriction of the 3D biped as a hybrid control system:

$$\mathscr{HC}_{2\mathrm{D}} = (\Gamma_{3\mathrm{D}}, D_{2\mathrm{D}}, U_{2\mathrm{D}}, G_{2\mathrm{D}}, R_{2\mathrm{D}}, FG_{2\mathrm{D}})$$

where $\Gamma_{2\mathrm{D}} = \Gamma_{3\mathrm{D}}$. To obtain this hybrid system we consider two configuration spaces $Q_u^{2\mathrm{D}} = \mathbb{T}^3$ and $Q_l^{2\mathrm{D}} = \mathbb{T}^2$ with coordinates $\theta_u = (\theta_{\mathrm{s}}, \theta_{\mathrm{ns}}, \theta_{\mathrm{k}})^T$ and $\theta_l = (\theta_{\mathrm{s}}, \theta_{\mathrm{ns}})^T$ and let

$$D_i^{2\mathrm{D}} = \left\{ \begin{pmatrix} \theta_i \\ \dot{\theta}_i \end{pmatrix} \in TQ_i^{2\mathrm{D}} : H_i^{2\mathrm{D}}(\theta_i) \geq 0 \right\},$$
$$G_{e_i}^{2\mathrm{D}} = \left\{ \begin{pmatrix} \theta_i \\ \dot{\theta}_i \end{pmatrix} \in TQ_i^{2\mathrm{D}} : H_i^{2\mathrm{D}}(\theta_i) = 0, dH_i^{2\mathrm{D}}(\theta_i)\dot{\theta}_i < 0 \right\},$$

for $i \in \{u, l\}$, with $H_i^{2\mathrm{D}}(\theta_i) = H_i^{3\mathrm{D}}(\theta_i, 0)$. We obtain the reset maps $R_{e_u}^{2\mathrm{D}}$ and $R_{e_l}^{2\mathrm{D}}$ by similarly projecting the reset maps to the $\varphi = 0$ subspace. For the set of admissible controls, we take $U_u^{2\mathrm{D}} = \mathbb{R}^3$ and $U_l^{2\mathrm{D}} = \mathbb{R}^2$. Finally, the dynamics $(f_i^{2\mathrm{D}}, g_i^{2\mathrm{D}})$, $i \in \{u, l\}$, are obtained from the Lagrangians given by:

$$L_i^{2\mathrm{D}}(\theta_i, \dot{\theta}_i) = \frac{1}{2}\dot{\theta}_i^T M_i^{2\mathrm{D}}(\theta_i)\dot{\theta}_i - V_i^{\theta}(\theta_i),$$

where $M_i^{2\mathrm{D}} = M_i^{\theta}$ as in (1) and $V_i^{2\mathrm{D}}(\theta_i) = V_i^{3\mathrm{D}}(\theta_i, 0)$, through the Euler Lagrange equations as was done in the 3D model, where in this case $B_u^{2\mathrm{D}}$ and $B_l^{2\mathrm{D}}$ are the 3×3 and 2×2 upper-left submatrices of $B_u^{3\mathrm{D}}$ and $B_l^{3\mathrm{D}}$, respectively.

The hybrid control system $\mathscr{HC}_{2\mathrm{D}}$ is similar, but not equivalent, to the typical 2D kneed walker (cf. [4]) (since the splayed legs affects the height of the planar robot) which motivates the control law to be introduced. That is, we utilize controlled symmetries of [17] by "rotating the world" via a group action in order to shape the potential energy of both $L_u^{2\mathrm{D}}$ and $L_l^{2\mathrm{D}}$ to obtain stable walking gaits on flat ground for $\mathscr{HC}_{2\mathrm{D}}$.

Consider the group action $\Psi_i : \mathbb{S}^1 \times Q_i^{2\mathrm{D}} \to Q_i^{2\mathrm{D}}$, $i \in \{u, l\}$, given by:

$$\Psi_l^{\gamma}(\theta_l) := \begin{pmatrix} \theta_{\mathrm{s}} + \gamma \\ \theta_{\mathrm{ns}} + \gamma \end{pmatrix}, \qquad \Psi_u^{\gamma}(\theta_u) := \begin{pmatrix} \theta_{\mathrm{s}} + \gamma \\ \theta_{\mathrm{ns}} + \gamma \\ \theta_{\mathrm{k}} + \gamma \end{pmatrix}$$

for slope angle $\gamma \in \mathbb{S}^1$. Using this, define the following two feedback control laws:

$$KR_i^{\gamma}(\theta_i) := (B_i^{2\mathrm{D}})^{-1} \left(\frac{\partial V_i^{2\mathrm{D}}}{\partial \theta_i}(\theta_i) - \frac{\partial V_i^{2\mathrm{D}}}{\partial \theta_i}(\Psi_i^{\gamma}(\theta_i)) \right), \tag{2}$$

for $i \in \{u, l\}$. Applying these control laws to the control systems $(f_i^{2\mathrm{D}}, g_i^{2\mathrm{D}})$ yields the dynamical systems:

$$f_i^{\gamma}(\theta_i, \dot{\theta}_i) := f_i^{2\mathrm{D}}(\theta_i, \dot{\theta}_i) + g_i^{2\mathrm{D}}(\theta_i, \dot{\theta}_i) KR_i^{\gamma}(\theta_i),$$

which are just the vector fields associated to the Lagrangians

$$L_i^{\gamma}(\theta_i, \dot{\theta}_i) = \frac{1}{2}\dot{\theta}_i^T M_i^{2\mathrm{D}}(\theta_i)\dot{\theta}_i - V_i^{2\mathrm{D}}(\Psi_i^{\gamma}(\theta_i)). \tag{3}$$

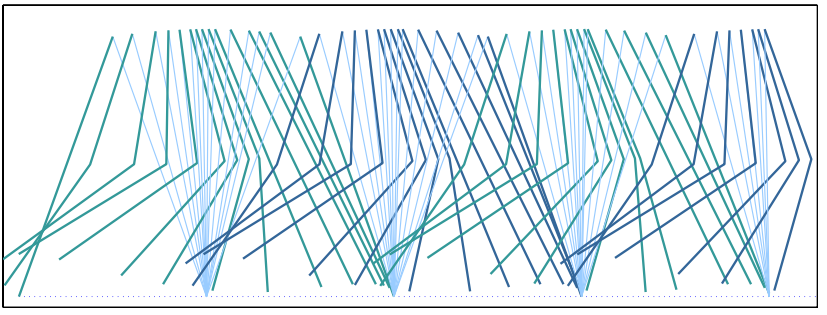


Fig. 3. A walking gait of the 2D biped obtained by restricting the 3D biped

We have thus defined a hybrid control system:

$$\mathscr{H}_{2\mathrm{D}}^{\gamma} := (\Gamma_{2\mathrm{D}}, D_{2\mathrm{D}}, G_{2\mathrm{D}}, R_{2\mathrm{D}}, F^{\gamma}),$$

where $F^{\gamma} = \{f_u^{\gamma}, f_l^{\gamma}\}$.

As with the typical kneed 2D biped, it can be verified that for certain γ, this hybrid system has a stable periodic orbit. An example of the 2D walking that is obtained for this 2D biped under this control law can be seen in Figure 3. Note that for this simulation, $\gamma = 0.0504$ (and the same model constants as used in Section 4) motivated by [4].

Lagrangian shaping controllers. The fundamental tool used in the construction of the second control law is *functional Routhian reduction* (see [2] for a complete discussion of this type of reduction). This is a variant of standard Routhian reduction [11], and allows one to reduce the dimensionality of dynamical systems obtained from "almost-cyclic" Lagrangians. Moreover, it differs from standard reduction techniques in that one can set the cyclic variables equal to a function, rather than a constant, thus affecting the behavior of these cyclic variables. This type of reduction is fundamental in the construction of our control law since the cyclic variable is the lean angle, so applying this reduction allows for the decomposition of the walker into its sagittal and lateral components.

More concretely, we introduce controllers to shape both the kinetic and potential energy of $L_i^{3\mathrm{D}}$, $i \in \{u, l\}$ so as to render them "almost-cyclic." This shaping is done so that the functional Routhians (the Lagrangians for the reduced systems) associated with these almost-cyclic Lagrangians are just the Lagrangians for the 2D kneed walker considered in the construction of the first control laws.

Consider the following almost-cyclic Lagrangians for $i \in \{u, l\}$:

$$L_i^{(\alpha,\gamma)}(\theta_i, \varphi, \dot{\theta}_i, \dot{\varphi}) = \frac{1}{2}\begin{pmatrix}\dot{\theta}_i^T & \dot{\varphi}\end{pmatrix} M_i^{\alpha}(\theta_i)\begin{pmatrix}\dot{\theta}_i \\ \dot{\varphi}\end{pmatrix} - W_i^{\alpha}(\theta_i, \varphi, \dot{\theta}_i) - V_i^{(\alpha,\gamma)}(\theta_i, \varphi),$$

where

$$M_i^{\alpha}(\theta_i) = \begin{pmatrix} M_i^{2\mathrm{D}}(\theta_i) + \frac{M_i^{\varphi,\theta}(\theta_i)^T M_i^{\varphi,\theta}(\theta_i)}{m_i^{\varphi}(\theta_i)} & M_i^{\varphi,\theta}(\theta_i)^T \\ M_i^{\varphi,\theta}(\theta_i) & m_i^{\varphi}(\theta_i) \end{pmatrix}$$

$$W_i^{\alpha}(\theta_i, \varphi, \dot{\theta}_i) = -\frac{\alpha\varphi}{m_i^{\varphi}(\theta_i)} M_i^{\varphi,\theta}(\theta_i)\dot{\theta}_i$$

$$V_i^{(\alpha,\gamma)}(\theta_i, \varphi) = V_i^{2\mathrm{D}}(\Psi_i^{\gamma}(\theta_i)) - \frac{1}{2}\frac{\alpha^2\varphi^2}{m_i^{\varphi}(\theta_i)}$$

with $M_i^{\varphi,\theta}(\theta_i)$, $M_i^{2\mathrm{D}}(\theta_i) = M_i^{\theta}(\theta_i)$, and $m_i^{\varphi}(\theta_i)$ as defined in (1)—the last two are positive definite since $M_i^{3\mathrm{D}}(q_i) > 0$. Referring to [2], for these almost-cyclic Lagrangians, we have taken $\lambda(\varphi) = -\alpha\varphi$. It follows that the functional Routhians associated with these cyclic Lagrangians are L_i^{γ} as given in (3).

Now we can define two feedback control laws that transform $L_i^{3\mathrm{D}}$ to $L_i^{(\alpha,\gamma)}$. In particular, for $i \in \{u, l\}$, let

$$\begin{aligned} KS_i^{(\alpha,\gamma)}(q_i, \dot{q}_i) := (B_i^{3\mathrm{D}})^{-1}(C_i^{3\mathrm{D}}(q_i, \dot{q}_i)\dot{q}_i + N_i^{3\mathrm{D}}(q_i) \\ + M_i^{3\mathrm{D}}(q_i)M_i^{\alpha}(q_i)^{-1}(-C_i^{\alpha}(q_i, \dot{q}_i)\dot{q}_i - N_i^{(\alpha,\gamma)}(q_i))), \end{aligned} \tag{4}$$

where C_i^{α} is the shaped Coriolis matrix and $N_i^{(\alpha,\gamma)} = \frac{\partial V_i^{(\alpha,\gamma)}}{\partial q_i}$. Note that these control laws implicitly use the two first control laws. Applying these to the control systems $(f_i^{3\mathrm{D}}, g_i^{3\mathrm{D}})$ yields the dynamic systems:

$$f_i^{(\alpha,\gamma)}(q_i, \dot{q}_i) := f_i^{3\mathrm{D}}(q_i, \dot{q}_i) + g_i^{3\mathrm{D}}(q_i, \dot{q}_i)KS_i^{(\alpha,\gamma)}(q_i, \dot{q}_i), \tag{5}$$

which are just the vector fields associated to the Lagrangians $L_i^{(\alpha,\gamma)}$. Moreover we have the following relationship between the behavior of $f_i^{\alpha,\gamma}$ and f_i^{γ} on each domain of the hybrid system; this result follows directly from Theorem 1 in [2].

Theorem 1. *Let $i \in \{u, l\}$, then $(\theta_i(t), \varphi(t), \dot{\theta}_i(t), \dot{\varphi}(t))$ is a solution to the vector field $f_i^{(\alpha,\gamma)}$ on $[t_0, t_F]$ with*

$$\dot{\varphi}(t_0) = \frac{-1}{m_i^{\varphi}(\theta_i(t_0))}(\alpha\varphi(t_0) + M_i^{\varphi,\theta}(\theta_i(t_0))\dot{\theta}_i(t_0)), \tag{6}$$

if and only if $(\theta_i(t), \dot{\theta}_i(t))$ is a solution to the vector field f_i^{γ} and $(\varphi(t), \dot{\varphi}(t))$ satisfies:

$$\dot{\varphi}(t) = \frac{-1}{m_i^{\varphi}(\theta_i(t))}(\alpha\varphi(t) + M_i^{\varphi,\theta}(\theta_i(t))\dot{\theta}_i(t)). \tag{7}$$

This result implies that on each domain, for certain initial conditions, i.e., those satisfying (6), the dynamics of the biped can effectively be decoupled into the sagittal and lateral dynamics. Moreover, according to (7), the lateral dynamics must evolve in a very specific fashion. These fundamental points will allow us to use the walking gait for the 2D biped obtained by restricting the biped to obtain walking gaits for the 3D biped. But first, we must address how to handle situations where (6) is not satisfied.

Zero dynamics controller. The decoupling effect of Theorem 1 can only be enjoyed when (6) is satisfied; this set of initial conditions forms a hypersurface in each domain. Since most initial conditions will not satisfy this constraint, i.e., lie on this surface, we will use the classical method of output linearization in non-linear systems to stabilize to this hypersurface (see [16] for the continuous case and [7], [14] for the hybrid analogue).

Before introducing the third control law, we define a new hybrid control system that implicitly utilizes the first two control laws. Specifically, let

$$\mathscr{HC}_{3\mathrm{D}}^{(\alpha,\gamma)} = (\Gamma_{3\mathrm{D}}, D_{3\mathrm{D}}, \mathbb{R}, G_{3\mathrm{D}}, R_{3\mathrm{D}}, FG^{(\alpha,\gamma)})$$

where $\Gamma_{3\mathrm{D}}$, $D_{3\mathrm{D}}$, $G_{3\mathrm{D}}$ and $R_{3\mathrm{D}}$ are defined as for $\mathscr{HC}_{3\mathrm{D}}$ and

$$FG^{(\alpha,\gamma)} = \{(f_i^{(\alpha,\gamma)}, g_i^{(\alpha,\gamma)})\}_{i\in\{u,l\}}.$$

Each control system $(f_i^{(\alpha,\gamma)}, g_i^{(\alpha,\gamma)})$ is given by:

$$f_i^{(\alpha,\gamma)}(q_i,\dot{q}_i) + g_i^{(\alpha,\gamma)}(q_i,\dot{q}_i)v_i = f_i^{(\alpha,\gamma)}(q_i,\dot{q}_i) + g_i^{3\mathrm{D}}(q_i,\dot{q}_i)b_{n_i}v_i,$$

where $v_i \in \mathbb{R}$ and b_{n_i} is the n_i^{th} basis vector in $\mathbb{R}^{n_i}$ with $n_u = 4$ and $n_l = 3$ and $f_i^{(\alpha,\gamma)}$ as given in (5).

Motivated by our desire to satisfy (6), we define the following two functions for $i \in \{u, l\}$,

$$h_i(q_i,\dot{q}_i) := \dot{\varphi} + \frac{1}{m_i^{\varphi}(\theta_i)}(\alpha\varphi + M_i^{\varphi,\theta}(\theta_i)\dot{\theta}_i).$$

The main idea in the construction of the third control law is that we would like to drive $h_i(q_i, \dot{q}_i)$ to zero, i.e., we would like to drive the system to the surface

$$\mathcal{Z}_i = \left\{ \begin{pmatrix} q_i \\ \dot{q}_i \end{pmatrix} \in TQ_i^{3\mathrm{D}} : h_i(q_i,\dot{q}_i) = 0 \right\}.$$

With this in mind, and motivated by the standard method for driving an output function to zero in a nonlinear control system, we define the following feedback control laws:

$$v_i = KZ_i^{(\epsilon,\alpha,\gamma)}(q_i,\dot{q}_i) := \frac{-1}{L_{g_i^{(\alpha,\gamma)}}h_i(q_i,\dot{q}_i)}\left(L_{f_i^{(\alpha,\gamma)}}h_i(q_i,\dot{q}_i) + \frac{1}{\epsilon}h_i(q_i,\dot{q}_i)\right),$$

where $L_{g_i^{(\alpha,\gamma)}}h_i$ is the Lie derivative of h_i with respect to $g_i^{(\alpha,\gamma)}$, $L_{f_i^{(\alpha,\gamma)}}h_i$ is the Lie derivative of h_i with respect to $f_i^{(\alpha,\gamma)}$ and $KZ_i^{(\epsilon,\alpha,\gamma)}$ is well-defined since $L_{g_i^{(\alpha,\gamma)}}h_i(q_i,\dot{q}_i) \neq 0$. Note that under these control laws, each h_i will decay exponentially when the solution is in domain i since its evolution will be governed by the differential equation:

$$\dot{h}_i = -\frac{1}{\epsilon}h_i.$$

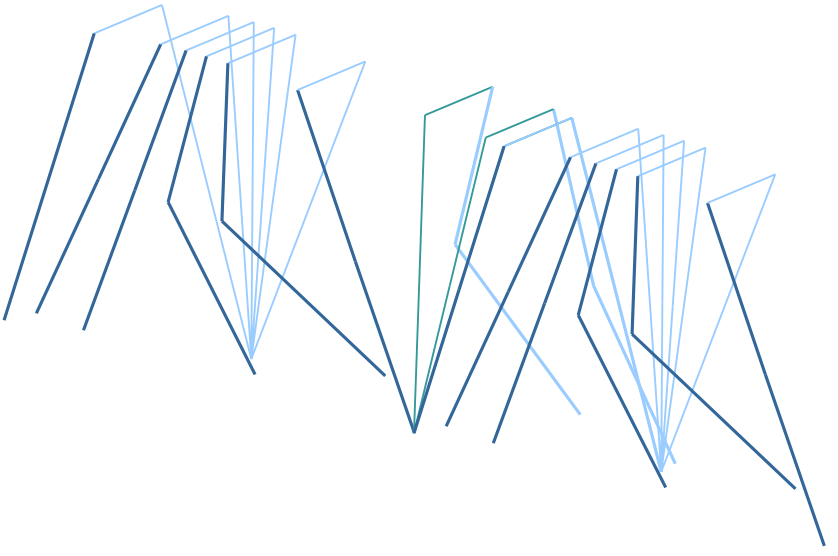


Fig. 4. A walking gait for the 3D biped

Utilizing the feedback control law $KZ_i^{(\epsilon,\alpha,\gamma)}$, we obtain a new hybrid system:

$$\mathscr{H}_{3D}^{(\epsilon,\alpha,\gamma)} := (\Gamma_{3D}, D_{3D}, G_{3D}, R_{3D}, F^{(\epsilon,\alpha,\gamma)}),$$

where $F^{(\epsilon,\alpha,\gamma)} = \{f_i^{(\epsilon,\alpha,\gamma)}\}_{i\in\{u,l\}}$ with

$$f_i^{(\epsilon,\alpha,\gamma)}(q_i,\dot{q}_i) := f_i^{(\alpha,\gamma)}(q_i,\dot{q}_i) + g_i^{(\alpha,\gamma)}(q_i,\dot{q}_i)KZ_i^{(\epsilon,\alpha,\gamma)}(q_i,\dot{q}_i).$$

Note that ϵ, α and γ can be thought of as control gains, as long as they are chosen so that $\epsilon > 0$, $\alpha > 0$, and γ such that $\mathscr{H}_{2D}^{\gamma}$ has a stable periodic orbit. We now proceed to examine the behavior of $\mathscr{H}_{3D}^{(\epsilon,\alpha,\gamma)}$.

4 Simulation Results

In this section we present simulation results supporting our claim that $\mathscr{H}_{3D}^{(\epsilon,\alpha,\gamma)}$ has a stable periodic orbit, i.e., that we obtain stable walking for the 3D biped.

We first choose model parameters $m_c = 0.05$kg, $m_t = 0.5$kg, $M_h = 0.5$kg, $\rho = 0.0188$rad, $w = 10$cm, $\ell = 1$m, $r_c = 0.372$m, $r_t = 0.175$m, $\gamma = 0.0504$rad, $\epsilon = \frac{1}{5}$, and $\alpha = 10$. The walking gait and stable limit cycle for our model with initial condition

$$x_0 = \big(0.000628 \quad 0.236309 \quad 0.236309 \quad -0.238929 \quad -0.238929 \\ 0.016716 \quad 1.513716 \quad 1.513716 \quad 1.590103 \quad 1.590103\big)^T$$

and these parameters is shown in Figure 4, 5 and 6, respectively. Note that each jump in the phase portraits shown corresponds to a jump from one vertex in

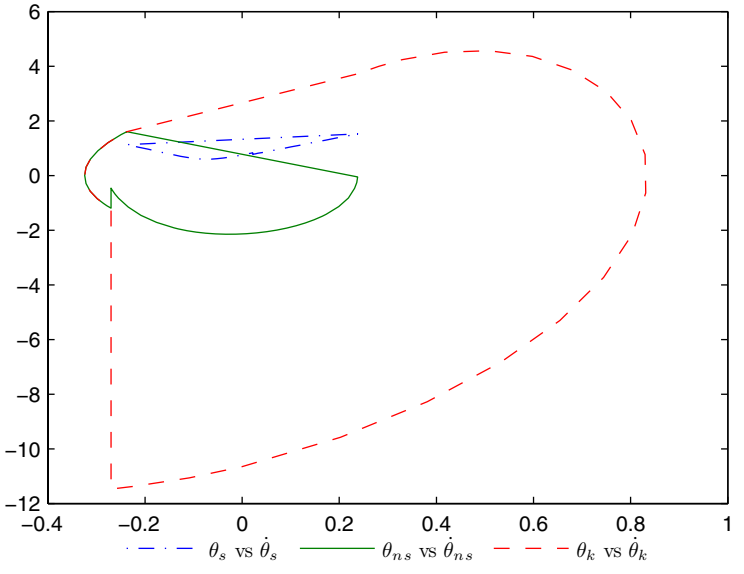


Fig. 5. A stable limit cycle of the 3D biped (top)

the graph Γ_{3D} to another and that, by inspection, the system appears to have a stable 2-periodic limit cycle.

We show that this limit cycle is (locally) exponentially stable by verifying that the eigenvalues of the linearized Poincare map at a fixed point of the limit cycle all have magnitude less than one [15]. Since our hybrid system consists of multiple domains we choose a fixed point right before footstrike, run the model forward two strides, and obtain five stable eigenvalues from the Jacobian of the Poincare map. The linearized Poincare map will always yield $n-1$ eigenvalues, where n is the dimension of the configuration space where the Poincare section of the Poincare map is located, since the Poincare section is by definition taken to be an $n-1$ dimensional hypersurface. Since our fixed point is in the knee-locked domain, our configuration space is Q_l^{3D}, of dimension 6. Thus, the 5 eigenvalues are $0.060149 \pm 0.593669\imath$, 0.000010, 0.004772 and 0.029407. The fact that these eigenvalues have magnitude much less than 1 suggests that the periodic orbit is both stable and that our third control law is effective at rejecting perturbations that might prevent the system from reaching a stable limit cycle.

The zero dynamics controller ensures that during the continuous evolution of the biped, solutions will converge exponentially to the surface $\mathcal{Z}_i$ where the sagittal and lateral dynamics are decoupled. What is interesting is that after each kneestrike or footstrike, the dynamics are thrown off the surface $\mathcal{Z}_i$ whereafter the zero dynamics controller again drives the system to the surface (this behavior can be seen in Figure 7). This could theoretically destroy the stability of walking in the sagittal plane, but fortunately does not due to two main facts: the perturbations away from the surface $\mathcal{Z}_i$ are not large, and the zero dynamics

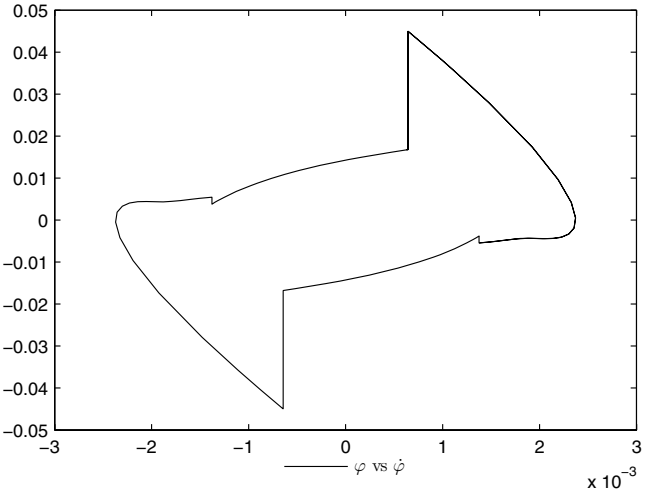


Fig. 6. A zoomed view of the lateral-plane (φ,$\dot{\varphi}$) limit cycle (bottom)

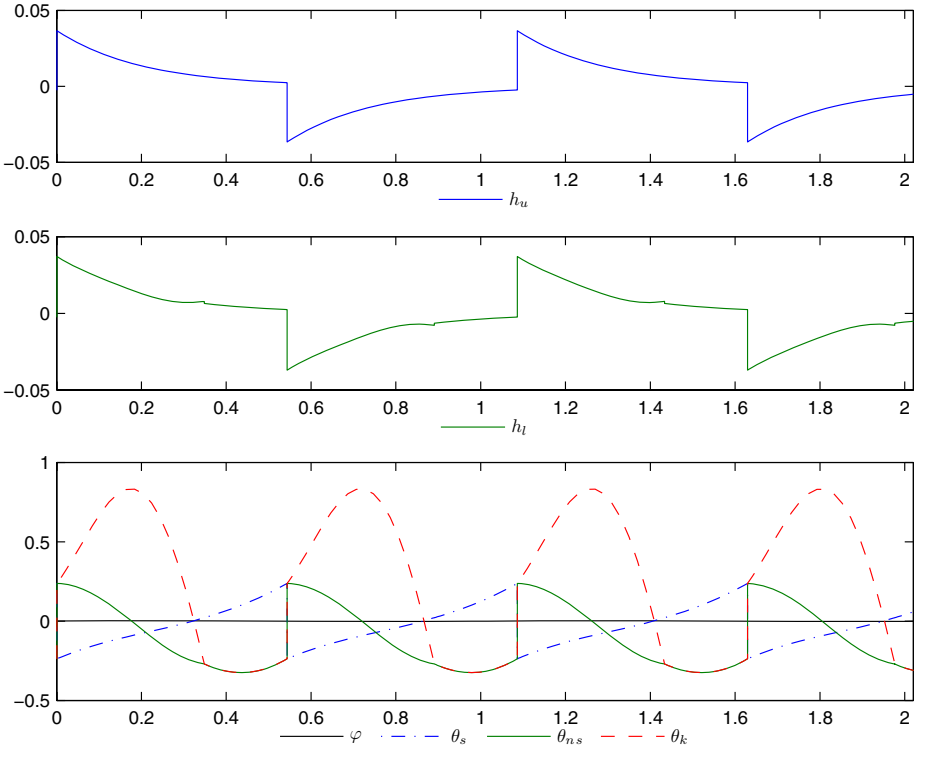


Fig. 7. The evolution of h_u and h_l for a walking gait of the 3D biped (top) and angles over time for the walking gait of the 3D biped (bottom)

brings the system close to the surface very quickly (exponentially fast) where again the decoupling effects are enjoyed. As a final note, the functions h_u and h_l will only decay exponentially when the solution is in domain u and l, respectively. This can be seen in the plot of h_l, where this function does not decay exponentially when the knee is unlocked, but does after kneestrike which occurs at the smaller jumps in the function.

The results of our simulation also indicate that we are able to obtain very natural walking using our three control laws. Looking at the time evolution of the knee angle θ_k in Figure 7, we see that in the knee-unlocked domain the leg swings naturally due to the passive dynamics, and then locks briefly before footstrike. It was shown in [2] that the natural side-to-side swaying, evident in the phase portrait of φ in Figure 6, is induced by the functional Routhian reduction used in the second control law. When the third control law brings the system close to the surface $\mathcal{Z}_i$, the phase portraits of the sagittal dynamics appear very similar to those of the 2D biped. As a result the stance and non-stance angles evolve like the 2D biped. In other words we have obtained stable, energy-efficient and natural walking gaits by virtue of the decoupling of the sagittal and lateral dynamics.

5 Conclusion

This paper presented a hybrid control law yielding stable walking for a three-dimensional biped with a hip and knees; while the result of this control law was natural-looking walking, indicating that it captures the natural dynamics of walking, there are numerous future research questions that result from this work. First, while the stability of the walking gait was verified numerically, the question is: can similar results be proven analytically? More importantly, in order to obtain these results, it was necessary to assume full actuation; since more complex walking involves phases of underactuation, dealing with underactuation in the context of the control scheme outlined here presents very interesting challenges. Finally, considering more complex bipedal robots is of fundamental importance, e.g., bipeds with feet. In considering these models, the corresponding hybrid systems will become increasingly complex with many more discrete domains and transitions between them. The final goal is to apply the general control strategy presented here to these more complex models in order to design bipedal walkers that display human looking walking.

Acknowledgments

The authors would like to thank Bobby Gregg and Mark Spong for their assistance in establishing results that lead to this paper. They would also like to thank Jessy Grizzle for many enlightening discussions on bipedal walking.

References

1. Ames, A.D., Gregg, R.D.: Stably extending two-dimensional bipedal walking to three dimensions. In: 26th American Control Conference, New York, NY (2007)
2. Ames, A.D., Gregg, R.D., Spong, M.W.: A geometric approach to three-dimensional hipped bipedal robotic walking. In: 45th Conference on Decision and Control, San Diago, CA (2007)
3. Ames, A.D., Gregg, R.D., Wendel, E.D.B., Sastry, S.: Towards the geometric reduction of controlled three-dimensional robotic bipedal walkers. In: 3rd Workshop on Lagrangian and Hamiltonian Methods for Nonlinear Control (LHMNLC 2006), Nagoya, Japan (2006)
4. Hsu Chen, V.F.: Passive dynamic walking with knees: A point foot model. Master's thesis, MIT (2007)
5. Collins, S.H., Wisse, M., Ruina, A.: A 3-d passive dynamic walking robot with two legs and knees. International Journal of Robotics Research 20, 607–615 (2001)
6. Goswami, A., Thuilot, B., Espiau, B.: Compass-like biped robot part I: Stability and bifurcation of passive gaits. Rapport de recherche de l'INRIA (1996)
7. Grizzle, J.W., Abba, G., Plestan, F.: Asymptotically stable walking for biped robots: Analysis via systems with impulse effects. IEEE Transactions on Automatic Control 46(1), 51–64 (2001)
8. Guobiao, S., Zefran, M.: Underactuated dynamic three-dimensional bipedal walking. In: Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006, pp. 854–859 (2006)
9. Kuo, A.D.: Stabilization of lateral motion in passive dynamic walking. International Journal of Robotics Research 18(9), 917–930 (1999)
10. Lygeros, J., Johansson, K.H., Simic, S., Zhang, J., Sastry, S.: Dynamical properties of hybrid automata. IEEE Transactions on Automatic Control 48, 2–17 (2003)
11. Marsden, J.E., Ratiu, T.S.: Introduction to Mechanics and Symmetry. Texts in Applied Mathematics, vol. 17. Springer, Heidelberg (1999)
12. McGeer, T.: Passive dynamic walking. International Journal of Robotics Research 9(2), 62–82 (1990)
13. McGeer, T.: Passive walking with knees. In: IEEE International Conference on Robotics and Automation, Cincinnati, OH (1990)
14. Morris, B., Grizzle, J.W.: A restricted Poincaré map for determining exponentially stable periodic orbits in systems with impulse effects: Application to bipedal robots. In: 44th IEEE Conference on Decision and Control and European Control Conference, Seville, Spain (2005)
15. Parker, T.S., Chua, L.O.: Practical numerical algorithms for chaotic systems. Springer, New York (1989)
16. Sastry, S.: Nonlinear Systems: Analysis, Stability and Control. Springer, Heidelberg (1999)
17. Spong, M.W., Bullo, F.: Controlled symmetries and passive walking. IEEE Transactions on Automatic Control 50(7), 1025–1031 (2005)
18. Westervelt, E.R., Grizzle, J.W., Chevallereau, C., Choi, J.H., Morris, B.: Feedback Control of Dynamic Bipedal Robot Locomotion. Taylor & Francis/CRC (2007)

Safe and Secure Networked Control Systems under Denial-of-Service Attacks

Saurabh Amin[1], Alvaro A. Cárdenas[2], and S. Shankar Sastry[2]

[1] Systems engineering, University of California, at Berkeley - Berkeley, CA, USA
amins@berkeley.edu
[2] EECS Department, University of California, at Berkeley - Berkeley, CA, USA
{cardenas,sastry}@eecs.berkeley.edu

Abstract. We consider the problem of security constrained optimal control for discrete-time, linear dynamical systems in which control and measurement packets are transmitted over a communication network. The packets may be jammed or compromised by a malicious adversary. For a class of denial-of-service (DoS) attack models, the goal is to find an (optimal) causal feedback controller that minimizes a given objective function subject to safety and power constraints. We present a semi-definite programming based solution for solving this problem. Our analysis also presents insights on the effect of attack models on solution of the optimal control problem.

1 Introduction

Attacks to computer networks have become prevalent over the last decade. While most control networks have been safe in the past, they are currently more vulnerable to malicious attacks [7,18]. The consequences of a successful attack on control networks can be more damaging than attacks on other networks because control systems are at the core of many critical infrastructures. Therefore, analyzing the security of control systems is a growing concern [4,7,12,13,15,18].

In the control and verification community there is a significant body of work on networked control [16], stochastic system verification [6,1], robust control [2, 11,3,10], and fault-tolerant control [21]. We argue that several major security concerns for control systems are not addressed by the current literature. For example, fault analysis of control systems usually assumes independent modes of failure, while during an attack, the modes of failure will be highly correlated. On the other hand, most networked control work assumes that the failure modes follow a given class of probability distributions; however, a real attacker has no incentives to follow this assumed distribution, and may attack in a non-deterministic manner. Finally, the work in stochastic system verification has addressed safety and reachability problems for fairly general systems; however, the potential applicability of these results for securing control systems has not been studied.

In this article, we formulate and analyze the problem of secure control for discrete-time linear dynamical systems. Our work is based on two ideas: (1) the

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 31–45, 2009.

introduction of safety-constraints as one of the top security requirements of a control system, and (2) the introduction of new adversary models—we generalize traditional uncertainty classes for control systems to incorporate more realistic attacks. The goal in our model is to minimize a performance function such that a safety specification is satisfied with high probability and power limitations are obeyed in expectation when the sensor and control packets can be dropped by a random or a resource-constrained attacker. Our analysis uses tools from optimal control theory such as dynamic and convex programming.

1.1 Attacks on Control Systems

Malicious cyber attacks to control systems can be classified as either *deception* attacks or *denial-of-service* DoS attacks.

In the context of control systems, integrity refers to the trustworthiness of sensor and control data packets. A lack of integrity results in deception: when a component receives false data and believes it to be true. In Figure 1, A1 and A3 represent deception attacks, where the adversary sends false information $\tilde{y} \neq y$ or $\tilde{u} \neq u$ from (one or more) sensors or controllers. The false information can include: an incorrect measurement, the incorrect time stamp, or the incorrect sender identity. The adversary can launch these attacks by compromising some sensors (A1) or controllers (A3).

On the other hand, availability of a control system refers to the ability of all components of being accessible. Lack of availability results in a DoS of sensor and control data. A2 and A4 represent *DoS attacks* in Figure 1, where the adversary prevents two entities from communicating. To launch a DoS the adversary can jam the communication channels, compromise devices and prevent them from sending data, attack the routing protocols, flood with network traffic some devices, etc.

Lastly, A5 represents a direct attack against the actuators or the plant. Solutions to these attacks, fall in the realm of detecting such attacks and improving the physical security of the system.

As shown by the analysis of a database that tracked cyber-incidents affecting industrial control systems from 1982 to 2003 [4], DoS is the most likely threat to control systems; therefore in this article we focus on DoS attacks, leaving deception attacks for future work.

2 Problem Setting

2.1 System Model

We consider a linear time invariant stochastic system over a time horizon $k = 0, \ldots, N-1$ with measurement and control packets subject to DoS attacks (γ_k, ν_k):

$$x_{k+1} = Ax_k + Bu_k^a + w_k \qquad k = 0, \ldots, N-1, \tag{1}$$

$$u_k^a = \nu_k u_k \qquad \nu_k \in \{0, 1\}, \tag{2}$$

$$x_k^a = \gamma_k x_k \qquad \gamma_k \in \{0, 1\}, \tag{3}$$

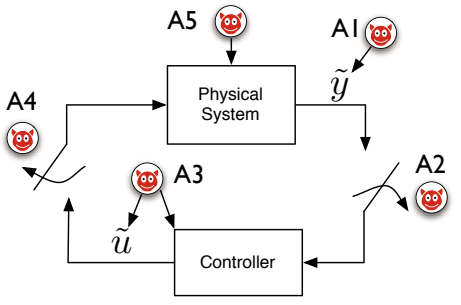


Fig. 1. Attacks on a control system: $A1$ and $A3$ indicate integrity attacks, $A2$ and $A4$ indicate DoS attacks, and $A5$ indicate direct physical attacks to the process

where $x_k \in \mathbb{R}^n$ and $u_k \in \mathbb{R}^m$ denote the state and the control input respectively, $w_k \in \mathbb{R}^n$ is independent, Gaussian distributed noise with mean 0 and covariance W (denoted as $w_k \sim \mathcal{N}(0, W)$), $x_0 \sim \mathcal{N}(\bar{x}, P_0)$ is the initial state, and $\{\gamma_k\}$ (resp. $\{\nu_k\}$) is the sensor (resp. actuator) attack sequence. Also, x_0 and w_k are uncorrelated. The available state (resp. available control input) is denoted by x_k^a (resp. u_k^a) after a DoS attack on the measurement (resp. control) packet. Following [16], for an acknowledgment based communication protocol such as TCP, the information set available at time k is $\mathcal{I}_k = \{x_0^a, \ldots, x_k^a, \gamma_0^k, \nu_0^{k-1}\}$ where $\gamma_i^j = (\gamma_i, \ldots, \gamma_j)$ and $\nu_i^j = (\nu_i, \ldots, \nu_j)$. Define $u_0^{N-1} = (u_0, \ldots, u_{N-1})$.

We note that due to (3), the controller receives perfect state information x_k when $\gamma_k = 1$ and 0 when $\gamma_k = 0$. However, our analysis presented can also be extended for the case of measurement equation $y_k^a = \gamma_k C_s x_k + v_k$.

2.2 Goals and Requirements

At this stage, we have not specified any restrictions on the DoS attack actions except that $(\gamma_k, \nu_k) \in \{0,1\}^2$ for $k = 0, \ldots, N-1$. We will impose constraints on the attacker actions in Section 3.1. Given such constraints, our goal is to synthesize a causal feedback control law $u_k = \mu_k(\mathcal{I}_k)$ such that for the system (1), (2), and (3), the following finite-horizon objective function is minimized

$$J_N(\bar{x}, P_0, u_0^{N-1}) = \mathbf{E}\left[x_N^\top Q^{xx} x_N + \sum_{k=0}^{N-1} \begin{pmatrix} x_k \\ u_k \end{pmatrix}^\top \begin{pmatrix} I_n & 0 \\ 0 & \nu_k I_m \end{pmatrix} Q \begin{pmatrix} x_k \\ u_k \end{pmatrix} \Big| u_0^{N-1}, \bar{x}, P_0\right] \tag{4}$$

where $Q^{xx} \succ 0$, and $Q \succeq 0$ is partitioned as

$$Q = \begin{pmatrix} Q^{xx} & 0 \\ 0 & Q^{uu} \end{pmatrix} \in \mathbb{R}^{(n+m)\times(n+m)},$$

and constraints on *both* the state and the input in an expected sense

$$\mathbf{E}\left[\begin{pmatrix} x_k \\ u_k \end{pmatrix}^\top \begin{pmatrix} I_n & 0 \\ 0 & \nu_k I_m \end{pmatrix} H_i \begin{pmatrix} x_k \\ u_k \end{pmatrix}\right] \leq \beta_i \quad \text{for } i = 1, \ldots, L, \text{ and } k = 0, \ldots, N-1 \tag{5}$$

with $H_i \succeq 0$ and scalar constraints on the state and the input in a probabilistic sense

$$\mathbf{P}\left[t_i^\top \begin{pmatrix} I_n & 0 \\ 0 & \nu_k I_m \end{pmatrix} \begin{pmatrix} x_k \\ u_k \end{pmatrix} \leq \alpha_i\right] \geq (1-\varepsilon) \quad \text{for } i = 1,\ldots,T, \text{ and } k = 0,\ldots,N-1 \quad (6)$$

with $t_i \in \mathbb{R}^{n+m}$ are satisfied. The constraints (5) can be viewed as *power constraints* that limit the energy of state and control inputs at each time step. The constraint (6) can be interpreted as a *safety specification* stipulating that the state and the input remain within the hyperplanes specified by t_i and α_i with a sufficiently high probability, $(1-\varepsilon)$, for $k = 0,\ldots,N-1$. Equations (5) and (6) are to be interpreted as conditioned on the initial state, i.e., $\mathbf{E}[\cdot] := \mathbf{E}[\cdot|x_0]$ and $\mathbf{P}[\cdot] := \mathbf{P}[\cdot|x_0]$.

3 Optimal Control with Constraints and Random Attacks

3.1 A Random DoS Attack Model

Networked control formulations have previously considered the loss of sensor or control packets and their impact on the system. While previous results model packet drops caused by random events (and not by an attacker) we believe these packet drop models can be used as a first-step towards understanding the impact of DoS attacks to our objective and constraints.

One of these models is the Bernoulli packet drop model, in which at each time, the attacker randomly jams a measurement (resp. control) packet according to independent Bernoulli trials with success probability $\bar{\gamma}$ (resp. $\bar{\nu}$). This attack model, referred as the $\mathrm{Ber}(\bar{\gamma},\bar{\nu})$ adversary, has the following admissible attack actions

$$\mathcal{A}_{\mathrm{Ber}(\bar{\gamma},\bar{\nu})} = \{(\gamma_0^{N-1},\nu_0^{N-1})|\mathbf{P}(\gamma_k=1)=\bar{\gamma},\mathbf{P}(\nu_k=1)=\bar{\nu},\ k=0,\ldots,N-1\}. \quad (7)$$

For the $\mathcal{A}_{\mathrm{Ber}(\bar{\gamma},\bar{\nu})}$ model, we can write the Kalman filter equations for the state estimate $\hat{x}_{k|k} := \mathbf{E}[x_k|\mathcal{I}_k]$ and the state estimation error $e_{k|k} := (x_k - \hat{x}_{k|k})$. For the update step we have

$$\hat{x}_{k+1|k} = A\hat{x}_{k|k} + \nu_k B u_k \text{ and, } e_{k+1|k} = Ae_{k|k} + w_k$$

and for the correction step

$$\hat{x}_{k+1|k+1} = \gamma_{k+1}x_{k+1} + (1-\gamma_{k+1})\hat{x}_{k+1|k} \text{ and, } e_{k+1|k+1} = (1-\gamma_{k+1})e_{k+1|k},$$

starting with $\hat{x}_{0|-1} = \bar{x}$ and $e_{0|-1} \sim \mathcal{N}(0,P_0)$. It follows that the error covariance matrices $\Sigma_{k+1|k} := \mathbf{E}[e_{k+1|k}e_{k+1|k}^\top|\mathcal{I}_k]$ and $\Sigma_{k|k} := \mathbf{E}[e_{k|k}e_{k|k}^\top|\mathcal{I}_k]$ do not depend on the control input u_k. Thus, the separation principle holds for TCP-like communication [16]. Furthermore, it is easy to see that

$$\mathbf{E}[e_{k|k}x_{k|k}^\top] = 0. \quad (8)$$

Taking expectations w.r.t. $\{\gamma_k\}$, the expected error covariances follow

$$\mathbf{E}_\gamma[\Sigma_{k+1|k}] = A\mathbf{E}_\gamma[\Sigma_{k|k}]A^\top + W \text{ and, } \mathbf{E}_\gamma[\Sigma_{k+1|k+1}] = (1-\bar{\gamma})\mathbf{E}_\gamma[\Sigma_{k+1|k}],$$

for $k = 0, \ldots, N-1$ starting with the initial condition $\Sigma_{0|-1} = P_0$. For the ease of notation, we denote $\hat{x}_{k+1} := \hat{x}_{k+1|k}$, $e_{k+1} := e_{k+1|k}$, and $\Sigma_{k+1} := \Sigma_{k+1|k}$. Using the Kalman filter equations we obtain for $k = 0, \ldots, N-1$

$$\hat{x}_{k+1} = A\hat{x}_k + \nu_k B u_k + \gamma_k A e_k \tag{9}$$

$$e_{k+1} = (1-\gamma_k)Ae_k + w_k \tag{10}$$

$$\mathbf{E}_\gamma[\Sigma_{k+1}] = (1-\bar{\gamma})A\mathbf{E}_\gamma[\Sigma_k]A^\top + W. \tag{11}$$

Definition 1. *For Bernoulli attacks, $(\gamma_0^{N-1}, \nu_0^{N-1}) \in \mathcal{A}_{Ber(\bar{\gamma},\bar{\nu})}$ over systems controlled over TCP-like communication protocols, the safety-constrained robust optimal control problem is equivalent to minimizing (4) subject to (9), (11), (5) and (6).*

3.2 Controller Parameterization

In this section, we deal with the safety-constrained optimal control problem as defined in Definition 1. Naive implementation of the control law $u_k^* = -L_k\hat{x}_{k|k}$ may not guarantee constraint satisfaction for any initial state. Recent research has shown that for the optimal control problems involving state and input constraints, more general causal feedback controllers can guarantee a larger set of initial states for which the constrained optimal control problem admits a feasible solution [3,10,17,14,19]. Specifically, these approaches consider the problem of designing causal controllers that are affine in all previous measurements such that a convex objective function is minimized subject to constraints imposed by the system dynamics, and the state and inputs constraints are satisfied.

When considering a system under DoS attacks, (1), (2), and (3), the class of causal feedback controllers can be defined as an affine function of the available measurements, i.e.,

$$u_k = \bar{u}_k + \sum_{j=0}^{k} \gamma_j M_{k,j} x_j, \qquad k = 0, \ldots, N-1 \tag{12}$$

where $\bar{u}_k \in \mathbb{R}^m$ is the open-loop part of the control, and $M_{k,j} \in \mathbb{R}^{m\times n}$ is the feedback gain or the recourse at time k from sensor measurement x_j. For a lost measurement packet, say $x_{j'}$ for $\gamma_{j'} = 0$, the corresponding feedback gain $M_{k,j'}$ has no contribution toward the control policy. We note that the above parameterization can be re-expressed as an affine function of innovations $v_{k|k-1} := \gamma_k(x_k - \hat{x}_{k|k-1}) = \gamma_k e_k$ as

$$u_k = u_k^\circ + \sum_{j=0}^{k} \gamma_j M_{k,j} e_j, \qquad k = 0, \ldots, N-1 \tag{13}$$

where $u_k^\circ := \bar{u}_k + \sum_{j=0}^{k} \gamma_j M_{i,j}\hat{x}_{j|j-1}$.

Remark 1. When only the current available measurement is used for computing the feedback policy, the mapping μ_k can be expressed as

$$u_k = \bar{u}_k + \gamma_k M_{k,k} x_k = u_k^\circ + \gamma_k M_k e_k, \qquad k = 0, \ldots, N-1, \tag{14}$$

where $M_k := M_{k,k}$ for ease of notation and $u_k^\circ := \bar{u}_k + \gamma_k M_k \hat{x}_{k|k-1}$. □

3.3 Convex Characterization

In this section, we will show that unlike (12), the use of control parameterization (13) yields an affine representation of state and control trajectories in terms of the control parameters $\bar{u}_k$ (or u_k°) and $M_{k,j}$. We use $\mathbf{x}$, $\hat{\mathbf{x}}$, $\mathbf{u}$, $\mathbf{e}$ and $\mathbf{w}$ to denote the respective trajectories over the time horizon $0, \ldots, N$. That is, $\mathbf{x} = (x_0^\top, \ldots, x_N^\top)^\top \in \mathbb{R}^{n(N+1)}$ and similarly for $\hat{\mathbf{x}} \in \mathbb{R}^{n(N+1)}$ and $\mathbf{e} \in \mathbb{R}^{n(N+1)}$; $\mathbf{u} = (u_0^\top, \ldots, u_{N-1}^\top)^\top \in \mathbb{R}^{mN}$ and similarly for $\mathbf{w} \in \mathbb{R}^{nN}$. Using this representation, the system (11) and the control parameterization (12) can be written as

$$\mathbf{x} = \mathbf{A}\mathbf{w} + \mathbf{B}\mathbf{N}\mathbf{u} + \mathbf{x_0}, \tag{15}$$

$$\mathbf{u} = \bar{\mathbf{u}} + \mathbf{M}\boldsymbol{\Gamma}\mathbf{x}, \tag{16}$$

where $\mathbf{x_0}$, $\mathbf{A}$, $\mathbf{B}$, $\boldsymbol{\Gamma}$, $\mathbf{N}$ are given in the Appendix and

$$\mathbf{M} = \begin{pmatrix} M_{0,0} & 0 & \ldots & 0 \\ M_{1,0} & M_{1,1} & \ldots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ M_{N-1,0} & \ldots & M_{N-1,N-1} & 0 \end{pmatrix} \in \mathbb{R}^{mN \times n(N+1)}, \quad \bar{\mathbf{u}} = \begin{pmatrix} \bar{u}_0 \\ \vdots \\ \bar{u}_{N-1} \end{pmatrix} \in \mathbb{R}^{mN} \tag{17}$$

Using (15) and (16), we can show that the closed-loop system response can be written as

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{u} \end{pmatrix} = \begin{pmatrix} \tilde{\mathbf{G}}_{\mathbf{xw}} \\ \tilde{\mathbf{G}}_{\mathbf{uw}} \end{pmatrix} \mathbf{w} + \begin{pmatrix} \tilde{\mathbf{x}} \\ \tilde{\mathbf{u}} \end{pmatrix} \tag{18}$$

where

$$\begin{aligned} \tilde{\mathbf{G}}_{\mathbf{xw}} &= \left(\mathbf{A} + \mathbf{BNM\Gamma}(I - \mathbf{BNM\Gamma})^{-1}\mathbf{A}\right) \\ \tilde{\mathbf{G}}_{\mathbf{uw}} &= \left(\mathbf{M\Gamma}(I - \mathbf{BNM\Gamma})^{-1}\mathbf{A}\right) \\ \tilde{\mathbf{x}} &= \mathbf{x_0} + \mathbf{BN}\bar{\mathbf{u}} + \mathbf{BNM\Gamma}(I - \mathbf{BNM\Gamma})^{-1}(\mathbf{x_0} + \mathbf{BN}\bar{\mathbf{u}}) \\ \tilde{\mathbf{u}} &= \mathbf{M\Gamma}(I - \mathbf{BNM\Gamma})^{-1}(\mathbf{x_0} + \mathbf{BN}\bar{\mathbf{u}}) + \bar{\mathbf{u}} \end{aligned}$$

Equation (18) is nonlinear in the control parameters $(\bar{\mathbf{u}}, \mathbf{M})$ and hence, parameterization (12) cannot be directly used for solving constrained stochastic optimal control problems. On the other hand, using (10), the error trajectory can be written as

$$\mathbf{e} = \mathbf{e}_0 + \mathbf{H}\mathbf{w} \tag{19}$$

where $\mathbf{e_0}$ and $\mathbf{H}$ are also given in the Appendix. Using (19), (15) and the control parameterization (13) we can re-express the closed-loop system response as

$$\begin{pmatrix}\mathbf{x}\\ \mathbf{u}\end{pmatrix} = \begin{pmatrix}\hat{\mathbf{G}}_{\mathbf{xw}}\\ \hat{\mathbf{G}}_{\mathbf{uw}}\end{pmatrix}\mathbf{w} + \begin{pmatrix}\hat{\mathbf{x}}\\ \hat{\mathbf{u}}\end{pmatrix} \tag{20}$$

where

$$\hat{\mathbf{G}}_{\mathbf{xw}} = (\mathbf{A}+\mathbf{BNM\Gamma H}), \qquad \hat{\mathbf{G}}_{\mathbf{uw}} = \mathbf{M\Gamma H}$$
$$\hat{\mathbf{x}} = \mathbf{BNM\Gamma e_0} + \mathbf{x_0} + \mathbf{BNu}^\circ, \qquad \hat{\mathbf{u}} = \mathbf{M\Gamma e_0} + \mathbf{u}^\circ$$

Thus, we arrive at the following result

Theorem 1. *Under the error feedback parameterization (13), the closed loop system response (20) is affine in the control parameters* $(\mathbf{u}^\circ, \mathbf{M})$. □

We will now use the error feedback parameterization (13) for our analysis. Alternatively, we also note the following result:

Remark 2. Using the transformation

$$\mathbf{Q} := \mathbf{M\Gamma}(I - \mathbf{BNM\Gamma})^{-1}, \qquad \mathbf{r} := (I + \mathbf{QBN})\bar{\mathbf{u}} \tag{21}$$

where $\mathbf{Q} \in \mathbb{R}^{mN\times n(N+1)}$ and $\mathbf{r} \in \mathbb{R}^{mn}$, the terms in equation (18) can be written as: $\mathbf{G_{xw}} = (I + \mathbf{BNQ})\,\mathbf{A}$, $\mathbf{G_{uw}} = \mathbf{QA}$, $\tilde{\mathbf{x}} = (I + \mathbf{BNQ})\,\bar{\mathbf{x}} + \mathbf{BNr}$, and $\tilde{\mathbf{u}} = \mathbf{Q}\bar{\mathbf{x}} + \mathbf{r}$. Using simple matrix operations, the relations in (21) can be inverted as $\mathbf{M\Gamma} = (I + \mathbf{QBN})^{-1}\mathbf{Q}$ and $\bar{\mathbf{u}} = (I - \mathbf{M\Gamma HN})\mathbf{r}$. Thus, under parameterization (21), the closed-loop system response also becomes affine in the control parameters $(\mathbf{r}, \mathbf{Q})$. □

3.4 Safety-Constrained Optimal Control for Bernoulli Attacks

For the control parameterization (12), and for the Bernoulli attack model, $\mathcal{A}_{\mathrm{Ber}(\bar{\gamma},\bar{\nu})}$ we will now solve the safety-constrained optimal control problem as stated in Lemma 1, i.e., minimize (4) subject to (9), (11), (5), and (6). We state the following useful lemma

Lemma 1 (Schur Complements). *For all $X \in \mathbb{S}^n$, $Y \in \mathbb{R}^{m\times n}$, $Z \in \mathbb{S}^m$, the following statements are equivalent:*

$$\text{a)} Z \succ 0, X - Y^\top Z^{-1} Y \succeq 0,$$
$$\text{b)} Z \succ 0, \begin{pmatrix} X & Y^\top \\ Y & Z \end{pmatrix} \succeq 0$$

For the sake of simplicity we will consider the parameterization (14). However, our results can be re-derived for the parameterization (12). First, we will derive the expression for

$$V_k = \mathbf{E}\left[\begin{pmatrix}\hat{x}_k\\ u_k^\circ\end{pmatrix}\begin{pmatrix}\hat{x}_k\\ u_k^\circ\end{pmatrix}^\top\right]$$

Using (14), the update equation for the state estimate (9) becomes

$$\hat{x}_{k+1} = A\hat{x}_k + \nu_k B u_k^\circ + \gamma_k (A + \nu_k B M_k) e_k, \tag{22}$$

and further defining $F = [I_n, 0] \in \mathbb{R}^{n \times (n+m)}$ we have,

$$\begin{aligned}
&FV_{k+1}F^\top = V_{k+1}^{\hat{x}\hat{x}} = \mathbf{E}\left[\hat{x}_{k+1}\hat{x}_{k+1}^\top\right] \\
&= \mathbf{E}\left[(A\hat{x}_k + \nu_k B u_k^\circ + \gamma_k(A + \nu_k B M_k)e_k)(A\hat{x}_k + \nu_k B u_k^\circ + \gamma_k(A + \nu_k B M_k)e_k)^\top\right] \\
&= \left[A \,|\sqrt{\bar{\nu}}B\right] \mathbf{E}\left[\begin{pmatrix}\hat{x}_k \\ u_k^\circ\end{pmatrix}\begin{pmatrix}\hat{x}_k \\ u_k^\circ\end{pmatrix}^\top\right]\left[A \,|\sqrt{\bar{\nu}}B\right]^\top \\
&+ \sqrt{\bar{\gamma}}(A + \sqrt{\bar{\nu}}BM_k)\mathbf{E}_\gamma[\Sigma_k](A + \sqrt{\bar{\nu}}BM_k)^\top\sqrt{\bar{\gamma}} \\
&= \left[AV_k \,|\sqrt{\bar{\nu}}BV_k\right](V_k)^{-1}\left[AV_k \,|\sqrt{\bar{\nu}}BV_k\right]^\top \\
&+ \sqrt{\bar{\gamma}}(A\mathbf{E}_\gamma[\Sigma_k] + \sqrt{\bar{\nu}}BU_k)(\mathbf{E}_\gamma[\Sigma_k])^{-1}(A\mathbf{E}_\gamma[\Sigma_k] + \sqrt{\bar{\nu}}BU_k)^\top\sqrt{\bar{\gamma}}
\end{aligned}$$

where we have used $U_k = M_k \mathbf{E}_\gamma[\Sigma_k]$. An upper bound on V can be obtained in the form of the following LMI by replacing the equality by $\succeq$ and using Schur complements for $k = 0, \ldots, N-1$:

$$\begin{bmatrix} (FV_{k+1}F^\top) & * & * & * \\ \left[AV_k \;\sqrt{\bar{\nu}}BV_k\right]^\top & 0 & V_k & * \\ \sqrt{\bar{\gamma}}(A\mathbf{E}_\gamma[\Sigma_k] + \sqrt{\bar{\nu}}BU_k)^\top & 0 & 0 & \mathbf{E}_\gamma[\Sigma_k] \end{bmatrix} \succeq 0 \tag{23}$$

The objective function (4) can be expressed as

$$\begin{aligned}
&\mathbf{E}\left[\mathbf{Tr}\left\{Q^{xx}x_N x_N^\top\right\}\right] + \sum_{k=0}^{N-1}\mathbf{E}\left[\mathbf{Tr}\left\{\begin{pmatrix}Q^{xx} & 0 \\ 0 & \nu_k Q^{uu}\end{pmatrix}\right\}\begin{pmatrix}x_k \\ u_k\end{pmatrix}\begin{pmatrix}x_k \\ u_k\end{pmatrix}^\top\right] \\
&= \mathbf{Tr}\left\{Q^{xx}\mathbf{E}\left[x_N x_N^\top\right]\right\} + \sum_{k=0}^{N-1}\mathbf{Tr}\left\{\begin{pmatrix}Q^{xx} & 0 \\ 0 & \mathbf{E}[\nu_k]Q^{uu}\end{pmatrix}\mathbf{E}\left[\begin{pmatrix}x_k \\ u_k\end{pmatrix}\begin{pmatrix}x_k \\ u_k\end{pmatrix}^\top\right]\right\} \\
&= \mathbf{Tr}\left\{Q^{xx}\mathbf{E}\left[\hat{x}_N\hat{x}_N^\top\right]\right\} + \sum_{k=0}^{N-1}\mathbf{Tr}\left\{\begin{pmatrix}Q^{xx} & 0 \\ 0 & \bar{\nu}Q^{uu}\end{pmatrix}\mathbf{E}\left[\begin{pmatrix}\hat{x}_k \\ u_k\end{pmatrix}\begin{pmatrix}\hat{x}_k \\ u_k\end{pmatrix}^\top\right]\right\} \\
&+ \sum_{k=0}^{N}\mathbf{Tr}\left\{Q^{xx}\mathbf{E}_\gamma[\Sigma_k]\right\}
\end{aligned}$$

Since Σ_k does not depend on the control input (refer to eq. (11)), $\sum_{k=0}^{N}\mathbf{Tr}\left\{Q^{xx}\mathbf{E}_\gamma[\Sigma_k]\right\}$ is a constant and minimizing $J_N(\bar{x}, P_0, u_0^{N-1})$ is the same as minimizing

$$\mathbf{Tr}\left\{Q^{xx}V_N^{\hat{x}\hat{x}}\right\} + \sum_{k=0}^{N-1}\mathbf{Tr}\left\{\begin{pmatrix}Q^{xx} & 0 \\ 0 & \bar{\nu}Q^{uu}\end{pmatrix}P_k\right\} \tag{24}$$

where $V_N^{\hat{x}\hat{x}}$ is equal to $\mathbf{E}\left[\hat{x}_N\hat{x}_N^\top\right]$ and the upper bound P_k is defined as

$$P_k \succeq \mathbf{E}\left[\begin{pmatrix}\hat{x}_k\\ u_k\end{pmatrix}\begin{pmatrix}\hat{x}_k\\ u_k\end{pmatrix}^\top\right] = \mathbf{E}\left[\begin{pmatrix}\hat{x}_k\\ u_k^\circ+\gamma_k M_k e_k\end{pmatrix}\begin{pmatrix}\hat{x}_k\\ u_k^\circ+\gamma_k M_k e_k\end{pmatrix}^\top\right]$$
$$= \mathbf{E}\left[\begin{pmatrix}\hat{x}_k\\ u_k^\circ\end{pmatrix}\begin{pmatrix}\hat{x}_k\\ u_k^\circ\end{pmatrix}^\top\right] + \begin{bmatrix}0 & 0\\ 0 & \bar{\gamma}U_k(\mathbf{E}_\gamma[\Sigma_k])^{-1}U_k^\top\end{bmatrix}$$

Again using Schur complement, we obtain for $k = 0, \ldots, N-1$

$$\begin{bmatrix} P_k & * & *\\ V_k & V_k & *\\ \begin{bmatrix}0\\ \sqrt{\bar{\gamma}}U_k\end{bmatrix}^\top & 0 & \mathbf{E}_\gamma[\Sigma_k]\end{bmatrix} \succeq 0 \tag{25}$$

The power constraints (5) can be written as

$$\mathbf{Tr}\left\{H_i\begin{bmatrix}I_n & 0\\ 0 & \mathbf{E}[\nu_k]I_m\end{bmatrix}\mathbf{E}\left[\begin{pmatrix}x_k\\ u_k\end{pmatrix}\begin{pmatrix}x_k\\ u_k\end{pmatrix}^\top\right]\right\}$$
$$= \mathbf{Tr}\left\{H_i\begin{bmatrix}I_n & 0\\ 0 & \bar{\nu}I_m\end{bmatrix}\mathbf{E}\left[\begin{pmatrix}\hat{x}_k\\ u_k\end{pmatrix}\begin{pmatrix}\hat{x}_k\\ u_k\end{pmatrix}^\top\right]\right\} + \mathbf{Tr}\left\{H_i^{xx}\mathbf{E}_\gamma[\Sigma_k]\right\}$$

Therefore the power constraints (5) become for $i = 1, \ldots, L, k = 0, \ldots, N-1$

$$\mathbf{Tr}\left\{H_i\begin{bmatrix}I_n & 0\\ 0 & \bar{\nu}I_m\end{bmatrix}P_k\right\} \leq \beta_i - \mathbf{Tr}\left\{H_i^{xx}\mathbf{E}_\gamma[\Sigma_k]\right\}. \tag{26}$$

Thus, we can now state the following theorem

Theorem 2. *For the $(\gamma_0^{N-1}, \nu_0^{N-1}) \in \mathcal{A}_{Ber(\bar{\gamma},\bar{\nu})}$ attack model the optimal causal controller of the form (14) for the system (1), (2), (3) that minimizes the objective function (4) subject to power constraints (5) is equivalent to solving the following semidefinite program (SDP):*

$$\mathcal{P}(\bar{x}, P_0, N) : \begin{cases}\min_{V_i,P_i,U_i}\ (24)\\ \textit{subject to } (23), (25), (26).\end{cases} \tag{27}$$

□

In order to handle the safety specification (6), we refer to Theorem 3.1 in [5] which says that for any $\epsilon \in (0,1)$, the chance constraint of the form

$$\inf_{d\sim\mathcal{D}} \mathbf{P}\left[d^\top\tilde{x} \leq 0\right] \geq 1-\epsilon$$

is equivalent to the second order cone constraint (SOCP)

$$\sqrt{\frac{1-\epsilon}{\epsilon}\tilde{x}^\top\Gamma\tilde{x}} + \hat{d}^\top\tilde{x} \leq 0$$

where $\mathcal{D}$ is the set of all probability distributions with mean $\hat{d}$ and covariance Γ, d is the uncertain data with distributions in the set of distributions $\mathcal{D}$, and $\tilde{x}$ is the decision variable. We claim without proof that safety specifications of type (6) can be converted to SOCP constraints following [5], [19].

4 Modeling General DoS Attacks

From the security viewpoint, it might be difficult to justify the incentive for the attacker to follow a $\mathcal{A}_{\text{Ber}(\bar{\gamma},\bar{\nu})}$ model. Therefore, in this section we introduce more general attack models that impose constraints on the DoS attack actions (γ_k, ν_k).

First, note that if we know in advance the strategy of the attacker—for any arbitrary sequence $(\gamma_0^{N-1}, \nu_0^{N-1})$—we can use the results from the previous theorem.

Corollary 1. *The results of Theorem 2 be specialized to any given attack signature* $(\gamma_0^{N-1}, \nu_0^{N-1}) \in \{0,1\}^{2N}$. □

However, in practice we do not know the strategy of the attacker, thus we need to prepare for all possible attacks. Our model constrains the attacker action in time by restricting the DoS attacks on the measurement (resp. control) packet for *at most* $p < N$ (resp. $q < N$) time steps anywhere in the time interval $i = 0, \ldots, N-1$. This attack model is motivated by limitations on the resources of the adversary—such as its battery power, or the response time of the defenders—which in turn limits the number of times it can block a transmission. We refer this attack model as the (p, q) adversary and it has the following admissible attack actions

$$\mathcal{A}_{pq} = \{(\gamma_0^{N-1}, \nu_0^{N-1}) \in \{0,1\}^{2N} \big| \parallel \gamma_0^{N-1} \parallel_1 \geq N-p, \parallel \nu_0^{N-1} \parallel_1 \geq N-q\}, \tag{28}$$

where $\|\cdot\|_1$ denotes the 1−norm. The size of $\mathcal{A}_{pq}$ is $\sum_{i=0}^{p} \binom{N}{N-i} \cdot \sum_{j=0}^{q} \binom{N}{N-j}$.

An interesting sub-class of $\mathcal{A}_{pq}$ attack actions is the class of block attack strategies

$$\mathcal{A}_{pq}^{\tau_x \tau_u} = \{(\gamma_0^{N-1}, \nu_0^{N-1}) \in \{0,1\}^{2N} | \gamma_{\tau_x}^{\tau_x+p-1} = 0, \nu_{\tau_u}^{\tau_u+q-1} = 0\} \tag{29}$$

where $\tau_x \in \{0, \ldots, N-p\}$ and $\tau_u \in \{0, \ldots, N-q\}$ are the times at which the attacker starts jamming the measurement and control packets respectively. The size of $\mathcal{A}_{pq}^{\tau_x \tau_u}$ is $(N-p+1) \cdot (N-q+1)$. The intuition behind this attack sub-class is that an attacker will consume all of its resources continuously in order to maximize the damage done to the system. In this attack sub-class, p and q can represent the response time of defensive mechanisms. For example, a packet-flooding attack may be useful until network administrators implement filters or replicate the node under attack; similarly a jamming attack may be useful only until the control operators find the jamming source and neutralize it.

We note that $\mathcal{A}_{pq}$ and $\mathcal{A}_{pq}^{\tau_x \tau_u}$ are *non-deterministic attack models* in that the attacker can choose its action non-deterministically as long as the constraints defined by the attack model are satisfied.

4.1 DoS Attacks against the Safety Constraint

One possible objective of the attacker can be to violate safety constraints:

Definition 2. *[Most unsafe attack] For a given attack model $\mathcal{A}$ and control strategy $\mu_k(\mathcal{I}_k)$, the best attack plan to violate safety specification that a output vector $z_k := (Cx_k + \nu_k D u_k)$ remains within safe set $\mathcal{S}$ is*

$$\max_{\mathcal{A}} \mathbf{P}[(Cx_k + \nu_k D\mu(I_k)) \in \mathcal{S}^c] \text{ for } k = 0, \ldots, N-1 \tag{30}$$

where S^c denotes the unsafe set.

We will now show that for control parameterization (12), the block pq attacks, $\mathcal{A}_{pq}^{\tau_x \tau_u}$ can be viewed as the best attack plan for violating the safety constraint (refer to Definition 2). We can write the system equation (1) as

$$x_{k+1} = Ax_k + \nu_k B\bar{u}_k + \nu_k \sum_{j=0}^{k} \gamma_j M_{k,j} x_j + w_k$$

and for the attack strategy $\mathcal{A}_{pq}^{\tau_x \tau_u}$:

$$x_{k+1} = \begin{cases} Ax_k + w_k \text{ for } k = \tau_u, \ldots, \tau_u + q - 1 \\ Ax_k + B\bar{u}_k + B\sum_{j=0}^{\min(\tau_x - 1, k)} M_{k,j} x_j \\ +\mathbf{1}(k \geq \tau_x + p) B \sum_{j=0}^{k} M_{k,j} x_j + w_k \text{ for } k = \begin{cases} 0, \ldots, \tau_u - 1 \\ \tau_u + q, \ldots, N-1. \end{cases} \end{cases} \tag{31}$$

Now, if we ignore $\bar{u}_k$ and substitute $\tau_x = 0$, $\tau_u = p$ in (31) we obtain

$$x_{k+1} = \begin{cases} Ax_k + w_k \text{ for } k = 0, \ldots, p + q - 1 \\ Ax_k + B\sum_{j=p}^{k} M_{k,j} x_j \text{ for } k = p + q, \ldots, N-1 \end{cases} \tag{32}$$

Thus, using the attack strategy $\mathcal{A}_{pq}^{0p}$, the first $p+q-1$ time steps evolve as open-loop and beyond time step $p+q$, the system evolves as closed using available measurements since time p. With this strategy output vector z_k is expected to violate the safety constraint in the shortest time.

5 Formulation of New Challenges

From the controller's viewpoint, it is of interest to design control laws that are robust against all attacker actions, i.e.:

Definition 3. *[Minimax (robust) control] For a given attack model $\mathcal{A}$, the security constrained robust optimal control problem is to synthesize a control law that minimizes the maximum cost over all $(\gamma_0^{N-1}, \nu_0^{N-1}) \in \mathcal{A}$, subject to the power and safety constraints. This can be written as the minimax problem*

$$\min_{\mu_k(\mathcal{I}_k)} \max_{\mathcal{A}} [(4) \text{ subject to } (1), (2), (3), (5) \text{ and, } (6)]. \tag{33}$$

In general, we note that the problem (33) may not always be feasible. When $\mathcal{A}$ is probabilistic, Definition 3 can be treated in sense of expectation or almost-surely.

On the other hand, from the attacker's viewpoint, it is of interest to determine the optimal *attack plan* that degrades performance, i.e.,

Definition 4. *[Maximin (worst-case) attack] For a given attack model $\mathcal{A}$, the optimal attack plan is the attacker action that maximizes the minimum operating costs. This can be written as the maximin problem*

$$\max_{\mathcal{A}} \min_{\mu_k(\mathcal{I}_k)} [(4) \text{ subject to } (1), (2), (3)]. \tag{34}$$

As a first effort to analyze these goals we first consider the classical linear quadratic control problem, and analyze the cost function for the case of (1) no attacks, (2) $\mathcal{A}_{\text{Ber}(\bar{\gamma},\bar{\nu})}$attacks, and (3) $\mathcal{A}_{pq}$ attacks.

The problem is to find the optimal control policy $u_k = \mu_k(\mathcal{I}_k)$ that minimizes the objective (4) for the system (1), (2), and (3). The solution of this problem can be obtained in closed form using dynamic programming (DP) recursions [9,16].

We recall that for the case of no-attack, i.e., $(\gamma_k, \nu_k) = (1, 1)$ for all k, the optimal control law is given by $u_k^* = -L_k x_k$ where $L_k := (B^\top S_{k+1} B + Q^{uu})^{-1} B^\top S_{k+1} A$ and the matrices S_k are chosen such that $S_N = Q^{xx}$ and for $k = N-1, \ldots, 0$,

$$S_k = A^\top S_{k+1} + Q^{xx} - R_k$$

with $R_k = L_k^\top (B^\top S_{k+1} B + Q^{uu}) L_k$. The optimal cost is given by

$$J_N^* = \bar{x}^\top S_0 \bar{x} + \mathbf{Tr}\{S_0 P_0\} + \sum_{k=0}^{N-1} \mathbf{Tr}\{S_{k+1} W\}. \tag{35}$$

Following [16], the optimal control law for the case of $\mathcal{A}_{\text{Ber}(\bar{\gamma},\bar{\nu})}$ attack model is given by $u_k^* = -L_k \hat{x}_{k|k}$ where $\hat{x}_{k|k}$ is given by the Kalman filter equations; the expressions for L_k, R_k, S_N are same as those for the no-attack case, and for $k = N-1, \ldots, 0$,

$$S_k = A^\top S_{k+1} A + Q^{xx} - \bar{\nu} R_k.$$

The optimal cost in this case is given by

$$J^*_{N,\mathcal{A}_{\text{Ber}(\bar{\gamma},\bar{\nu})}} = \bar{x}^\top S_0 \bar{x} + \mathbf{Tr}\{S_0 P_0\} + \sum_{k=0}^{N-1} \mathbf{Tr}\{S_{k+1} W\} + \sum_{k=0}^{N-1} \mathbf{Tr}\{\bar{\nu} R_k \mathbf{E}_\gamma[\Sigma_{k|k}]\} \tag{36}$$

Lemma 2. *$J^*_{N,\mathcal{A}_{Ber(\bar{\gamma},\bar{\nu})}} \geq J^*_N$ for all $(\bar{\gamma}, \bar{\nu}) \in [0, 1]$.* □

We now consider the case of $\mathcal{A}_{pq}$ attacks. We can solve the problem of optimal attack plan for the $\mathcal{A}_{pq}$ attack class (refer to Definition 4):

For any *given* attack signature, $(\gamma_0^{N-1}, \nu_0^{N-1}) \in \{0, 1\}^{2N}$, the update equations of error covariance are $\Sigma_{k+1|k} = A \Sigma_{k|k} A^\top + W$ and $\Sigma_{k+1|k+1} = (1 - \gamma_{k+1}) \Sigma_{k+1|k}$ and the optimal cost is given by

$$J_{N,\mathcal{A}_{pq}} = \bar{x}^\top S_0 \bar{x} + \mathbf{Tr}\{S_0 P_0\} + \sum_{k=0}^{N-1} \mathbf{Tr}\{S_{k+1} Q\} + \sum_{k=0}^{N-1} \mathbf{Tr}\{(A^\top S_{k+1} A + Q^{xx} - S_k)\Sigma_{k|k}\} \quad (37)$$

where $S_N = Q^{xx}$ and for $k = N-1, \ldots, 0$,

$$S_k = A^\top S_{k+1} A + Q^{xx} - \nu_k A^\top S_{k+1} B (B^\top S_{k+1} B + Q^{uu})^{-1} B^\top S_{k+1} A. \quad (38)$$

and for $k = 1, \ldots, N-1$,

$$\Sigma_{k|k} = \prod_{j=1}^{k} (1-\gamma_j) A^k P_0 {A^k}^\top + \sum_{i=0}^{k-1} \prod_{j=(k-i)}^{k} (1-\gamma_j) A^i W {A^i}^\top. \quad (39)$$

Proposition 1 *An optimal attack plan for $\mathcal{A}_{pq}$ attack model is a solution of the following optimization problem:*

$$\max_{\mathcal{A}_{pq}} (37) \text{ subject to } (38), (39),$$
$$\| \gamma_0^{N-1} \|_1 \geq (N-p), \text{ and } \| \nu_0^{N-1} \|_1 \geq (N-q).$$

We note that while $\Sigma_{k|k}$ is affected by the *past* measurement attack sequence $\{\gamma_0^k\}$, S_k is affected by the *future* control attack sequence $\{\nu_k^{N-1}\}$.

Remark 3. We can use dynamic programming or convex duality theory to solve the problem without the ℓ_1 constraints on γ_0^{N-1} and ν_0^{N-1}, see [9]. In this case, it is well-known that the optimal control policy is given by the linear feedback law that depends only on the current state. To solve the constrained problem as posed in Proposition 1, we propose to use suitable convex relaxations for the ℓ_1 constraints and solve the relaxed problem using semidefinite programming. □

In future work we intend to address these problems and extend our results to deception attacks.

Acknowledgments. We thank Laurent El Ghaoui for his help in the initial part of the project. We also thank Manfred Morari and Bruno Sinopoli for helpful discussions.

References

1. Amin, S., Abate, A., Prandini, M., Lygeros, J., Sastry, S.: Reachability Analysis for Controlled Discrete Time Stochastic Hybrid Systems. In: Hespanha, J.P., Tiwari, A. (eds.) HSCC 2006. LNCS, vol. 3927, pp. 49–63. Springer, Heidelberg (2006)
2. Amin, S., Bayen, A.M., El Ghaoui, L., Sastry, S.S.: Robust feasibility for control of water flow in a reservoir canal system. In: Proceedings of the 46th IEEE Conference on Decision and Control, pp. 1571–1577 (2007)

3. Ben-Tal, A., Boyd, S., Nemirovski, A.: Control of uncertainty-affected discrete time linear systems via convex programming. Technical Report, Minerva Optimization Center, Technion, Haifa, Israel (2005)
4. Byres, E., Lowe, J.: The myths and facts behind cyber security risks for industrial control systems. In: Proceedings of the VDE Congress, VDE Association for Electrical Electronic & Information Technologies (2004)
5. Calafiore, G.C., El Ghaoui, L.: Linear programming with probability constraints. In: Proceedings of the 2007 American Control Conference, New York, USA (2007)
6. Chatterjee, K., de Alfaro, L., Henzinger, T.A.: Termination criteria for solving concurrent Safety and reachability games (2008), http://arxiv.org/abs/0809.4017
7. Cárdenas, A.A., Amin, S., Sastry, S.: Research Challenges for the Security of Control Systems. In: 3rd USENIX workshop on Hot Topics in Security (HotSec 2008). Associated with the 17th USENIX Security Symposium, San Jose, CA (2008)
8. Downs, J.J., Vogel, E.F.: A plant-wide industrial process control problem. Computers & Chemical Engineering 17(3), 245–255 (1993)
9. Gattami, A.: Optimal decision with limited information. PhD thesis, Department of Automatic Control, Lund University (2007)
10. Goulart, P.J., Kerrigan, E.C., Maciejowski, J.M.: Optimization over state feedback policies for robust control with constraints. Automatica 42(2), 523–533 (2006)
11. Mayne, D.Q., Rawlings, J.B., Rao, C.V., Scokaert, P.O.M.: Constrained model predictive control: stability and optimality. Automatica 36(6), 789–814 (2000)
12. Nguyen, K.C., Alpcan, T., Basar, T.: A decentralized Bayesian attack detection algorithm for network security. In: Proc. of 23rd Intl. Information Security Conf., Milan, pp. 413–428 (2008)
13. Pinar, A., Meza, J., Donde, V., Lesieutre, B.: Optimization strategies for the vulnerability analysis of the power grid. Submitted to SIAM Journal on Optimization (2008)
14. Primbs, J., Sung, C.: Stochastic receding horizon control of constrained linear systems with state and control multiplicative noise. Submitted to IEEE Transactions on Automatic Control (2007)
15. Salmeron, J., Wood, K., Baldick, B.: Analysis of electric grid security under terrorist threat. IEEE Transactions on power systems 19, 905–912 (2004)
16. Schenato, L., Sinopoli, B., Franceschetti, M., Poolla, K., Sastry, S.: Foundations of control and estimation over lossy networks. Proceedings of the IEEE, Special issue on networked control systems 95(1), 163–187 (2007)
17. Skaf, J., Boyd, S.: Design of affine controllers via convex optimization. Submitted to the IEEE Transactions on Automatic Control (2008)
18. Turk, R.J.: Cyber Incidents Involving Control Systems. Technical Report, Idaho National Laboratory (2005)
19. van Hessem, D., Bosgra, O.: A full solution to the constrained stochastic closed-loop MPC problem via state and innovations feedback and its receding horizon implementation. In: Proceedings of the 2003 Conference on Decision and Control, Maui, Hawaii, USA (2003)
20. Wang, Y., Boyd, S.: Fast model predictive control using online optimization. Submitted to IEEE Transactions on Control Systems Technology (2008)
21. Yu, Z.H., Li, W., Lee, J.H., Morari, M.: State estimation based model predictive control applied to shell control problem: a case study. Chemical engineering science 49(3), 285–301 (1994)

Appendix

$$\mathbf{x_0} := \begin{pmatrix} I_n \\ A \\ A^2 \\ \vdots \\ A^N \end{pmatrix} x_0 \in \mathbb{R}^{n(N+1)}, \quad \mathbf{A} := \begin{pmatrix} 0 & 0 & 0 & \dots & 0 \\ I_n & 0 & 0 & \dots & 0 \\ A & I_n & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ A^{N-1} & A^{N-2} & A^{N-3} & \dots & I_n \end{pmatrix} \in \mathbb{R}^{n(N+1)\times nN},$$

$$\mathbf{B} := \mathbf{A}(I_N \otimes B) = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 \\ B & 0 & 0 & \dots & 0 \\ AB & B & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ A^{N-1}B & A^{N-2}B & A^{N-3}B & \dots & B \end{pmatrix} \in \mathbb{R}^{n(N+1)\times mN},$$

$$\mathbf{\Gamma} = \operatorname{diag}(\gamma_0^{N-1}) \otimes I_n = \begin{pmatrix} \gamma_0 I_n & & \\ & \ddots & \\ & & \gamma_{N-1} I_n \end{pmatrix} \in \mathbb{R}^{nN\times nN},$$

$$\mathbf{N} = \operatorname{diag}(\nu_0^{N-1}) \otimes I_m = \begin{pmatrix} \nu_0 I_m & & \\ & \ddots & \\ & & \nu_{N-1} I_m \end{pmatrix} \in \mathbb{R}^{mN\times mN},$$

and

$$\mathbf{e_0} = \begin{pmatrix} I_n \\ (1-\gamma_0)A \\ (1-\gamma_0)(1-\gamma_1)A^2 \\ \vdots \\ \prod_{j=0}^{N-1}(1-\gamma_j)A^N \end{pmatrix} e_0 \in \mathbb{R}^{n(N+1)}$$

$$\mathbf{H} = \begin{pmatrix} 0 & 0 & \dots & 0 \\ I_n & 0 & \dots & 0 \\ (1-\gamma_1)A & I_n & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \prod_{j=1}^{N-1}(1-\gamma_j)A^{N-1} & \prod_{j=2}^{N-1}(1-\gamma_j)A^{N-2} & \dots & I_n \end{pmatrix} \in \mathbb{R}^{n(N+1)\times nN}$$

Actors without Directors: A Kahnian View of Heterogeneous Systems*

P. Caspi[1], A. Benveniste[2], R. Lublinerman[3], and S. Tripakis[1]

[1] Verimag, Grenoble, France
[2] INRIA/IRISA, Rennes, France
[3] Pennsylvania State University

Abstract. This paper aims to simplify recent efforts proposed by the Berkeley school in giving a formal semantics to the Ptolemy toolbox. We achieve this by developing a simple and elegant *functional* theory of deterministic tag systems that is a generalisation of Kahn Process Network theory (KPN). Our theory extends KPN by encompassing networks of processes labelled by tags from *partially ordered* sets and makes deeper use of Scott theory of Complete Partial Orders (CPO). Since CPO compose well under direct sums, heterogeneous systems are simply captured by *direct sums of homogeneous systems*, which are in turn constructed by connecting systems over different tag sets by means of *tag conversion* processes. For the (large) class of tag systems of "stream" type, we show how to define tag conversion processes and how to implement process communication. The resulting architecture is fully decentralised and does not require Ptolemy's directors. Last but not least, it provides distribution for free.

1 Introduction

The semantics of heterogeneity. The need for heterogeneity in modelling and development tools has been increasing while applications are becoming more and more complex. In view of this state of matters, pioneering frameworks like Ptolemy [11,13] which have started addressing the issue of heterogeneity a long time ago are becoming always more popular and raising an ever growing interest. Thus, concepts of this framework like *models of computation and communication* (MoCC), *actors*, *directors*, and so on, have been getting an increasingly larger acceptance.

Among the problems raised by this subject, the semantic question is important. While homogeneous applications are in general well-mastered, problems start at their interfaces, when several subsystems are composed to form a larger application. Ambiguities, semantic inconsistencies, etc., are likely to produce undesired behaviours which can badly impair the overall functioning of the composed application. To this end, Lee & Sangiovanni have introduced their celebrated tagged signal model [10] which was meant to provide a precise semantics to such frameworks as Ptolemy.

* This work was funded in part by the European Artist-Design Network of Excellence and the European STREP-COMBEST project number 215543.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 46–60, 2009.

Yet, there was still a large gap between this denotational formalism and the behaviour of Ptolemy which is still largely bound to the operational semantics of the simulation engine. Efforts have been devoted to filling this gap: for instance BIP which is based on operational semantics [2] and 42 [14] which provides building blocks for designing MoCCs in a comprehensive way.

The application of Scott theory to tag systems. A comprehensive step toward closing this gap has recently been taken in [11], so as to make things simpler by getting rid of non-determinism, that is, by restricting from relations to functions. After all, determinism is something designers are fond of, most simulators like Ptolemy are deterministic and, when non-determinism is needed, in most cases it can be emulated by adding extra inputs to functions, aiming at choosing between several possible futures.[1]

Yet, this was not sufficient: when composing functions, inputs of one function can become outputs of another one and conversely, creating feedback loops and resulting in the functional aspect being lost: we get systems of equations which can have no solution as well as several solutions.

But this is a well-known issue of denotational semantics and well-known solutions exist. The most widely adopted one is Scott's semantics [15]: if the domain of interest is a complete partial order (CPO) and we restrict ourselves to continuous functions, then we know that every system of equations has a least solution and it is sensible to decide that this is the semantics of the system. Moreover, the least solution is itself a continuous function of its free inputs and thus can in turn be composed at will. The framework is thus closed by composition (and even by lifting to higher orders) and works perfectly well.

But there was another problem. The basic objects of the tagged signal model are signals which in a deterministic point of view can be seen as functions from tags to values. Scott approaches turn these signals into CPOs by turning the value set into a CPO. In this way, the CPO property gets automatically lifted from the image set to the function set. Thus, in this Scott theory applied to tag systems, the tag set does not need to have any order property. But, in tag system theory, tag sets are partially ordered and have a strong time flavour: in Ptolemy, computations go from past to future, while in the Scott framework, it does not matter (tags may not have any order and there may be neither past nor future!).

Towards Kahn semantics. Thus [13] had to modify Scott's order by requiring a prefix ordering principle in the spirit of the Kahn order [9]: a signal is larger than another one not only if it is more defined but also if both signals are defined on some initial segment of the tag set. In this way, a signal s_1 is larger than another signal s_2 if the initial segment over which s_1 is defined is larger than the initial segment over which s_2 is defined. In this way, computations can only extend the initial segments on which signals are defined and naturally flow from past to future.

[1] This is the way probability theory works: by adding input spaces about which the only knowledge we can have is their probability measure.

There was still a problem due to the fact that some tag sets, for instance associated with the discrete event (DE) domain, are infinite in several dimensions: in this case, initial segments are infinite and thus a signal defined over an initial segment has to have an infinity of values. In some sense, time may not progress, as in the so-called *Zeno* phenomenon of timed systems. But it is not possible to compute an infinite number of values in a simulator. In [13,11] the problem is solved using the idea of *absent value* from the French synchronous language school [3]: thus a signal defined on an initial segment may have only a finite number of computed non-absent values (while absent values need not be computed).

Paper's objectives and organisation. In this paper we develop a simple and elegant *functional* theory of deterministic tag systems that is a generalisation of Kahn's theory of Process Networks (KPN); KPN theory is recalled in section 2. As developed in section 3, our theory extends KPN by encompassing networks of processes labelled by tags from *partially ordered* sets and makes deeper use of Scott theory of Complete Partial Orders (CPO); since CPO compose well under direct sums, heterogeneous systems are simply captured by *direct sums of homogeneous systems*, which are in turn constructed by connecting systems over different tag sets by means of *tag conversion* processes. For the (large) class of tag systems of *stream* type introduced in section 4, we show how to implement process communication and how to define tag conversion processes (see section 6). Examples of tag systems are provided in section 5. Finally, we show in section 6 that the resulting architecture: 1) is fully decentralised; 2) does not require Ptolemy's directors, and 3) provides distribution for free. An extended presentation of this work can be found in [1].

2 Background on Deterministic Tag Systems and Kahn Theory

Signals, Deterministic Signals, and Processes. The basic idea of the Tagged Signal Model [10] is to consider a signal $x \in S$ as a set of *events,* consisting of a pair "(tag, value)". Signals can thus be formalised as: $S = \{s \mid s \subseteq \mathbb{T} \times V\}$, where V is a set of values, and $(\mathbb{T}, \leq)$ is a partially ordered set of tags. These signals are non-deterministic ones: several values can be associated with the same tag. As we aim at considering deterministic tag systems, we first need to consider deterministic signals. This amounts to saying that we only consider signals that are partial functions from tags to values which we denote as: $S = \mathbb{T} \hookrightarrow V$.

In the deterministic setting, processes (or actors, following the Ptolemy terminology) are just functions transforming input signals into output signals. For the sake of simplicity, we do not consider the types of signal values and assume an "universal" type V. Thus, the set of processes, P, is just the set of total functions from S^m to S^n: $P = S^m \mapsto S^n$, where m, n are the input and output arities.

Functional Composition and Feedback Loops. In this deterministic setting, things are very simple. Processes are composed by functional composition and a composed process is just a system of equations, *e.g.,*

$$x_3 = p_1(x_1, x_2) \qquad x_4 = p_2(x_1, x_3)$$

which can define another process p_3 such that $(x_3, x_4) = p_3(x_1, x_2)$. However, this raises the question of feedback loops: for instance consider the system:

$$x_3 = p_1(x_1, x_2) \qquad x_2 = p_2(x_1, x_3)$$

What does it compute? This system of equations may have no solution or it may have several solutions. Then determinism can be lost.

Scott Semantics. Scott semantics [15] provides a well-known solution to this issue. It consists of the following changes to the previous framework:

1. Add to V an undefined element $\perp$ and a partial order relation $\leq$ such that:
   - $V^\perp = V \cup \{\perp\}$
   - $\leq$ is the least order relation over $V^\perp$ generated by: $\forall v \in V, \perp \leq v$.

   This makes $(V^\perp, \leq, \perp)$ a (flat) CPO. $\perp$ is the least element of $V^\perp$ and any sequence of ordered elements (a chain) has a least upper bound ($\bigvee$) which is $\perp$ if the chain contains only $\perp$'s, or some v_1 if the chain contains this v_1: note that in the latter case the chain cannot contain another v_2 distinct from v_1 as the two are incomparable.
2. Redefine S as the set of *total* functions from $\mathbb{T}$ to $V^\perp$.[2] $S = \mathbb{T} \mapsto V^\perp$. Then S inherits the CPO property of $V^\perp$ by defining:
   - $x \leq x'$ if for all $t \in \mathbb{T}$, $x(t) \leq x'(t)$ which amounts to saying that x is smaller than x' if it is less defined,
   - the bottom element of S, also denoted $\perp$, as the signal which is undefined everywhere: $\perp(t) = \perp$.

   Given a chain of signals $x_0, \ldots x_n, \ldots$, and given any tag t, $x_0(t), \ldots x_n(t), \ldots$ is chain of values and

$$\bigvee\{x_0, \ldots x_n, \ldots\}(t) = \bigvee\{x_0(t), \ldots x_n(t), \ldots\}$$

3. Restrict processes to *continuous* functions from input to output signals, which means that, given a chain of inputs, that is to say a sequence of more and more defined signals, the outputs should form a chain and

$$\bigvee\{p(x_0), \ldots p(x_n), \ldots\} = p(\bigvee\{x_0, \ldots x_n, \ldots\}) \tag{1}$$

   Note that continuity implies *order preservation*: $s \leq s' \Rightarrow p(s) \leq p(s')$ and note that this definition for single-input/single-output processes can be extended naturally to processes of different arities because products of CPOs inherit the CPO structure of their components. In particular, the order on the product is the component-wise order.

[2] A partial function can be made total by giving it the value $\perp$ whenever it is not defined.

4. Then the Kleene theorem says that *any system of equations has a (unique) least solution, which is in turn a continuous function of its free input signals.* Thus composition preserves determinism and confluence of unscheduled distributed executions is guaranteed.

Kahn Theory. But this solution is still unsatisfactory because it does not take advantage of the ordering over tags which have a flavour of time. In particular, a process may as well compute from future to past—we can easily design a process that is continuous in the Scott sense but not causal. This issue of causality is properly addressed by Kahn theory.

Kahn's world is a special case of Scott's world. In Kahn's world, the tag domain is $\mathbb{N}$, which is a totally-ordered and enumerable set. Signals are partial functions from $\mathbb{N}$ to a set of values V. In addition, all signals are assumed to be *prefix-closed*, meaning that if they are undefined at some time n then they remain undefined for all $n' > n$.

Note that in Kahn's original paper [9] signals are elements of $V^\infty = V^* \cup V^\omega$ where V^* is the set of all finite sequences over V and V^ω is the set of all infinite sequences over V. The set of all prefix-closed signals from $\mathbb{N}$ to V is isomorphic to V^∞: partially-defined signals correspond to finite sequences and totally-defined signals to infinite sequences.

Looking at signals x and y as sequences, $x \leq y$ means that x is a prefix of y (there exists a sequence x' such that $y = x \cdot x'$, where $x \cdot x'$ denotes the concatenation of x and x'). With this order, the set of all signals becomes a poset. It is in fact a CPO: (1) $\bot$ is the empty sequence ϵ; (2) the least upper bound of a chain of increasingly defined signals is either: (2.1) the most defined one if the chain is finite or: (2.2) the infinite sequence defined by the chain, because an infinite sequence is a maximal element of the CPO (concatenating a sequence to an infinite sequence does not change this sequence).

Processes are assumed to be continuous functions from S^m to S^n. Again we can define the semantics as the least solution of systems of equations. Thus the Kahn theory solves the feedback loop problem. But it also solves the causality problem: a continuous process is order preserving and, in terms of Kahn order, this means that if x is a prefix of y, it is also the case that $f(x)$ is a prefix of $f(y)$. This means that the future of x cannot influence the present of x. Computations are guaranteed to flow from past to future.

3 Kahn Generalisation to Partially Ordered Tag Sets

Kahn network over a Partially Ordered Tag set. Kahn signals can be seen as tagged signals over the particular tag set $\mathbb{N}$. In order to address heterogeneity, we would like to generalise Kahn's approach to other tag sets. The technical difficulty here is that in general, tag signals cannot be prefix-closed because prefixes can be infinite. While in [13,11] this problem is solved by introducing a special "absent" value, we propose here an alternative, more general

approach, that does not require absent values and is still compatible with infinite prefixes.

Let S be the set of all total functions

$$S = \mathbb{T} \mapsto V^{\perp}, \tag{2}$$

where $\mathbb{T}$ is a poset. Following Kahn, we endow S with the following partial order:

Definition 1 (Prefix order over signals). *A signal x is a prefix of a signal y, if for all t, $y(t) \neq x(t)$ implies for all $t' \geq t$, $x(t') = \perp$.*

It is easy to see that this is indeed an order relation. Please note also that this definition allows "holes" in the defined values; for instance we could have:

$$x : 1, \perp, 2, \perp, \perp, \perp \ldots \qquad y : 1, \perp, 2, \perp, 3, \perp \ldots$$

In the above example we have indeed $x \leq y$. Note that in this example $\mathbb{T} = \mathbb{N}$.

Proposition 1 (CPO). *S endowed with the prefix order is a CPO.*

Proof. First, take $\perp(t) = \perp$. Then we notice that given $x \leq y$ in S, for any t, $x(t) \leq y(t)$ according to the CPO $V^{\perp}$. Thus, if $\{x_n\}$ is a chain, then, for any t, $\{x_n(t)\}$ is a chain and we can take: $\bigvee\{x_n\}(t) = \bigvee\{x_n(t)\}$.

Definition 2. *A process p is* order-preserving *if, for any two signals, x, y if x is a prefix of y, then $p(x)$ is a prefix of $p(y)$; p is* continuous *if it satisfies (1).*

This means that only the past can influence the present value of a process. The mathematical framework of this section is both simple and very powerful. Restricting to order-preserving processes allows us to preserve *determinism, causality,* and *confluence* of unscheduled distributed executions.

Capturing Heterogeneity via Sum of CPOs. The next problem is to extend the previous generalised Kahn theory to encompass heterogeneity, that is, systems involving different tag sets. But this in our framework comes for free: The *sum* of two CPOs S_1 and S_2 is a CPO $S_1 + S_2$, defined as

$$S_1 + S_2 = (S_1 - \{\perp_{S_1}\}) \cup (S_2 - \{\perp_{S_2}\}) \cup \{\perp\}$$

where $\perp, \perp_{S_1}$, and $\perp_{S_2}$ are the corresponding bottom elements. The order on each of S_1, S_2 is maintained in the sum, but two elements from two different sets are not comparable. Therefore, chains can only be formed of elements of a single set and the least upper bounds are preserved.

Heterogenous architectures. At this point, suppose that we know how to construct *tag conversion functions,* i.e., continuous functions

$$f : S_1 \to S_2. \tag{3}$$

Let us see now how such a function can be seen as operating on the sum CPO $S_1 + S_2$. Indeed, $f : S_1 \to S_2$ can be seen as a function $f' : (S_1 + S_2) \to (S_1 + S_2)$ by setting:

$$f'(\bot) = \bot, \qquad f'(in_1(x)) = f(x), \qquad f'(in_2(x)) = \bot$$

where in_1, in_2 are the canonical injection of each CPO into the sum. Next, consider the following toolkit of functions, consisting of:

- homogeneous functions, mapping input signals to output signals belonging to a domain S of signals over a same partially ordered tag set $\mathbb{T}$;
- tag conversion functions, mapping an input signal over $\mathbb{T}_1$ to an output signal over $\mathbb{T}_2$.

By using the previous reasoning, a finite network of such functions can be seen as a network of homogeneous functions acting on the direct sum $\sum_{i \in I} S_i$, where finite set I indexes the set of homogeneous functions of the considered network. By proposition 1, the network itself is an homogeneous function acting on the direct sum $\sum_{i \in I} S_i$. Thus this network itself can be encapsulated as a function acting on $S =_{\text{def}} \sum_{i \in I} S_i$, so the same construction can be reused, hierarchically. Observe that we can also encapsulate tuples of signals over different tag sets as a single signal defined over the sum of the considered tag sets. In other words, hierarchy can be used for both boxes (functions) and wires (signals). Having this architecture model addresses the main objective of this paper.

The remaining problems. From the previous analysis, the following two central issues remain to be addressed:

Problem 1. How to construct tag conversion functions?

Having a solution to problem 1 provides us immediately with a framework of heterogeneous Kahn-like architectures, as explained just above.

Problem 2. How to implement wires carrying signals defined over a partially ordered tag set $\mathbb{T}$*?*

This problem also remains to be solved in order to make our approach effective—recall that such an implementation exists for basic Kahn networks since the latter rely on a communication medium of unbounded FIFOs. Other media may be needed for other partially ordered tag sets. We address Problem 1 in Section 4 and Problem 2 in Section 6 and show how to solve them in the restricted case of "streams".

4 Streams: From Generalised to Ordinary Kahn Theory

In the generalised Kahn theory developed in section 3, signals are total functions from a partially ordered tag set $\mathbb{T}$ to a set of values $V^{\bot}$, or, equivalently, partial functions from $\mathbb{T}$ to V. These signals can thus be seen as "labelled partial orders" and are fairly general. Yet, in many practical cases, for instance those which

correspond to what is considered in Ptolemy [13] and which are addressed in section 5, this generality is not needed and the approach can be simplified. This simplification is based on two assumptions. When these assumptions are in force, signals can be seen as streams and the theory boils down to an ordinary Kahn theory. While this reduction is unnecessary from a mathematical standpoint, it has practical applicability, since we know that *Kahn networks* (unlike general tag systems of section 3) *are implementable on networks of processors related by FIFO links, cf. our problem 2.*

Assumption 1 (DTOS). *In the considered set S of signals:*

1. *The tag set is a total order.*
2. *The defined values of any signal can be indexed in non-decreasing order, meaning that there is an order preserving isomorphism from the domain of any signal, $dom(x) = \{t \mid x(t) \neq \perp\}$, to (an initial segment of) $\mathbb{N}$.*

Call Discrete over Totally Ordered tag Set (DTOS) *such a set S of signals.*

This means that there is an order preserving isomorphism from the domain of a signal ($dom(x) = \{t \mid x(t) \neq \perp\}$) to (an initial segment of) $\mathbb{N}$. We call this a *discrete signal*. Assumption 1 yields signals whose defined tags are order isomorphic to (an initial segment of) $\mathbb{N}$.[3] In this case we speak of a *discrete total order*.

An important question arising from Assumption 1 is whether the restriction of signals based on these assumptions still preserves the CPO structure defined in proposition 1. This is by no means a trivial issue. The following proposition provides a positive answer.

Proposition 2 (DTOS). *The set of* Discrete signals over a Totally Ordered tag set (DTOS) *endowed with the prefix order is indeed a CPO.*

The proof can be found in [1].

DTOS signals as streams. When dealing with DTOS signals, tags associated to defined values can be indexed in increasing order and signals can be seen as streams of pairs (value, tag). In other words, to a DTOS signal x we can associate its *stream* $St(x) : (V \times \mathbb{T})^{\infty}$ where there is no more need to consider an undefined value.[4] The tag ordering constraint is then for any $s \in DTOS$ such that $s = (v_1, t_1).(v_2, t_2).s'$,

1. $t_1 < t_2$
2. $s' \in$ DTOS

It is then clear that the stream view of DTOS signals enjoys the same properties as the functional one. Formally the DTOS-to-stream transformation is as follows:

$$\begin{aligned} St(\perp) &= \epsilon, \text{ the empty sequence} \\ St(x) &= (x(t_1), t_1).St(x[t_1 \rightarrow \perp]) \end{aligned} \tag{4}$$

where:

[3] Note the importance of requiring an order-preserving isomorphism in Assumption 1. Rationals are both totally ordered and countable but not order isomorphic to $\mathbb{N}$.

[4] This view is inspired by our previous work [4].

– t_1 is the least tag yielding a defined value in x
– $x[t_1 \rightarrow \bot]$ is the function x where the value at t_1 has been changed to $\bot$.

To conclude this section, we formally state the following property:

Proposition 3. *St defined in (4) is a CPO isomorphism between* DTOS *signals and streams. Moreover, it preserves parallel composition. (This solves problem 2.)*

Thus, DTOS systems can be brought back to streams, i.e., ordinary Kahn networks.

5 Examples

Kahn Process Networks. Kahn Process Networks (KPN) naturally fit into that landscape. $\mathbb{N}$ is the tag set and there are no "holes" between defined values. Thus signals are just streams of defined values. An operator like the sum operator over numbers can be lifted to streams according to the following Haskell-like definition:

$$\begin{aligned} \mathrm{sum}_K\ \epsilon\ y &= \mathrm{sum}_K\ x\ \epsilon; = \epsilon \\ \mathrm{sum}_K\ v.x\ v'.y &= (v+v').\mathrm{sum}_K\ x\ y \end{aligned}$$

where ϵ denotes the empty stream and "." denotes concatenation. Basically, the Kahn actor sum_K waits until both its input queues are non-empty. Then, it removes their heads, adds them, puts the result into its output queue and starts again. Note that waiting until both queues are non-empty can be implemented using Kahn's *blocking read* operator, without having to test both queues *simultaneously*: sum_K simply blocks on one queue, then on the other. The order in which queues are read can be arbitrary.

Discrete Event. We begin with a tagged view of Discrete Event Signals and then present a streamed view for them.

A Tagged View of Discrete Event (DE) Signals. Discrete event (DE) signals are discrete signals (according to assumption 1) with real-time stamps. A tag set for DE is:

$$\mathbb{T} = \mathbb{R}_+ \times \mathbb{N}$$

where $\tau = (t, n)$, t denotes a time stamp, and n is the index of events sharing the same time stamp.[5] This tag set is ordered with the lexicographic order:

$$(t, n) \leq (t', n') \text{ iff either } t < t' \text{ or } t = t' \text{ and } n \leq n'$$

which is a total order. Then a discrete event signal x is a total function $x : \mathbb{T} \mapsto V^{\bot}$ satisfying the following constraint: for any two tags $\tau, \tau' \in \mathbb{T}$, with $\tau = (t, n)$, $\tau' = (t, n')$, and $n \leq n'$, if $x(\tau') \neq \bot$ then $x(\tau) \neq \bot$. Figure 1 shows an example of such a signal. In this example the following table provides the correspondence between tags and values:

[5] This approach, which has been called *super-dense time* by some authors [13], could be easily extended to tag sets $\mathbb{T} = \mathbb{R}_+ \times \mathbb{N}^{\mathbb{N}}$ to account for so-called "nested oversamplings". For the sake of simplicity, we do not address such an extension here.

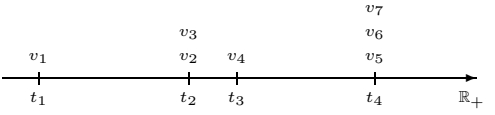


Fig. 1. A discrete-event signal

$$\begin{array}{rccccccc} \text{tag}: & (t_1,1) & (t_2,1) & (t_2,2) & (t_3,1) & (t_4,1) & (t_4,2) & (t_4,3) \\ \text{value}: & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 \end{array}$$

We can remark that in this definition, there are many undefined (or absent) values namely between two consecutive time stamps holding defined values and after the last defined value sharing a given time stamp.

A Streamed View of Discrete Event Signals. DE signals are DTOS, so we can apply the results of section 4, thus providing a streamed view of them:

$$sx : (V \times (\mathbb{R}_+ \times \mathbb{N}))^\infty$$

where $\mathbb{R}_+$ is the set of non-negative reals modelling the physical (or *real*) time. Furthermore we observe that in this definition, the second component $\mathbb{N}$ of the tag set is not necessary because we can always rebuild it by applying the following index rebuilding mapping $\mathrm{Ir} : (V \times \mathbb{R}_+)^\infty \to (V \times (\mathbb{R}_+ \times \mathbb{N}))^\infty$:

$$\begin{array}{lll} \mathrm{Ir}_1(t', n, \epsilon) & = \epsilon & \\ \mathrm{Ir}_1(t', n, (v,t).sx) & = \text{if } t == t' & \\ & \quad \text{then } (v,(t,n+1)).\mathrm{Ir}_1(t,n+1,sx) & \\ & \quad \text{else } (v,(t,1)).\mathrm{Ir}_1(t,1,sx) & \\ \mathrm{Ir}(\epsilon) & = \epsilon & \\ \mathrm{Ir}((v,t).sx) & = (v,(t,1)).\mathrm{Ir}_1(t,1,sx) & \end{array}$$

An Actor Example. It is interesting to see how to define some primitive actors in DE. Let us start by defining the sum of two signals. There are several ways of defining it, each having, perhaps surprisingly, very different properties [1]. Here we present only one possibility, which states that, when both input signals appear with the same tag, we output the sum, otherwise we just output the defined signal:

$$\begin{array}{lll} \mathrm{sum}_{DE2}\ x\ \epsilon & = \mathrm{sum}_{DE2}\ \epsilon\ y\ = \epsilon & \\ \mathrm{sum}_{DE2}\ (v,\tau).x\ (v',\tau').y & = \text{if } \tau < \tau' & \\ & \quad \text{then } (v,\tau).\mathrm{sum}_{DE2}\ x\ (v',\tau').y & \\ & \quad \text{else if } \tau = \tau' & \\ & \qquad \text{then } (v+v',\tau).\mathrm{sum}_{DE2}\ x\ y & \\ & \qquad \text{else } (v',\tau')'.\mathrm{sum}_{DE2}\ (v,\tau).x\ y & \end{array}$$

Figure 2 illustrates this definition which can be proved to be continuous ([1]) though it uses, unlike in KPN, the infamous[6] operation that tests values in input queues without removing (consuming) them.

[6] Because its undisciplined usage may result in non-continuous processes.

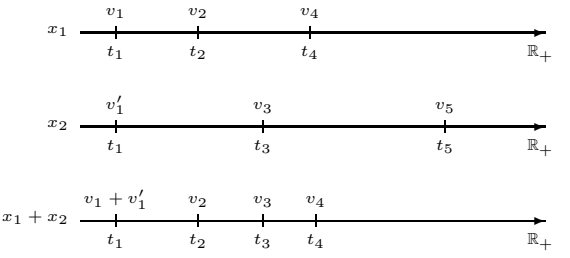


Fig. 2. Sum (DE2) of two discrete-event signals

Other operators can be found in [1].

Continuous Time. In general, the continuous time (CT) case is more involved, and its study is part of our on-going work. Some preliminary ideas can be found in [1]. We summarise these here.

First, note that there are different CT domains, depending on whether we want to define signals with *exact* (i.e., ideal) CT semantics, or *approximate* CT semantics, as computed using a numerical solver. Exact semantics is linked to the theory of ordinary differential equations (ODEs) (see also [12] and [6]). Yet it seems to us that exact CT does not fit into the Kahn landscape: in order to exactly solve a differential equation, this equation has to be considered globally as a whole and it cannot be decomposed into its components. For instance we cannot define the exact behaviour of an integrator, independently from the network of operators which feeds it. Indeed, in theory we would need to check whether this network computes a Lipschitz function or not.

Regarding approximate semantics, observe that ODEs, when discretised using explicit schemes with fixed step size, are simply DE systems, thus can be handled in the DE domain. However, this no longer holds if more sophisticated schemes are used, e.g. implicit schemes and/or variable step size.

Synchronous Reactive. Synchronous Reactive systems have been addressed among others in [5,7]. It is in this domain that absent values have been first introduced. Here also, we begin with a *Tagged view of Synchronous Reactive* (SR) systems. These are very similar to discrete event ones, but real-time is replaced with a logical integer time. Thus the tag set is[7] $\mathbb{N} \times \mathbb{N}$ where the first component gives the reaction logical time and the second one the multiplicity index in that reaction. The reason we need both components is to model easily *multi-clock* systems: in such systems, a signal may be "absent" in some reactions (captured by $\perp$) and occur several times in other reactions. In some sense, the reaction logical time acts as a replacement of absent values.

[7] The same remark on a possible extension to nested oversamplings as stated at section 5 applies here.

For the *Streamed view of Synchronous Reactive*, we can proceed as with DE signals: $St(x) : (V \times \mathbb{N})^{\infty}$. In this interpretation $St(x)(n) = (v, r)$ means the n-th occurrence of x has value v and takes place within the r-th reaction.

An example that is not DTOS. So far all examples we have presented are DTOS. It will not be the case for the following one, however. The MoCC we present here is that of signals that are themselves streams of events, however with causality relations between events belonging to different signals. To formalise this example we need an underlying set X of signal *names*. The set $\mathbb{T}$ of tags has the following form, where $\mathbb{N}_{\infty} = \mathbb{N} \cup \{-\infty\}$:

$$\mathbb{T} = X \times (X \to \mathbb{N}_{\infty})$$

In other words, $t \in \mathbb{T}$ has the form $t = (x, \tau)$, where τ is a *vector clock*, i.e., a total function mapping X to $\mathbb{N}_{\infty}$. The interpretation of $t = (x, \tau)$ is as follows:

- tag t belongs to a signal with name x;
- signal x is indexed by the set of positive integers $\mathbb{N}$ and its rank is given by $n = \tau(x)$, which must therefore be $> -\infty$;
- a positive value for $\tau(y) = m > -\infty$, where $y \in X \setminus \{x\}$ indicates a causality constraint of the nth event of signal x with respect to the mth event of signal y; having $\tau(y) = -\infty$ indicates lack of causality constraint of the nth event of signal x with respect to any event of y.

We may (but do not need to) restrict $\mathbb{T}$ to tags whose vector clock τ takes a value $\tau(y) \neq -\infty$ for only finitely many y's. $\mathbb{T}$ is equipped with the following order relation, making it a partial order:

$$t \geq t' \text{ iff } \forall y \in X \text{ s.t. } \tau'(y) > -\infty \Rightarrow \tau(y) \geq \tau'(y)$$

6 Actors without Directors

6.1 Tag Conversion Actors in Lieu of Directors

When dealing with heterogeneity, there is generally no "golden rule" saying what the meaning of composing actors with different tag sets should be. This information must instead be provided by the designer.

In Ptolemy this problem is solved using the concept of *directors*. Roughly speaking, a director schedules the operation of a set of concurrent actors in time, thus in essence defining the concurrency (and time) semantics of the model. There are many types of directors in Ptolemy, each implementing a given MoCC: discrete-event, synchronous-reactive, etc.

Here, we take a different approach: we define *tag-conversion* actors, *i.e.,* heterogeneous actors operating on different tag sets and transforming signals on one tag set to signals on another tag set. Compared to directors, our approach has two main advantages. First, we do not need to introduce an additional concept in our modelling framework, actors is all we need. Second, our approach allows to separate the issue of semantic compatibility from that of using different MoCCs.

We give now some standard conversion actors to allow the interconnection of signals of different tags. These are only a few examples and other tag-conversion actors can obviously be defined.

From DE and SR to KPN. Going from DE and SR to KPN can be done by "forgetting" the tag:

$$\begin{array}{lll} \text{forget} & \epsilon & = \epsilon \\ \text{forget} & (x,t).xs & = x.\text{forget}\ \ xs \end{array}$$

From KPN to DE and SR. In the opposite direction, a "timestamping" actor can be used. This actor uses a clock that specifies the timestamps:

$$\begin{array}{lll} \text{timestamp} & \epsilon\ cl & = \epsilon \\ \text{timestamp} & xs\ \epsilon & = \epsilon \\ \text{timestamp} & (x.xs)\ (t.ts) & = (x,t).(\text{timestamp}\ xs\ ts) \end{array}$$

6.2 Distribution

A stated at the end of section 3, restricting to order-preserving processes in the sense of definition 2 allows us to preserve determinism, causality, and *confluence of unscheduled distributed executions*. Thus, distribution comes for free and does not need coordination. This holds in particular for heterogeneous models mentioned at the end of section 6.1.

Still, the following issue remains, namely: *which type of communication link is needed in such distributed implementations?* Since tag conversion is performed by actors, *links involve only homogeneous tag sets*. So, in general, our (directed) links only need to preserve the prefix order of definition 1 for a given (homogeneous) tag set $\mathbb{T}$, from source node to sink node. This holds in particular for heterogeneous models mentioned at the end of section 6.1. In particular, standard FIFO links can be used to implement communications for such architectures.

Take for instance the definition of the discrete event sum illustrated in figure 2 of section 5. The actor has two input FIFO queues `x` and `y` and an output FIFO queue `z`. The queues contain pairs $(v : V, t : \mathbb{R}_+)$. Indeed the $\mathbb{N}$ component of the DE tag set is useless because the FIFO queues preserve the order of production. Thus an operational version of the sum actor can be defined in C-like syntax as:

```
void sum(input queue x, input queue y, output queue z) {
  if (x.empty() OR y.empty()) return;
  if (x.head().tag() < y.head().tag()) {
     z.append(x.head()); x.erase_head(); return; }
  if (x.head().tag() == y.head().tag()) {
     z.append(x.head().tag(), x.head().val() + y.head().val());
     x.erase_head(); y.erase_head(); return; }
  // it must be that: x.head().tag() > y.head().tag()
  z.append(y.head()); y.erase_head(); return;
}
```

Basically, the sum process needs that its two input files be non-empty to execute. Otherwise it waits. If it can execute, it takes the two tagged heads and

compares their tags. If they are equal, it sums up the two values, tags the result with the common tag, puts it in the output queue and erases the two heads from the input queues. If the two tags are different, the earlier tagged value is erased from its input queue (as a matter of fact we know that, since the input queue values are produced in an orderly manner, it will not be possible that the other queue will later contain an item matching this earlier tag) and the other queue is left unchanged. Nothing is produced and the process waits.

Indeed, we could say, adopting the Ptolemy terminology that such networks do not need directors, *i.e.,* some *deus ex machina* able to schedule the executions of each actor. In a simulation engine, the only need is that execution is fairly distributed between actors in such a way that no actor is infinitely excluded from execution. Also note that there is no "event queue" like what is found in most simulation engines like Ptolemy and this feature avoids the burden of building a distributed event queue. Distributed actors are truly autonomous, they only know of the heads of their input queues.

6.3 Hierarchy

It has been advocated that the use of directors enforces a clean separation between several MoCCs in a hierarchical way: in order to get a communication between two different MoCCs these have to be encapsulated within a "larger" MoCC which encompasses the former ones. It is true that this is a good design practice but the "flat" directorless approach we present here is fully compatible with hierarchy: Kahn actors can be gathered so as to form compound actors and this hierarchical composition can be extended at will.

7 Conclusion

This paper has intended to simplify recent efforts proposed by the Berkeley school in giving a formal semantics to the Ptolemy toolbox. We have proposed a simple and elegant functional theory of deterministic tag systems that is a generalisation Kahn's theory of Process Networks (KPN). Our theory encompasses networks of processes labelled by tags from *partially ordered* sets and makes deeper use of Scott theory of Complete Partial Orders (CPO). Since CPO compose well under direct sums, heterogeneous systems are simply captured by *direct sums of homogeneous systems*, which are in turn constructed by connecting systems over different tag sets by means of *tag conversion* processes. For the (large) class of tag systems of *stream* or DTOS type, we have shown how to define tag conversion processes and how to implement process communication. The resulting architecture is fully decentralised and does not require Ptolemy's directors. Last but not least, it provides distribution for free.

A natural question is to find broader frameworks than just DTOS in which problems 1 and 2 can be properly solved. This is left for future work.

An important issue not addressed in the paper is the issue of liveness (also called productivity in the co-algebraic framework). This issue has already been

partially addressed in [13] and its adaptation to our stream approach will be a subject for future work.

Another semantic theory for stream-based systems, alternative to Kahn, is the co-algebraic theory of streams (see for instance [8]). Basically, moving to co-algebraic streams would consist, in the stream programs shown in the paper, to remove the ϵ cases. Indeed, this is what has been done in the Haskell prototype we have implemented of our framework. Examining the consequences of such an alternative choice is also a subject for future work.

References

1. Full version of this paper available as technical report TR-2008-6, http://www-verimag.imag.fr/index.php?page=techrep-list
2. Basu, A., Bozga, M., Sifakis, J.: Modeling Heterogeneous Real-time Components in BIP. In: SEFM 2006, pp. 3–12 (2006)
3. Benveniste, A., Berry, G.: The synchronous approach to reactive and real-time systems. IEEE Proceedings 79, 1270–1282 (1991)
4. Benveniste, A., Caillaud, B., Carloni, L.P., Caspi, P., Sangiovanni-Vincentelli, A.L.: Composing heterogeneous reactive systems. ACM Trans. Embedded Comput. Syst. 7(4) (2008)
5. Berry, G., Sentovich, E.: An implementation of constructive synchronous programs in polis. Formal Methods in System Design 17, 135–161 (2000)
6. Bliudze, S., Krob, D.: Towards a functional formalism for modelling complex industrial systems. In: Complex Systems (ECCS 2005), pp. 163–176 (2005)
7. Edwards, S.A., Lee, E.A.: The semantics and execution of a synchronous block-diagram language. Science of Computer Programming 48(1) (2003)
8. Jacobs, B., Rutten, J.: A tutorial on (co)algebras and (co)induction. Bulletin of EATCS 62, 229–259 (1997)
9. Kahn, G.: The semantics of a simple language for parallel programming. In: IFIP (1974)
10. Lee, E.A., Sangiovanni-Vincentelli, A.: A unified framework for comparing models of computation. IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems 17(12), 1217–1229 (1998)
11. Lee, E.A., Zheng, H.: Leveraging synchronous language principles for heterogeneous modeling and design of embedded systems. In: EMSOFT 2007 (2007)
12. Liu, J., Lee, E.A.: On the causality of mixed-signal and hybrid models. In: Maler, O., Pnueli, A. (eds.) HSCC 2003. LNCS, vol. 2623, pp. 328–342. Springer, Heidelberg (2003)
13. Liu, X., Lee, E.A.: CPO Semantics of Timed Interactive Actor Networks. Theoretical Computer Science 409(1), 110–125 (2008)
14. Maraninchi, F., Bouhadiba, T.: 42: Programmable models of computation for a component-based approach to heterogeneous embedded systems. In: GPCE (2007)
15. Scott, D.: Data types as lattices. SIAM J. on Computing 10(3), 522–587 (1976)

Simultaneous Optimal Control and Discrete Stochastic Sensor Selection*

D. Bernardini[1], D. Muñoz de la Peña[2], A. Bemporad[1,**], and E. Frazzoli[3]

[1] Department of Information Engineering, University of Siena, Italy
{bernardini,bemporad}@dii.unisi.it
[2] Dep. de Ingeniería de Sistemas y Automática, University of Seville, Spain
davidmps@cartuja.us.es
[3] Massachusetts Institute of Technology, MA, USA
frazzoli@mit.edu

Abstract. In this paper we present the problem of combining optimal control with efficient information gathering in an uncertain environment. We assume that the decision maker has the ability to choose among a discrete set of sources of information, where the outcome of each source is stochastic. Different sources and outcomes determine a reduction of uncertainty, expressed in terms of constraints on system variables and set-points, in different directions. This paper proposes an optimization-based decision making algorithm that simultaneously determines the best source to query and the optimal sequence of control moves, according to the minimization of the expected value of an index that weights both dynamic performance and the cost of querying. The problem is formulated using stochastic programming ideas with decision-dependent scenario trees, and a solution based on mixed-integer linear programming is presented. The results are demonstrated on a simple supply-chain management example with uncertain market demand.

1 Introduction

A large number of problems in production planning and scheduling, location, transportation, finance, and engineering design require taking optimal decisions in the presence of uncertainty. Uncertainty, for instance, governs the prices of fuels, the availability of electricity, and the demand for chemicals. In general, these uncertainties affect the constraints of the corresponding optimization problem. A standard approach to deal with uncertain constraints is to guarantee constraint satisfaction for all possible cases. In order to reduce the conservativeness of this solution, additional information about the uncertainties may be gathered, for example by carrying a demand field study to better estimate the value of future demand of a certain product in a production planning problem. With this additional information, the optimization problem is updated in a less conservative

* This work was partially supported by the European Commission under the HYCON Network of Excellence, contract number FP6-IST-511368, and under the WIDE project, contract number FP7-IST-224168.

** Corresponding author.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 61–75, 2009.

way and an improved solution is obtained. In addition, with the current developments in networked control systems (NCS) [18,7,12], efficient information gathering has become a very relevant problem in modern industrial automation. Possible examples of this framework are given by control over wireless networks, where communication is subject to strong energy constraints, and more in general by any kind of NCS in which measurement acquisition is expensive. For such process control problems a selection criterion for the kind of information that is convenient to retrieve is recommended.

In general, however, the outcome of these information queries is not known a priori. Moreover, queries have fixed costs that do not depend on the quantitative outcome of the information gathered, i.e., costs associated with the querying process per se. This poses a difficult problem of whether a query would be profitable or not. The difficulty increases when there are several possible queries at hand and, even more difficult, when a whole sequence of queries must be planned. There are different ways of approaching the problem. It can be cast as a Markov decision problem (MDP) [13], but the cardinality of the state space of this representation grows exponentially with the number of events, due to the number of possible combinations of events which could take place. Hence, the exact solution of such a problem becomes computationally intractable very quickly, even for relatively small problems. The approach taken in [6] for a similar problem (the bridge problem) is based on reinforcement learning, which is a set of techniques aimed at approximating the MDP value function. We refer the reader to the literature on the subject for further details [4,17].

In this paper we take a different route and propose a stochastic recursive optimization scheme in which we have to decide not only an optimal sequence of future control actions, but also which measurements/queries are worth to be carried out. Each query is defined by its own fixed cost and a series of possible outcomes described by a discrete probability function. The constraints on the sequence of future actions and performance indices depend on such outcomes. Consequently, the optimal control problem becomes stochastic as well, for which we employ a stochastic programming formulation to minimize expected values under stochastic constraint sets. Stochastic programming is a special class of mathematical programming that involves optimization under uncertainty (see [5,9,14]). The first applications of stochastic programming date back to the 50's and nowadays it is becoming a mature theory that is successfully applied in several domains [15]. A stochastic programming problem is defined by a sequence of random events and recourse decisions. Each decision is a different stage and stages are divided by random events. In the proposed formulation, there are two stages, that is, two sets of decision variables separated by a random event: First the query has to be chosen without knowing the outcome of the response, then the outcome of the query is obtained (the random event takes place) and the second stage decision (the dynamic optimization variables) is made based on this information.

For long optimization horizons, we advocate the use of recursive shorter-horizon optimization to obtain suboptimal solutions within a manageable

computational burden (see for example [3] for the application of recursive stochastic hybrid optimal control in the management of power distribution networks). The proposed scheme is demonstrated on a supply-chain management example in which the future demand is uncertain, but additional information can be obtained from market studies.

2 Stochastic Querying Model

Consider the generic problem of linear programming (LP)

$$\begin{aligned} \min_{z} \ & c'z \\ \text{s.t.} \ & z \in \mathcal{Z}, \end{aligned} \tag{1}$$

in which $\mathcal{Z}$ is a polyhedron that defines the region of feasibility. As in general by expanding $\mathcal{Z}$ one improves the optimum achieved in (1), we consider the case in which we can perform a query Q in order to obtain additional information that allows us to enlarge the size of $\mathcal{Z}$. The main idea is that in the presence of uncertainty, if a robust approach is taken, the feasible set takes into account all possible values of uncertain parameters. Hence, by obtaining additional information that reduces the set of possible values of the uncertain parameters, the size of $\mathcal{Z}$ increases, and hence the optimal cost is improved. We define a query Q as follows:

Definition 1. *A* query $Q(q) \in \mathcal{Q}$ *is defined as*

$$Q(q) = \{C(q), \mathcal{V}_q\}, \ q \in \{0, 1, \ldots, n_q\}, \tag{2}$$

where q *is the* query index*,* $C(q) \geq 0$ *is the* querying cost*, and* $\mathcal{V}_q = \{V_1^q, V_2^q, \ldots, V_{m_q}^q\}$ *is the set of the* m_q *possible* outcomes*.*

Definition 2. *A* query outcome V_v^q *for the query* q *is defined as*

$$V_v^q = \{\mathcal{Z}(q, v)\}, \tag{3}$$

where $v \in \{1, 2, \ldots, m_q\}$ *is the* outcome index*, and* $\mathcal{Z}(q, v)$ *is the* updated feasibility set*.*

Note that, in general, the number m_q of possible outcomes depends on the query q. For compactness of notation, in the sequel we will often refer to a "query" $Q(q)$ directly by its corresponding query index q, and to an "outcome" V_v^q directly by its corresponding outcome index v. To model the case where the source of information is not queried at all, we introduce the *null query*, indexed by $q = 0$ and defined below.

Definition 3. *The* null query *is defined as*

$$Q(0) = \{0, \mathcal{V}_0\}, \qquad \mathcal{V}_0 = \left\{V_1^0 = \{\mathcal{Z}\}\right\}, \tag{4}$$

The query $Q(q)$ can be chosen among the finite set $\mathcal{Q}$ of different queries, however the information obtained from each query is stochastic. Each query is defined by a cost $C(q)$ and a set of possible outcomes $\mathcal{V}_q$ with a given probability, which we assume to be available.

Definition 4. *For every query* $q \in \{0, 1, \ldots, n_q\}$*, the* outcome probability distribution *is a discrete distribution given by*

$$\mathcal{P}_i = \left\{ p_{ij} : p_{ij} = Pr[v = V_j^i | q = i],\ j = 1, 2, \ldots, m_i,\ \sum_{j=1}^{m_i} p_{ij} = 1 \right\}. \quad (5)$$

The objective is to choose the query $Q(q) \in \mathcal{Q}$ such that J_q is minimized, where J_q is the expected value of the cost function with respect to the possible outcomes of the query plus the cost of the query itself

$$\min_{q \in \mathcal{Q}} \ \mathbb{E}_v \left[J_{qv} | q \right] + C(q), \quad (6)$$

with J_{qv} the optimal cost corresponding to the outcome v of the query q defined as

$$J_{qv} = \min_z \ c'z \quad (7a)$$

$$\text{s.t. } \ z \in \mathcal{Z}(q, v). \quad (7b)$$

Problem 7 can be posed as a two-stage stochastic optimization problem [5,9,14]. As observed earlier, here the query q is the first-stage variable, and z is the second-stage variable which is decided after the random outcome event v takes place. In the following section we propose to apply this general framework to the the problem of combining optimal control with efficient information gathering in an uncertain environment.

3 Simultaneous Optimal Control and Sensor Selection Problem

Consider the discrete-time linear model of the process

$$x(t+1) = Ax(t) + Bu(t), \quad (8)$$

where the input $u \in \mathbb{R}^{n_u}$ and the input rate $\Delta u(t) = u(t) - u(t-1)$ are subject to known component-wise constraints[1] $u_{\min} \leq u \leq u_{\max}$, and $\Delta u_{\min} \leq \Delta u \leq \Delta u_{\max}$. The state $x \in \mathbb{R}^{n_x}$ is subject to uncertain constraints. The only available a priori information on the admissible state set is given by the set-membership relation $x \in \mathcal{X}$, where $\mathcal{X}$ is a conservative estimate of the admissible state set that guarantees robust constraint satisfaction for all possible values of queries

[1] Here component-wise constraints are considered for simplicity, but it is straightforward to extend the approach to the more general case of polytopic constraints.

and outcomes. The goal of the control action is to make the state $x(t)$ and the input $u(t)$ track an uncertain reference value $r_x(t)$, $r_u(t)$, respectively, where $r(t) = \begin{bmatrix} r_x(t) \\ r_u(t) \end{bmatrix} \in \mathcal{R}$. The set $\mathcal{R}$ is a conservative estimate of all the possible values that the reference can take[2].

Without additional information, a recursive optimal control problem formulation based on model (8), the conservative estimates $\mathcal{X}$ and $\mathcal{R}$, and a min-max cost function can be formulated using standard min-max model predictive control ideas [16,1,11]. We refer to this problem as the standard min-max problem. However, in this paper, we assume that every T time steps the decision maker is allowed to reduce the conservativeness by querying additional sources of information at a certain cost. This additional information in general may provide a reduced conservativeness on the admissible state sets (i.e., a larger domain $\mathcal{X}$), and/or a better estimate of the reference $r(t)$ (i.e., a smaller domain $\mathcal{R}$). In both cases, the obtained solution is less conservative, with consequent improvement of the overall performance of the process. By following the problem formulation of Section 2, the outcome v related to the query q is denoted by

$$V_v^q = \{\mathcal{X}(q, v|t), \mathcal{R}(q, v|t)\}, \tag{9}$$

where $\mathcal{X}(q, v|t)$, $\mathcal{R}(q, v|t)$ are the updated state constraints and reference sets, such that $\mathcal{X}(q, v|t) \supseteq \mathcal{X}(t)$, $\mathcal{R}(q, v|t) \subseteq \mathcal{R}(t)$. Moreover, the outcome set for the *null* query is

$$V_1^0 = \{\mathcal{X}(t), \mathcal{R}(t)\}, \tag{10}$$

where $\mathcal{X}(t)$, $\mathcal{R}(t)$ is the available information at time t on state constraints and reference set. Note that we consider here problems in which the estimates of the uncertain sets are time varying. This may be the case for instance in which the outcomes obtained after each query accumulate.

The querying mechanism can be modeled in different ways, for example by introducing delays between the query transmission and the availability of the outcome. For simplicity, in the following sections we restrict ourselves to the following assumption.

Assumption 1. *The outcome V_v^q of a query $Q(q)$ performed at time step t is immediately available, and the provided information is supposed to be significant only for time step $t + 1$.*

We aim at defining a stochastic optimal control setup that, at each time step t, provides at the same time a sequence of optimal input values $u(t)$, $u(t+1)$, …, $u(t+N-1)$, $N \geq 1$, and the most profitable query $q(t)$, by taking into account model (8), the set of possible outcomes (9), and the corresponding probability distributions (5). As mentioned before, we assume that a query can be done every T time steps, where T is constant and such that $T \geq 1$. We also assume that a query is done at time step $t = 0$. This implies that a query will be done

[2] Note that this setup can be easily extended to the case of bounded additive disturbances, as they can be modeled without loss of generality by means of more conservative state or input constraints.

at time steps $t = kT$, $k \in \mathbb{Z}$, $k \geq 0$. Given a generic time t, the next query will be henceforth carried out at time $t + H$, where

$$H = \left\lceil \frac{t}{T} \right\rceil T - t, \tag{11}$$

and where $\lceil a \rceil$ denotes the smallest integer greater than or equal to a. When H is smaller than the optimal control horizon N, the future query has to be decided by the optimal decision mechanism, otherwise a standard min-max problem (no query) is solved.

Note that in general, instead of choosing off-line a constant value for T, any time-varying, state-dependent interval $T(t)$ could be considered, as long as the condition $0 < H(t) < N \Rightarrow H(t+1) \leq H(t) - 1$ is enforced to preserve the consistency of the receding horizon control.

Based on the above description, at time step $t \in \mathbb{N}$ the simultaneous optimal control and sensor selection problem is defined as

$$\min_{q} \ \mathbb{E}_v \left[J_{qv} | q \right] + cC(q) \tag{12a}$$

$$\text{s.t.} \ \begin{cases} q \in \{0, 1, 2, \ldots, n_q\} \text{ if } H < N, \\ q = 0 \qquad\qquad\quad \text{otherwise,} \end{cases} \tag{12b}$$

with

$$\begin{aligned} J_{ij} \ = \ \min_{\Delta u} \Bigg\{ & \max_{r} \sum_{k=0}^{N-1} \ell(x(t+k,i,j|t) - r_x(t+k,i,j|t), \\ & u(t+k,i,j|t) - r_u(t+k,i,j|t), \Delta u(t+k,i,j|t)) \Bigg\} \end{aligned} \tag{13a}$$

$$\text{s.t. } x(t+k+1,i,j|t) = Ax(t+k,i,j|t) + Bu(t+k,i,j|t), \tag{13b}$$

$$u(t+k,i,j|t) = u(t+k-1,i,j|t) + \Delta u(t+k,i,j|t), \tag{13c}$$

$$u_{\min} \leq u(t+k,i,j|t) \leq u_{\max}, \tag{13d}$$

$$\Delta u_{\min} \leq \Delta u(t+k,i,j|t) \leq \Delta u_{\max}, \tag{13e}$$

$$x(t+k,i,j|t) \in \begin{cases} \mathcal{X}(i,j|t) \text{ if } k = H+1, \\ \mathcal{X}(t) \qquad \text{otherwise,} \end{cases} \tag{13f}$$

$$r(t+k,i,j|t) \in \begin{cases} \mathcal{R}(i,j|t) \text{ if } k = H+1, \\ \mathcal{R}(t) \qquad \text{otherwise,} \end{cases} \tag{13g}$$

$$x(t,i,j|t) = x(t|t), \tag{13h}$$

$$\Delta u(t+h,i,j|t) = \Delta u(t+h,w,z|t), \ \forall w \neq i, \ \forall z \neq j, \tag{13i}$$

$$h = 0, 1, \ldots, \min(H, N) - 1,$$

$$k = 0, 1, \ldots, N - 1,$$

for $i = 0, 1, \ldots, n_q$, $j = 1, 2, \ldots, m_i$, where $x(t|t) = x(t)$ is the current state, used as the initial condition for the optimal control problem, $x(t+k,i,j|t)$, $r(t+k,i,j|t)$, $u(t+k,i,j|t)$, $\Delta u(t+k,i,j|t)$ are the predicted state, the input, the

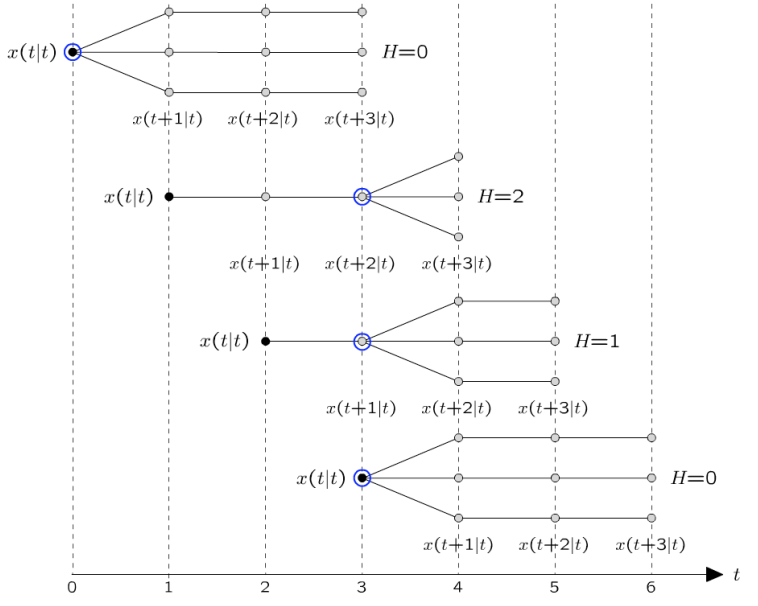


Fig. 1. Optimization tree structure for different time steps t, where to enforce causality the inputs are not branched until the query is performed ($N = T = 3$, circled dots denote decisions taken on q)

input rate and uncertain reference at time step $t+k$ corresponding to making a query i with the outcome j at time $t+H$, $r = \left[\begin{smallmatrix} r_x \\ r_u \end{smallmatrix}\right]$, $c \geq 0$ is the tradeoff coefficient between performance and querying costs, N is the prediction horizon, T is the time period between two consecutive queries, and the stage cost $\ell : \mathbb{R}^{n_x+2n_u} \to \mathbb{R}$ is a nonnegative function.

The positive scalar H defined in (11) represents the time step at which a query decision will take place. Until that time, the causality constraint (13i) enforces the same input sequence for all the possible sequences of states, regardless of the dependence on future decision on q. Note that the optimization problem has the time-varying structure depicted in Figure 1, as the imposed constraints at time t depend on the current value of $H = \left\lceil \frac{t}{T} \right\rceil T - t$.

In principle, Problem 13 is an infinite dimensional optimization problem, due to the maximization part that involves an infinite number of realizations of the reference r. However, it is well known that when the process is linear and the constraints and the cost function are convex, the max problem can be solved by considering only the "extreme" realizations, namely the vertices of the reference set $\mathcal{R}$ (see, e.g., [16]). In the next section we will exploit this property to reformulate Problem 13 as a stochastic mixed-integer linear programming (MILP) problem.

According to the aforementioned stochastic optimization nomenclature [5], Problem 12 is a two-stage optimization problem in which the second-stage

variables are Δu's. Since only one decision on q is modeled in the problem, the proposed formulation is exact with respect to the system behavior only for $N \leq T$. For $N > T$ a more complex *multi-stage* stochastic programming formulation would be necessary. By using the two-stage formulation (12) also when $N > T$, i.e., by modeling just the first decision on q, one gets a conservative solution which does not exploit all the available information, but nonetheless is computationally more viable.

The following Algorithm 1 summarizes the proposed recursive stochastic simultaneous optimal control and sensor selection decision mechanism.

Algorithm 1. Recursive stochastic simultaneous optimal control and sensor selection.

For all $t \geq 0$**:**
- **1.** get $x(t)$ and compute H as in (11);
- **2.** solve Problem 12 and get the optimal solution $q^*(t)$, $u^*(t, i, j|t)$, $\forall i, j$;
- **3. if** $H = 0$
  - **3.1.** perform the query $q^*(t)$ and get the query outcome $v^*(t)$;
  - **3.2.** set $u(t) = u(t, q^*, v^*|t)$ in (8);
- **4. else**
  - **4.1.** set $u(t) = u(t, 0, 1|t)$ in (8);
- **5. end.**

Next section focuses on computational methods for solving Problem 12. A closed-loop stability analysis of the receding horizon control scheme proposed by Algorithm 1 is beyond the scope of this paper and will addressed in future works, based on an adaptation of convergence properties existing for deterministic min-max model predictive control schemes [16] to the present stochastic min-max setting.

4 Solution Methods

Let the stage cost ℓ be based on infinity norms

$$\ell\left(x - r_x, u - r_u, \Delta u\right) = ||Q_x(x - r_x)||_\infty + ||Q_u(u - r_u)||_\infty + ||Q_{\Delta u}\Delta u||_\infty, \quad (14)$$

and, for the sake of generality, assume that a terminal cost

$$\ell\left(x(t + N, i, j|t) - r_x\right) = \|Q_N(x(t + N, i, j|t) - r_x)\|_\infty \quad (15)$$

is added in the cost function (13a), where Q_x, Q_u, $Q_{\Delta u}$, Q_N are full row-rank matrices, and $\|Qx\|_\infty = \max_{i=1,\ldots,n_x} |Q_i x|$ with Q_i the ith row of Q[3]. In this case Problem 12 can be solved using an MILP problem by following the so-called "scenario enumeration" approach of stochastic programming [5], as detailed below, where we exploit the convexity of (14), (15), to get rid of the max problem

[3] The results of this paper extend to any convex piecewise affine function ℓ.

in (12) through enumeration of vertices, introduce slack variables that upper bound each stage term of the stage cost [1], and use big-M techniques to transform a multiplication between a binary variable and a continuous variable into a set of linear constraints [19]. The case of quadratic cost in (14) can also be handled similarly, by using mixed-integer quadratic programming (MIQP). Then, Problem 12 can be formulated as the following MILP

$$
\begin{aligned}
\min_{\delta,\Delta u,F,\gamma} \quad & \sum_{i=0}^{n_q} F_i && \text{(16a)}\\
\text{s.t.} \quad & x(t+k+1,i,j|t) = Ax(t+k,i,j|t) + Bu(t+k,i,j|t),\\
& u(t+k,i,j|t) = u(t+k-1,i,j|t) + \Delta u(t+k,i,j|t),\\
& u_{\min} \le u(t+k,i,j|t) \le u_{\max},\\
& \Delta u_{\min} \le \Delta u(t+k,i,j|t) \le \Delta u_{\max},\\
& x(t+k,i,j|t) \in \begin{cases} \mathcal{X}(i,j|t) \text{ if } k = H+1,\\ \mathcal{X}(t) \quad \text{otherwise,} \end{cases}\\
& x(t,i,j|t) = x(t|t),\\
& \Delta u(t+h,i,j|t) = \Delta u(t+h,w,z|t),\ \forall w \ne i,\ \forall z \ne j,\\
& \gamma_{ij}^{kx} \ge \|Q_x(x(t+k,i,j|t) - r_x(t+k,i,j|t))\|_\infty,\\
& \quad k = 0,\dots,N-1,\\
& \gamma_{ij}^{ku} \ge \|Q_u(u(t+k,i,j|t) - r_u(t+k,i,j|t))\|_\infty,\\
& \quad k = 0,\dots,N-1,\\
& \gamma_{ij}^{k\Delta u} \ge \|Q_{\Delta u}\Delta u(t+k,i,j|t)\|_\infty,\\
& \quad k = 0,\dots,N-1,\\
& \underbrace{\gamma_{ij}^{Nx} \ge \|Q_N(x(t+N,i,j|t) - r_x(t+N,i,j|t))||_\infty,}_{\forall r(t+k,i,j|t) \in \begin{cases} \mathcal{R}_v(i,j|t) \text{ if } k = H+1,\\ \mathcal{R}_v(t) \quad \text{otherwise,} \end{cases}} && \text{(16b)}\\
& -M\delta_i \le F_i \le \sum_{j=1}^{m_i} p_{ij}(t)\left(\gamma_{ij}^{Nx} + \sum_{k=0}^{N-1}\gamma_{ij}^{kx} + \gamma_{ij}^{ku} + \gamma_{ij}^{k\Delta u}\right)\\
& \quad + cC(i) + M(1-\delta_i), && \text{(16c)}\\
& M\delta_i \ge F_i \ge \sum_{j=1}^{m_i} p_{ij}(t)\left(\gamma_{ij}^{Nx} + \sum_{k=0}^{N-1}\gamma_{ij}^{kx} + \gamma_{ij}^{ku} + \gamma_{ij}^{k\Delta u}\right)\\
& \quad + cC(i) - M(1-\delta_i), && \text{(16d)}\\
& \sum_{i=0}^{n_q}\delta_i = 1,\ \delta_i \in \{0,1\},\ i = 0,\dots,n_q,\\
& h = 0,1,\dots,\min(H,N)-1,\\
& i = 0,1,\dots,n_q,\ j = 1,2,\dots,m_i,\ k = 0,\dots,N-1.
\end{aligned}
$$

where the array of binary variables $\delta = \{\delta_0, \delta_1, \dots, \delta_{n_q}\}$, $\delta_i \in \{0,1\}$, $i = 0,\dots,n_q$, one for every possible query choice, is used to choose the query to

be done among all possibilities; that is, if query i is chosen, then $\delta_i = 1$ and the rest are equal to zero. The slack variables γ_{ij}^{kx}, γ_{ij}^{ku}, $\gamma_{ij}^{k\Delta u}$, γ_{ij}^{Nx} in (16b) define the value of the min-max problem (13) for every couple (i, j), where $\mathcal{R}_v(t)$, $\mathcal{R}_v(i, j|t)$ are the sets of the vertices of $\mathcal{R}(t)$ and $\mathcal{R}(i, j|t)$, respectively. Note that (16b) are linear constraints, since in general $\gamma \geq \|z\|_\infty$ can be rewritten as $\gamma \geq \pm z_i$, $\forall i$. By means of the big-M constraints (16c)-(16d), all the continuous variables F_i take zero value, except for the one referred to the chosen query. Then, the cost function (16a) is equivalent to (12a). M is a large enough positive scalar, satisfying the condition

$$\begin{aligned} M \geq \textstyle\sum_{j=1}^{m_i} p_{ij}(t)\big(&\|Q_N(x(t+N,i,j|t) - r_x(t+N,i,j|t))||_\infty \\ &+ \sum_{k=1}^{N-1} \|Q_x(x(t+k,i,j|t) - r_x(t+k,i,j|t))\|_\infty + \|Q_u(u(t+k,i,j|t) \\ &- r_u(t+k,i,j|t))\|_\infty + \|Q_{\Delta u}\Delta u(t+k,i,j|t)\|_\infty\big) + cC(i), \end{aligned}$$

for all $i = 0, 1, \ldots, n_q$. Note that it is not strictly necessary to model the input sequences for all the possible pairs (q, v), since only $\max_q(m_q)$ scenarios are evaluated simultaneously. However, in this case a number of additional constraints would be needed, resulting in a higher computational burden. Reducing the number of the inputs can be desirable if some or all of them are integer variables.

5 Illustrative Example

The use of receding horizon control policies in supply chain management have been investigated in [10,8], and approached by hybrid techniques in [2]. In this paper we consider the supply chain shown in Figure 2, where a single product is processed through a network of four nodes. A product is distributed, stored, and sold to the customer. The goal of the control problem is to minimize a performance index, mainly given by the satisfaction of customer demand and production costs, while fulfilling constraints on production, storage and transport capacities. The process is modeled as

$$F_1(t+1) = F_1(t) + P_1(t) - T_{11}(t) - T_{12}(t), \tag{17a}$$
$$F_2(t+1) = F_2(t) + P_2(t) - T_{21}(t) - T_{22}(t), \tag{17b}$$
$$R_1(t+1) = R_1(t) + T_{11}(t) + T_{21}(t) - D_1(t), \tag{17c}$$
$$R_2(t+1) = R_2(t) + T_{12}(t) + T_{22}(t) - D_2(t), \tag{17d}$$

where, at time t, $P_i(t)$ is the number of products which enter the supply chain and are stored in Factory i, $T_{ij}(t)$ is the number of transported products from Factory i to Retailer j, and $D_j(t)$ is the number of products sold by Retailer j.

We define the state vector $x = \begin{bmatrix} F_1 & F_2 & R_1 & R_2 \end{bmatrix}' \in \mathbb{R}^4$ and the input vector $u = \begin{bmatrix} P_1 & P_2 & T_{11} & T_{12} & T_{21} & T_{22} & \bar{D}_1 & \bar{D}_2 \end{bmatrix}' \in \mathbb{R}^8$, where $\bar{D}_i$ is the nominal value for the

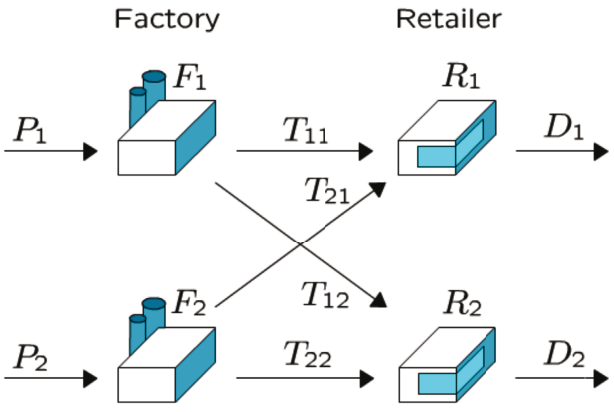


Fig. 2. Supply chain scheme

demand D_i. Then, the dynamics (8) of the supply chain model is described by the matrices

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & -1 \end{bmatrix}. \tag{18}$$

The bounds on states and inputs are

$$x_{max} = \begin{bmatrix} 100 & 100 & 100 & 100 \end{bmatrix}', \; u_{max} = \begin{bmatrix} 100 & 100 & 50 & 50 & 50 & 50 & 100 & 100 \end{bmatrix}', \tag{19a}$$

$$x_{min} = \begin{bmatrix} 0 & 0 & 20 & 20 \end{bmatrix}', \; u_{min} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}', \tag{19b}$$

respectively, and input increments are considered unbounded. In addition, the model is subject to the following constraints on product availability:

$$T_{11}(t) + T_{12}(t) \leq F_1(t), \tag{20a}$$

$$T_{21}(t) + T_{22}(t) \leq F_2(t), \tag{20b}$$

$$D_1(t) \leq R_1(t), \tag{20c}$$

$$D_2(t) \leq R_2(t). \tag{20d}$$

In this example the state constraints are fully known, but customer demand is uncertain. In particular, we assume that at every time step t customer demand can be described by two probability distributions, called *low mode* and *high mode*, respectively. They are essentially modeled as a mixture of Gaussians, normalized in the demand space:

$$\begin{bmatrix} D_1(t) \\ D_2(t) \end{bmatrix} \sim \begin{cases} \frac{\mathcal{N}(\mu_0, \Sigma_0)}{\int_0^{80} \int_0^{80} \mathcal{N}(\mu_0, \Sigma_0) dD_1 dD_2} & \text{with 70\% prob. } \textit{(low mode)} \\ \frac{\mathcal{N}(\mu_0, \Sigma_0) + 0.75\mathcal{N}(\mu_1, \Sigma_1)}{\int_0^{80} \int_0^{80} (\mathcal{N}(\mu_0, \Sigma_0) + 0.75\mathcal{N}(\mu_1, \Sigma_1)) dD_1 dD_2} & \text{with 30\% prob. } \textit{(high mode)} \end{cases} \tag{21}$$

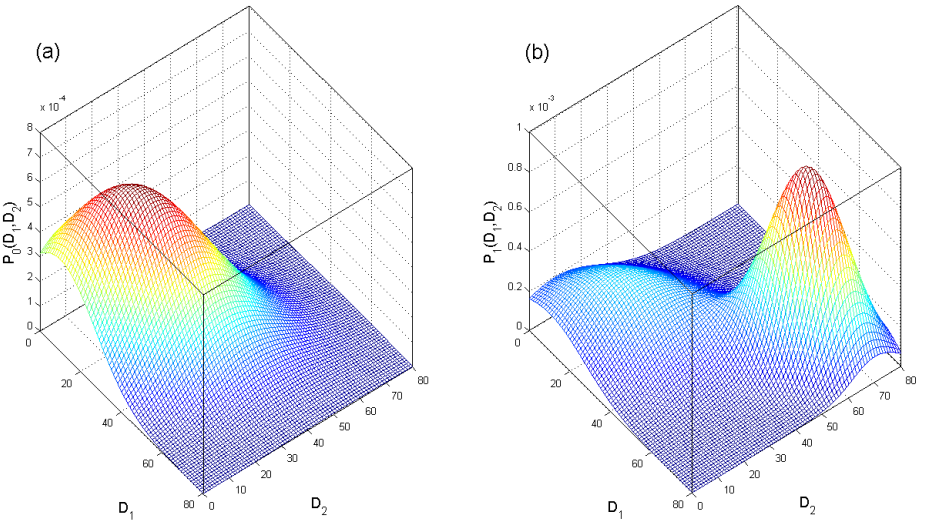


Fig. 3. Probabilistic models for customer demand in low mode (a) and high mode (b)

where $\mu_0 = \begin{bmatrix} 20 \\ 20 \end{bmatrix}$, $\mu_1 = \begin{bmatrix} 60 \\ 60 \end{bmatrix}$, $\Sigma_0 = \begin{bmatrix} 400 & 200 \\ 200 & 400 \end{bmatrix}$, $\Sigma_1 = \begin{bmatrix} 100 & 50 \\ 50 & 100 \end{bmatrix}$. The probability distributions of the customer demand associated to each of the two modes are shown in Figure 3. The decision maker is supposed to know the current demand mode at each time step by freely available market polls, but to ignore the exact value of the demand. Henceforth the reference values are time-varying and uncertain. Let $\mathcal{R}_D$ denote the subset of the reference set related to the demand D_1, D_2. We assume to be able to perform two different queries to get an approximated description of the actual demand probability distribution: The first is simpler and cheaper, the second is more accurate but more expensive. The numerical values for available queries and their outcomes are given in Table 1. Note that by Assumption 1 we consider a query outcome to be reliable only for the time step following the time at which the query was sent.

We consider four different decision-making policies:

(1) A *deterministic* policy obtained by setting $q(t) = 0$, $\forall t$, corresponding to a standard min-max problem where no additional information is retrieved by the querying mechanism (LP);
(2) A *stochastic random* policy in which $q(t)$ is picked up randomly in $\{1, \ldots, n_q\}$, $\forall t$, and, therefore, $q(t)$ does not depend on the current state $x(t)$ of the model (LP);
(3) A *stochastic heuristics-based* policy in which $q(t)$ is selected according to deterministic conditions on the market state (LP). The following rule is applied: at time t, if the market is in *high mode*, select $q(t) = 2$, else select $q(t) = 1$;
(4) The *stochastic optimized* policy of Problem 12 (MILP).

Table 1. Queries and outcomes definition

(q,v)	$C(q)$	p_{qv} in Low Mode	p_{qv} in High Mode	$\mathcal{R}_D(q,v\|t)$
$(0,1)$	0	$p_{01,L}=1$	$p_{01,H}=1$	$r_{min}=[\ 0\ \ 0]$, $r_{max}=[80\ 80]$
$(1,1)$	1	$p_{11,L}=0.804$	$p_{11,H}=0.423$	$r^{11}_{min}=[\ 0\ \ 0]$, $r^{11}_{max}=[40\ 80]$
$(1,2)$	1	$p_{12,L}=0.196$	$p_{12,H}=0.577$	$r^{12}_{min}=[40\ \ 0]$, $r^{12}_{max}=[80\ 80]$
$(2,1)$	15	$p_{21,L}=0.684$	$p_{21,H}=0.351$	$r^{21}_{min}=[\ 0\ \ 0]$, $r^{21}_{max}=[40\ 40]$
$(2,2)$	15	$p_{22,L}=0.120$	$p_{22,H}=0.072$	$r^{22}_{min}=[\ 0\ 40]$, $r^{22}_{max}=[40\ 80]$
$(2,3)$	15	$p_{23,L}=0.120$	$p_{23,H}=0.072$	$r^{23}_{min}=[40\ \ 0]$, $r^{23}_{max}=[80\ 40]$
$(2,4)$	15	$p_{24,L}=0.076$	$p_{24,H}=0.505$	$r^{24}_{min}=[40\ 40]$, $r^{24}_{max}=[80\ 80]$

Table 2. Simulation results

Control policy	Performance J_{exp}	Avg. CPU time
Deterministic min-max with *null* query	401.90	16.0 ms
Stochastic min-max with random query selection	360.84	16.9 ms
Stochastic min-max with heuristic query selection	345.76	16.7 ms
Stochastic min-max with optimized query selection	319.74	36.3 ms

We run $N_s = 10$ simulations of $T_{sim} = 10$ time steps each, using parameters $T = 1$, $N = 4$, $c = 1$, $Q_{\Delta u} = 0$, $Q_x = Q_N = \mathrm{Diag}([0.1,\ 0.1,\ 0.2,\ 0.2])$, $Q_u = \mathrm{Diag}([10,\ 10,\ 0.1,\ 0.2,\ 0.2,\ 0.1,\ 10,\ 10])$. The initial state is $x(0) = [40\ 40\ 60\ 60]'$. Table 2 shows the obtained results in terms of the achieved average performance evaluated as

$$J_{exp} = \frac{1}{N_s T_{sim}} \sum_{i=1}^{N_s} \sum_{t=1}^{T_{sim}} \big(\|Q_x(x^i(t) - r_x^i(t))\|_\infty + \|Q_u(u^i(t) - r_u^i(t))\|_\infty + \|Q_{\Delta u} u^i(t)\|_\infty + cC(q^i(t)) \big), \tag{22}$$

where $i = 1, \ldots, N_s$ indexes the state, input, references, and query values related to the i-th simulation. The table also reports the average CPU time for solving Problem 12 on a Macbook 2.4GHz running Matlab 7.6 and Cplex 9.0.

As one can see from the results reported in Table 2, the proposed stochastic min-max policy achieves the best average performance, with an improvement of 20.4% with respect to the deterministic min-max policy, an additional 11.4% with respect to the stochastic min-max policy with random query, and a further 7.5% with respect to the stochastic min-max policy with heuristics-based query. Moreover, the computation times for all the policies are of the same order of magnitude for this particular application, which demonstrates the viability of the methodology from a computational viewpoint.

6 Conclusions

In this paper we proposed a stochastic programming approach to the problem of simultaneous optimal information gathering and decision making in an

uncertain environment. In particular, we dealt with linear optimization problems in which the feasibility set can be enlarged via a set of possible queries with stochastic outcomes. This class of problems was posed as a two-stage mixed integer stochastic optimization problems with endogenous uncertainty, that can be solved recursively in time for optimal performance of systems subject to uncertain constraints and uncertain references, where it is possible to reduce uncertainty bounds through queries. The proposed scheme minimizes the expected optimal cost with respect to the chosen query, while still guaranteeing robust constraint satisfaction. The results are demonstrated using a supply-chain example, which also shows the viability of the methodology from a computational viewpoint.

References

1. Bemporad, A., Borrelli, F., Morari, M.: Min-max control of constrained uncertain discrete-time linear systems. IEEE Transactions On Automatic Control 48(9), 1600–1606 (2003)
2. Bemporad, A., Giorgetti, N.: Logic-based methods for optimal control of hybrid systems. IEEE Transactions On Automatic Control 51(6), 963–976 (2006)
3. Bemporad, A., Muñoz de la Peña, D., Piazzesi, P.: Optimal control of investments for quality of supply improvement in electrical energy distribution networks. Automatica 42(8), 1331–1336 (2006)
4. Bertsekas, D.P., Tsitsiklis, J.T.: Neuro-Dynamic Programming. Athena Scientific, Belmont (1996)
5. Birge, J.R., Louveaux, F.V.: Introduction to Stochastic Programming. Springer, New York (1997)
6. Blei, D.M., Kaelbling, L.P.: Shortest paths in a dynamic uncertain environment. In: IJCAI Workshop on Adaptive Spatial Representations of Dynamic Environments (1999)
7. Chong, C.Y., Kumar, S.P.: Sensor networks: Evolution, opportunities, and challenges. Proceedings of the IEEE 91, 1247–1256 (2003)
8. Perea-López, E., Ydstie, B.E., Grossmann, I.E.: A model predictive control strategy for supply chain optimization. Computers and Chemical Engineering 27(8-9), 1201–1218 (2003)
9. Kall, P., Wallace, S.W.: Stochastic Programming. Wiley, Chichester (1994)
10. Brauna, M.W., Rivera, D.E., Floresa, M.E., Carlyleb, W.M., Kempf, K.G.: A Model Predictive Control framework for robust management of multi-product, multi-echelon demand networks. Annual Reviews in Control 27(2), 229–245 (2003)
11. Muñoz de la Peña, D., Alamo, T., Bemporad, A., Camacho, E.F.: A decomposition algorithm for feedback min-max model predictive control. IEEE Transactions On Automatic Control 51(10), 1688–1692 (2006)
12. Neumann, P.: Communication in industrial automation - what is going on? Control Engineering Practice 15, 1332–1347 (2007)
13. Puterman, M.L.: Markov Decision Processes. Wiley and Sons, Chichester (1994)
14. Ross, S.: Introduction to Stochastic Dynamic Programming. Academic Press, London (1983)
15. Sahinidis, N.V.: Optimization under uncertainty: State-of-the-art and opportunities. Computers & Chemical Engineering 28(6-7), 971–983 (2004)

16. Scokaert, P.O.M., Mayne, D.Q.: Min-max feedback model predictive control for constrained linear systems. IEEE Transactions On Automatic Control 43, 1136–1142 (1998)
17. Sutton, R.S., Barto, A.T.: Reinforcement Learning. MIT Press, Cambridge (1998)
18. Tatikonda, S., Mitter, S.: Control under communication constraints. IEEE Transactions on Automatic Control 49, 1056–1068 (2004)
19. Williams, H.P.: Model Building in Mathematical Programming, 3rd edn. John Wiley & Sons, Chichester (1993)

Hybrid Modelling, Power Management and Stabilization of Cognitive Radio Networks*

Alessandro Borri[1], Maria Domenica Di Benedetto[1], and Maria-Gabriella Di Benedetto[2]

[1] Department of Electrical and Information Engineering, Centre of Excellence DEWS, University of L'Aquila, Monteluco di Roio, 67040 L'Aquila (AQ), Italy
{borri,dibenede}@ing.univaq.it
[2] School of Engineering, University of Rome La Sapienza, Infocom Department, Via Eudossiana, 18, 00184 Rome, Italy
dibenedetto@newyork.ing.uniroma1.it

Abstract. In this paper, we deal with hybrid modelling, optimal control and stability in cognitive radio networks. Networks that are based on cognitive radio communications are intelligent wireless communication systems. They are conscious about changes in the environment and are able to react in order to achieve an optimal utilization of the radio resources. We provide a general hybrid model of a network of nodes operating under the cognitive radio paradigm. The model abstracts from the physical transmission parameters of the network and focuses on the operation of the control module. The control problem consists in minimizing the consumption of the network, in terms of average transmitted power or total energy spent by the whole network. A hybrid optimal control problem is solved and the power-optimal control law is computed. We introduce the notion of network configuration stability and derive a control law achieving the best compromise between stability and optimal power consumption. Finally, we apply our results to the case of a cognitive network based on UWB technology.

1 Introduction

In recent years, much interest has arisen in cognitive networks and their applications. The cognitive terminology was coined by Joseph Mitola III [1] and refers to radio devices that are able to sense the external environment, learn from history and make intelligent decisions in order to adjust their transmission parameters according to the current state of the environment [2]. The main features of cognitive networks have been mostly studied from the radio perspective (see, for example, [3], [4] and [5]). Some of the topics that have been investigated are spectrum management, cognitive architecture, power control, security issues. A successful approach is given by the game theory [6]. The power control problem, in wireless (not necessarily cognitive) contexts, has been addressed in

* This work has been partially supported by the Center of Excellence for Research DEWS, University of L'Aquila, 67040 Monteluco di Roio, L'Aquila (AQ), Italy.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 76–89, 2009.

many works, mainly as a noncooperative game [7], [8], in partially hybrid contexts [9] or in UWB networks with cognitive features [10].

In [13] and [14], we introduced hybrid modelling of self-organizing communication networks, and more specifically of overlay UWB networks. In this paper, we expand the above model by applying the concepts to any network of nodes operating under the cognitive radio paradigm. In particular, we provide a model that abstracts from the physical transmission parameters of the network and focuses on the operation of the control module. The control problem consists in minimizing the consumption of the network, in terms of average transmitted power or total energy spent by the whole network. For simplicity, the focus is on the uplink communications and the network topology is a star, that is, the control action is centralized in a Central Node (usually referred to as CNode). The CNode selects at each time t one out of several sets of transmission parameters that must be used by the other nodes for their transmissions. In particular, the selection is made on the basis of power minimization. We provide an optimal solution to the power minimization problem for a generally defined cognitive network. Based on the observation that the optimal solution lacks providing stability guarantees, we further refine the model by introducing an energetic cost that weighs the energy loss provoked by switching from one set of transmission parameters to another. We then derive the solution to the power minimization problem under stability constraints, and compare it to the original optimal solution.

The paper is organized as follows. In Section 2, we review some definitions of hybrid systems and provide a complete description of the cognitive network model. In Section 3, we introduce the energy minimization problem for the hybrid system and compute the hybrid power-optimal control strategy. Then, we introduce the notion of configuration stability, and use this concept to find a sub-optimal configuration-stabilizing solution of the optimum problem. Section 4 addresses the case study of a cognitive network based on UWB technology. Section 5 offers some concluding remarks.

2 Hybrid Modelling

2.1 Basic Definitions

We define the class of hybrid systems we consider in this paper, following the framework introduced by [11]. Our definition includes continuous control input, continuous disturbance and continuous output. Moreover, both discrete control inputs and discrete disturbances act on the system.

Definition 1 (Hybrid System). *A hybrid system $\mathcal{H}$ is a collection*

$$\mathcal{H} = (Q \times X, Q_0 \times X_0, U, D, Y, Inv, S, \Sigma, E, R) \tag{1}$$

where

- $Q \times X$ is the hybrid state space, where $Q \subset \mathbb{N}$ is a finite set of discrete states and X is the continuous state space. $Q_0 \times X_0 \subseteq Q \times X$ is the set of initial discrete and continuous conditions.
- U, D, Y are subsets of finite dimensional vector spaces and are respectively the continuous input, disturbance and output space. We denote by $\mathcal{U}_c$ the set of piecewise continuous control functions $u : \mathbb{R} \longrightarrow U$ and by $\mathcal{U}_d$ the set of disturbance functions $d : \mathbb{R} \longrightarrow D$.
- $Inv : Q \longrightarrow 2^X$ is a map associating to each discrete state $q \in Q$ a domain of acceptable continuous states.
- $S = \{S_q\}_{q \in Q}$ associates to each discrete state $q \in Q$ the nonlinear time-variant continuous system

$$S_q : \begin{cases} \dot{x}(t) = f_q(t, x(t), u(t)) \\ y(t) = h_q(t, x(t), u(t), d(t)) \end{cases}$$

  where $t \in \mathbb{R}$, $x(t) \in X$, $u(t) \in U$, $d(t) \in D$. Given $q \in Q$, $f_q(\cdot)$ is a function such that, $\forall u(\cdot) \in \mathcal{U}_c$, the solution $x(t)$ exists and is unique for all $t \in \mathbb{R}$. Given $q \in Q$, $t \in \mathbb{R}$, $x(t) \in X$, $u(t) \in U$, $d(t) \in D$, $y(t) = h_q(t, x(t), u(t), d(t)) \in Y$, where $h_q : T \times X \times U \times D \longrightarrow Y$.
- Σ is the finite set of discrete inputs, collecting discrete control inputs and discrete disturbances. Each input is associated to one or more edges $e \in E$.
- $E = E_c \cup E_d \subset Q \times \Sigma \times Q$ is a collection of edges, including the set of the controlled transitions E_c, determined by discrete control inputs, and the set of the switching transitions E_d, determined by discrete disturbances. We assign higher priority to switching transitions with respect to controlled ones: if a switching transition and a controlled transition occur at the same time, only the switching one is considered, while the controlled one is ignored.
- $R : E \times X \to X$ is a deterministic map called *reset*.

Definition 2 (Execution). *An execution of the hybrid system $\mathcal{H}$ is a collection $\chi = (\tau, q, \sigma, x, y, u, d)$, consisting of a set of switching times $\tau = \{t_i\}_{i=0}^{L}$ and the functions $q(\cdot) : [t_0, t_L) \longrightarrow Q$, $\sigma(\cdot) : [t_0, t_L) \longrightarrow \Sigma$, $x(\cdot) : [t_0, t_L) \longrightarrow X$, $y(\cdot) : [t_0, t_L) \longrightarrow Y$, $u(\cdot) : [t_0, t_L) \longrightarrow U$, $d(\cdot) : [t_0, t_L) \longrightarrow D$, satisfying the following conditions:*

- *Initial condition: $(q(t_0), x(t_0)) \in Q_0 \times X_0$.*
- *Discrete evolution: for all $i = 1, ..., L-1$*
  1. *$q(\cdot)$ and $\sigma(\cdot)$ are constant over the intervals $[t_i, t_{i+1})$;*
  2. *$(q^-(t_i), \sigma(t_i), q(t_i)) \in E$;*
  3. *$x(t_i) = R(q^-(t_i), \sigma(t_i), q(t_i), x^-(t_i))$ where $q^-(t_i) = \lim_{t \longrightarrow t_i^-} q(t)$ and $x^-(t_i) = \lim_{t \longrightarrow t_i^-} x(t)$.*
- *Continuous evolution: for all $i = 1, ..., L-1$, at time $t \in [t_i, t_{i+1})$*
  1. *$x(t)$ is the (unique) state trajectory of the dynamical system $S_{q(t_i)}$ with initial time t_i, initial state $x(t_i)$ and control law u;*
  2. *$x(t) \in Inv(q(t_i))$;*
  3. *$y(t) = h_{q(t_i)}(t, x(t), u(t), d(t))$.*

2.2 Hybrid Modelling of Cognitive Networks

We model the set of wireless nodes as a social network, forming one single entity [13]. We consider a self-organizing network of nodes that adopt a multiple access scheme in which coexistence is foreseen, that is signals originating from different users share in principle a same resource in terms of time and frequency. Users separation is obtained by appropriate coding.

An important hypothesis, that is fundamental in our model, is the possibility of selecting among different sets of transmission parameters, which can range from coding, to modulation formats, to pulse shaping. We therefore assume W different configurations, that is W different sets of transmission parameters w_q, with $q = 1, ..., W$. We associate an energetic cost $c \geq 0$ to the operation of switching from one set to another.

We assume that system performance is described by a specification on the system behavior, for example the level of signal to noise ratio or the transmission delay.

The topology of the network is a star, that is, nodes communicate through the CNode, and implement the cognitive radio paradigm. If a new node asks for admission, the CNode evaluates the possibility of admitting it, by checking whether constraints for admission are compatible with network specifications.

At each time t, the CNode communicates to the other nodes the set of transmission parameters $q \in \{1, ..., W\}$, the number of active nodes N and the average power level P_{RX} that it wants to receive. We suppose the signal containing the above information is sent by the controller at a fixed power level that is predetermined and known by all nodes. Each active node j receives this signal and, on the basis of received power level, can estimate the attenuation $a_j(t)$ characterizing its path to the coordinator and can determine the power to be used in its transmissions, namely $p_{TX,j}(t) = a_j(t)\, p_{RX}(t)$. In this work, we disregard w.l.o.g. any assumption about the maximum transmission power of each node [13], in order to decouple completely the power-minimization problem (object of the present paper) and the problem of nodes leaving the network for lack of available power.

Since a node can enter/leave the network several times, one has $j \in \{1, ..., N_{\max}\}$, where $N_{\max}$ is the maximum number of nodes which can be admitted to the network. We define a time-varying attenuation vector $A(t) \in \mathbb{R}^{N_{\max}}$, that includes the attenuations $a_j(t)$ for each node j, and an activity vector $S(t) \in \mathbb{R}^{N_{\max}}$, whose generic element $s_j(t)$ equals 1 if node j is transmitting at time t, and 0 otherwise.

The instantaneous transmission power consumption of the network can expressed as:

$$P_{TX}(t) = \sum_{j=1}^{N_{\max}} s_j(t)\, p_{TX,j}(t) = \left(\sum_{j=1}^{N_{\max}} s_j(t)\, a_j(t) \right) p_{RX}(t) = A'(t) S(t)\, u(t).$$

Following the assumptions, the network can be modeled as a hybrid system as follows:

- The set of discrete states is $Q = \{1, 2, ..., W\}$. Each discrete state $q \in Q$ is associated to a configuration w_q, that is a set of transmission parameters that are used for communication. The continuous state $x \in X = \mathbb{R}^2$ represents the number N of active nodes and the energy spent by the network from the beginning of its life, which includes the energy spent for transmission and for switching among different configurations but does not include the energy spent by nodes to stay in state of idle/receiving.
- The set of initial states is $Q_0 \times X_0 = Q \times \{n \in \mathbb{N},\ n \geq 2\} \times \{0\}$: the network begins its life when there are at least 2 nodes, the minimum for a communication. At the beginning, the energy consumption is zero.
- The domains are
$$Inv\,(q) = \mathbb{N} \times \mathbb{R}^+ \quad \forall q \in Q.$$
- The continuous dynamics associated to a discrete state $q \in Q$ is
$$S_q : \begin{cases} \begin{bmatrix} \dot{x}_1\,(t) \\ \dot{x}_2\,(t) \end{bmatrix} = f\,(t, u\,(t)) = \begin{bmatrix} 0 \\ A^{'}\,(t) S\,(t)\,u\,(t) \end{bmatrix} \\ y\,(t) = \begin{bmatrix} h_q\,(x\,(t), u\,(t)) \\ d\,(t) \end{bmatrix} \end{cases}$$
$f\,(t, u)$ includes a trivial dynamics (the number of nodes may change only as a consequence of a discrete transition) and the instantaneous transmission power consumption of the network $A'\,(t)\,S\,(t)\,u$. The continuous control input $u\,(t) \in U = \mathbb{R}^+$ represents the power level $p_{RX}\,(t)$ that the CNode wants to receive from each transmitting node. The output vector $y\,(t) \in Y = Q \times \mathbb{N} \times U \times D$ includes the set of variables sent to each node by the CNode
$$h_q\,(x, u) = \begin{bmatrix} q\ x_1\ u \end{bmatrix}^T$$
and the measurable continuous disturbance vector $d\,(t) \in D \subseteq \mathbb{R}^W$ ($d\,(t)$ is not sent to all active nodes).
- The discrete inputs are $\Sigma = \Sigma_c \cup \{\sigma_d\}$, where σ_d is the discrete disturbance representing the uncontrollable event that a node leaves the network, while $\Sigma_c = \{\sigma_q,\ q \in Q\} \cup \{\sigma_a\}$ is the set of discrete controls:
  - σ_q is the control action occurring when the coordinator decides to commute from the current set of parameters to the set w_q, $q \in Q$;
  - σ_a models the decision to accept a new candidate node in the network; we assume here that the decision procedure, after an admission request, requires a negligible time to be performed.
- The edges are $E = E_c \cup E_d$, where E_c is the set of the *controlled transitions*:
$$E_c = E_{c,W} \cup E_{c,a}$$
$$E_{c,W} = \{(p, \sigma_q, q)\,,\ p,\ q \in Q,\ p \neq q,\ \sigma_q \in \Sigma_c\}$$
$$E_{c,a} = \{(q, \sigma_a, q)\ \ ,\ q \in Q,\ \sigma_a \in \Sigma_c\}$$
and $E_d = \{(q, \sigma_d, q)\ \ ,\ q \in Q\}$ is the set of the *switching transitions*.

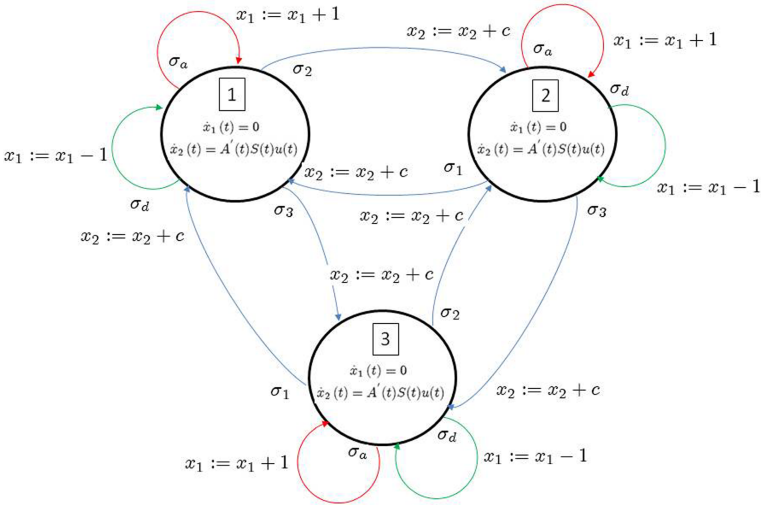


Fig. 1. Hybrid Model

– Reset map: $\forall x \in X$

$$R(e,x) = \begin{cases} \begin{bmatrix} x_1 \\ x_2 + c \end{bmatrix} & e \in E_{c,W} \\ \begin{bmatrix} x_1 + 1 \\ x_2 \end{bmatrix} & e \in E_{c,a} \\ \begin{bmatrix} x_1 - 1 \\ x_2 \end{bmatrix} & e \in E_d \end{cases}$$

Note that the x_2 dynamics is reset only when a change in the transmission parameters occurs, modelling the energetic switching cost c for the network.

The network model is a non-deterministic hybrid system because of the presence of the discrete disturbance σ_d and of the continuous disturbance $d(t)$.

3 Energy Minimization as a Hybrid Optimal Control Problem

Since the admission of a candidate node is allowed only if constraints for admission are compatible with network specifications and none of the current active nodes is forced to leave the network as a result of its admission [13], we focus here on the choice of a discrete control σ_q (a set of transmission parameters) and of a continuous control $u(t)$ that minimize the energy consumption of the network. In this section, we define the energy minimization problem on the hybrid system representing the cognitive network and solve it by defining the discrete and continuous control action.

3.1 Energy Minimization

In the following definitions, we refer to the hybrid system (1). Let $\chi_{T,(q_0,x_0)}$ denote the set of all executions $\chi = (\tau, q, \sigma, x, y, u, d)$ defined on the same time horizon $T = [t_0, t_L) \subset \mathbb{R}$ with initial condition $(q_0, x_0) \in Q_0 \times X_0$, i.e. $(q(t_0), x(t_0)) = (q_0, x_0)$. Given an execution $\chi \in \chi_{T,(q_0,x_0)}$, we define its *value* at time t, and we abuse notation by writing $\chi(t)$, as

$$\chi(t) := (q(t), \sigma(t), x(t), y(t), u(t), d(t))$$

We partition the set of the switching times $\tau = \tau_c \cup \tau_{nc}$, where $\tau_c := \{t_{c,i}\}_{i=1}^{L_c}$ includes all the switching times due to configuration transitions (elements of $E_{c,W}$) and τ_{nc} is its complement $\tau \setminus \tau_c$. Hence τ_{nc} includes switching times due to switching transitions and to admission transitions.

*Problem 1 (**Energy minimization**).* Given the hybrid system $\mathcal{H}$, a set $\chi_{T,(q_0,x_0)}$ including all the executions $\chi = (\tau, q, \sigma, x, y, u, d)$ defined on a time horizon $T = [t_0, t_{fin}) \subset \mathbb{R}$ with $(q_0, x_0) \in Q_0 \times X_0$, where $d \in \mathcal{U}_d$ is a given disturbance function and $\tau \supseteq \tau_{nc}$, where τ_{nc} is given. Let Ξ be the space of all discrete strategies $\sigma : T \longrightarrow \Sigma$ compatible with τ_{nc}, i.e. such that $\sigma(t_{nc,i}) \in \{\sigma_a, \sigma_d\}$ $\forall t_{nc,i} \in \tau_{nc}$, and $\mathcal{U}_{bound}$ the space of all continuous functions u that satisfy a constraint $u(t) \geq u_{LB}(q, t)$ for all $t \in T$. The *energy minimization control problem* consists in minimizing the functional

$$J_e(u, \sigma, \tau_c, c) = \int_T A'(t) S(t) u(t) dt + c * card(\tau_c) \quad = x_2(t_{fin})$$

over all $\sigma \in \Xi$ and $u \in \mathcal{U}_{bound}$.

Solution 1 (case $c = 0$). We solve the energy minimization problem for the case $c = 0$ and refer to the corresponding optimal strategy as (σ_0^*, u_0^*). With this assumption, we can also write

$$\min_{(\sigma,u)} x_2(t_{fin}) = \min_{(\sigma,u)} J_e(u, \sigma, \tau_c, 0) = \min_{(\sigma,u)} \int_T A'(t) S(t) u(t) dt$$

so the discrete optimal trajectory and the continuous optimal control action are clearly

$$q_0^*(t) := \arg\min_{q \in Q} u_{LB}(q, t)$$

$$u_0^*(t) = u_{LB}(q_0^*(t), t) \quad \forall t \in T$$

The function $q_0^* : T \longrightarrow Q$ "induces" the set of switching times $\tau_0^* = \tau_c^* \cup \tau_{nc}$, where τ_c^* is the set of controlled switching times with cardinality L_c^*, namely such that for all $i = 1, ..., L_c^*$, $q_0^*(t_{c,i}^*) \neq \lim_{t \longrightarrow (t_{c,i}^*)^-} (q_0^*(t))$. Finally the discrete control function $\sigma_0^* : T \longrightarrow \Sigma$ is a piecewise constant function, in which the control-dependent part is defined by the relation

$$\sigma_0^*(t_{c,i}^*) = \sigma_{q_0^*(t_{c,i}^*)} \in \Sigma_c \qquad i = 1, ..., L_c^*$$

Remark 1. For $c = 0$, the solution to the energy minimization problem (σ_0^*, u_0^*) is obtained by minimizing the power at each time t. Hence, it is computable in real-time and the control (σ_0^*, u_0^*) is indeed achievable. We refer to it as *hybrid power-optimal strategy*.

For $c > 0$, the solution (σ^*, u^*) to the energy minimization problem depends on the values of the time-varying constraint $u_{LB}(q,t)$ over the whole interval $[t_0, t_{fin})$, which may not be known a priori. For example, if the constraint corresponds to a minimum signal-to-noise ratio requirement, $u_{LB}(q,t)$ would depend on disturbances such as external noise and interference. Those disturbances are supposed to be measurable in real-time but cannot be known a priori. Hence, in general, the control strategy (σ^*, u^*) is not computable in real-time and is therefore not achievable. The power-optimal strategy (σ_0^*, u_0^*) is not optimal for $c > 0$, but is a sub-optimal solution of the energy minimization problem as it will be precisely described in the next subsection, where we look for a strategy achieving the best compromise between computability, power consumption and stability of the configuration of the network.

Remark 2. Note that since the dynamic equation in each discrete state and the reset maps are deterministic, an optimal strategy (σ^*, u^*) leads to a unique maximal hybrid execution $(\tau^*, \sigma^*, q^*, x^*, y^*, u^*, d) \in \chi_{T,(q_0,x_0)}$. The discrete optimal trajectory q^* is well-defined if there are no multiple optimal configurations at the same time. If this situation occurs, the optimal configuration can be either chosen arbitrarily among the optimal ones, or chosen according to additional constraints or specifications.

3.2 Configuration Stability

In the previous subsection, we showed that the hybrid strategy (σ_0^*, u_0^*) minimizes both power and energy consumption if the cost of configuration switchings is negligible. However, the discrete control strategy σ_0^* does not assure stability of the network, in the sense that too many switchings may occur in a finite amount of time. Switchings may also be due to switching transitions or admission requests, but since those are uncontrollable transitions, we focus here on ensuring stability at least from the controlled switchings point of view. In this subsection, we propose a sub-optimal hybrid strategy $(\sigma_\delta^*, u_\delta^*)$ that guarentees good performance and stable behavior.

The sub-optimal continuous control u corresponding to any sub-optimal discrete state $\tilde{q}(t) \neq q^*(t)$ is $\tilde{u}(t) := u_{LB}(\tilde{q}(t), t)$, that is the lowest value of continuous control satisfying the constraint. This simple consideration allows us, in the following, to focus only on the discrete control of the system.

Definition 3 (Configuration stability). *Let $\mathcal{H}_c$ be the hybrid system $\mathcal{H}$ controlled by a hybrid strategy (σ, u). If there exists $\delta > 0$ such that, for any execution $(\tau, q, \sigma, x, y, u, d)$, the inequality $0 < \delta \leq t_{c,i+1} - t_{c,i}$ holds $\forall i \in \{1, ..., L_c - 1\}$, then $\mathcal{H}_c$ is said to be δ-configuration stable and σ is said to be a* δ-configuration stabilizing strategy.

Configuration stability is related to the existence of a dwell-time, but with reference only to controlled switching times. The system $\mathcal{H}_c$ controlled by the power-optimal control strategy (σ_0^*, u_0^*) is not necessarily δ-configuration stable, for some $\delta > 0$, since consecutive configuration switchings can be arbitrarily close in time. We propose here to modify the optimal strategy in order to achieve configuration stability.

In the space Ξ of all discrete strategies $\sigma : T \longrightarrow \Sigma$ compatible with τ_{nc}, we consider a pseudometric $d : \Xi \times \Xi \longrightarrow \mathbb{R}^+$:

$$d(x, y) := \lambda(\{t \in T : x(t) - y(t) \neq 0\}) \quad \forall x, y \in \Xi$$

where $\lambda(\cdot)$ is the Lebesgue measure. The chosen pseudometric compares how different two discrete-valued functions are in terms of the duration of the time intervals where they assume different values. In the following, we propose a strategy σ_δ^* that is achieved by deferring any controlled switching so that it occurs not before a time δ from the previous one. If one or more than a switching occur within a time δ, only the last one is taken into account.

Algorithm. Given the collection of optimal switching times $\tau_c^* = \left\{t_{c,i}^*\right\}_{i=1}^{L_c^*}$, we build a sequence $\tilde{\tau}_c = \left\{\tilde{t}_{c,j}\right\}_{j=1}^{\tilde{L}_c}$, depending on τ_c^*, as follows:

Initialization: $\tilde{t}_{c,1} = t_{c,1}^*$; flag=FALSE; $i = 2$; $k = 2$.

Iteration: while $(i \leq L_c^*)$
{ if $(t_{c,i}^* < \tilde{t}_{c,k-1} + \delta)$ then {flag=TRUE; $i++$;}
else {if (flag==TRUE) then {$\tilde{t}_{c,k} = \tilde{t}_{c,k-1} + \delta$; flag=FALSE;}
else {$\tilde{t}_{c,k} = t_{c,i}^*$; $i++$;}
$k++$;}
}

Conclusion: if ((flag==TRUE) and $(\tilde{t}_{c,k-1} + \delta < t_{fin})$) then $\tilde{t}_{c,k} = \tilde{t}_{c,k-1} + \delta$;
else $k--$;
$\tilde{L}_c = k$;
define the collection $\tilde{\tau}_c = \left\{\tilde{t}_{c,j}\right\}_{j=1}^{\tilde{L}_c}$;
set $\sigma_\delta^*\left(\tilde{t}_{c,j}\right) = \sigma_{q^*\left(\tilde{t}_{c,j}\right)}$ for all $j = 1, ..., \tilde{L}_c$.

Theorem 1. *Given a hybrid system $\mathcal{H}$, a time horizon $T = [t_0, t_f) \subseteq \mathbb{R}$, an initial condition $(q_0, x_0) \in Q_0 \times X_0$ and the pseudometric space Ξ of all discrete strategies $\sigma : T \longrightarrow \Sigma$ compatible with τ_{nc}. If $\sigma_0^* \in \Xi$ is the power-optimal discrete control strategy, the sub-optimal strategy σ_δ^*, with controlled switching times $\tilde{\tau}_c$, has the following properties:*

1. *σ_δ^* is a δ-configuration stabilizing strategy;*
2. *σ_δ^* does not anticipate σ_0^* (causality principle), i.e. "jumps" in σ_δ^* cannot occur earlier than corresponding "jumps" in σ_0^*;*
3. *if $\tilde{\sigma}$ is any other δ-configuration stabilizing strategy, then $d(\sigma_\delta^*, \sigma_0^*) < d(\tilde{\sigma}, \sigma_0^*)$, i.e. σ_δ^* is at minimum distance from the power-optimal strategy σ_0^*.*

Proof. The algorithm builds the sequence $\tilde{\tau}_c$ such that $0 < \delta \leq \tilde{t}_{c,j+1} - \tilde{t}_j$ $\forall j \in \left\{0, 1, ..., \tilde{L}_c - 1\right\}$. Hence, property 1 is fulfilled by construction. Moreover, the final step of the algorithm shows that σ_δ^* is built causally starting from σ_0^*, so that property 2 holds. Property 3 is also satisfied because it is not possible to find another discrete strategy $\tilde{\sigma}$, not anticipating σ_0^*, such that $d(\tilde{\sigma}, \sigma_0^*) < d(\sigma_\delta^*, \sigma_0^*)$; in fact σ_δ^* equals σ_0^* except for time intervals in which the dwell-time constraint is not fulfilled. In such cases, it guarantees the controlled system to have exactly a dwell time equal to δ. Any other function $\tilde{\sigma}$ such that $d(\tilde{\sigma}, \sigma_0^*) < d(\sigma_\delta^*, \sigma_0^*)$ equals σ_0^* in at least one of the time intervals in which the dwell-time constraint is not fulfilled, so it itself cannot fulfill the constraint. So we can finally deduce that, given the optimal strategy σ_0^*, the strategy σ_δ^* is the "nearest" function satisfying the dwell-time constraint.

Call $T_\delta \subset T$ the subset of the time horizon in which σ_δ^* and σ_0^* are different. Notice that $\lambda(T_\delta) \leq L_c^* \ \delta$. Then consider the transmission energy consumption $J_e(u, \sigma, \tau_c, 0)$ already defined. The δ-configuration stabilizing strategy σ_δ^* uniquely defines the discrete evolution q_δ^* and the continuous sub-optimal control u_δ^*

$$\sigma_\delta^*\left(\tilde{t}_{c,j}\right) = \sigma_{q_\delta^*\left(\tilde{t}_{c,j}\right)} \in \Sigma_c \qquad i = 1, ..., \tilde{L}_c$$

$$u_\delta^*(t) = u_{LB}\left(q_\delta^*(t), t\right).$$

The energy loss between power-optimal and sub-optimal strategy is

$$\begin{aligned} \Delta J_\delta &:= J_e(u_\delta^*, \sigma_\delta^*, \tilde{\tau}_c, 0) - J_e(u_0^*, \sigma_0^*, \tau_c^*, 0) = \\ &= \int_{T_\delta} A'(t) S(t) \left[u_{LB}\left(q_\delta^*(t), t\right) - u_{LB}\left(q_0^*(t), t\right)\right] dt. \end{aligned}$$

Notice that the integrand is not null in T_δ. Moreover the integration domain is continuous with respect to the variable δ, and $\lim_{\delta \longrightarrow 0} \lambda(T_\delta) = 0$. Hence we can conclude that

Theorem 2. *For any $\varepsilon > 0$, there exists $\delta := \delta(\varepsilon)$ and a δ-configuration stabilizing strategy σ_δ^* such that $\Delta J_\delta := J_e(u_\delta^*, \sigma_\delta^*, \tilde{\tau}_c, 0) - J_e(u_0^*, \sigma_0^*, \tau_c^*, 0) < \varepsilon$.*

The previous theorem is a continuity result stating that any energy requirement can be approached with as much precision as desired, by tuning the duration of the dwell-time δ.

Remark 3. Since the functional $J_e(u, \sigma, \tau_c, c)$ is unbounded if the switching cost c grows, then $\forall \delta > 0$, $\exists \bar{c} > 0$ such that for all costs $c \geq \bar{c}$, the sub-optimal control strategy $(\sigma_\delta^*, u_\delta^*)$, with a lower number of switchings than the power-optimal one (σ_0^*, u_0^*), results to be even better than (σ_0^*, u_0^*) in terms of energy consumption:

$$J_e(u_\delta^*, \sigma_\delta^*, \tilde{\tau}_c, c) \leq J_e(u_0^*, \sigma_0^*, \tau_c^*, c) \quad \forall c \geq \bar{c}.$$

4 Case Study: Cognitive Networks Based on UWB Technology

Refer, for example, to [10], [13], [14] for the assumptions about UWB Communication.

We consider as set of transmission parameters a set of W waveforms that are used for the pulse shaping. The system specification is expressed in terms of signal-to-noise ratio on a pulse at the correlator output, that has the following expression

$$SNR_p\left(u,q,d,N\right)=\frac{T_S\ u}{d_q+\sigma_m^2\left(q\right)\left(N-1\right)u}$$

where T_S is the chip duration, N is the number of nodes, $\sigma_m^2\left(q\right)$ is the MUI weight for the waveform q and d_q is the external noise power for the waveform q, i.e. the $q-th$ component of the disturbance vector d.

The specification is $SNR_p\left(u,q,d,N\right)\geq SNR_0$ where $SNR_0>0$ is given. It leads to the lower-bound constraint on the minimum received power

$$u\left(t\right)\geq u_{LB}\left(q\left(t\right),t\right)=\frac{SNR_0 d_{q(t)}\left(t\right)}{T_S-SNR_0\sigma_m^2\left(q\left(t\right)\right)\left(N\left(t\right)-1\right)}\qquad\forall t\in T$$

Notice that such a lower bound, corresponding to the minimum signal-to-noise ratio requirement, increases with external noise power and MUI weight. The power-optimal solution is given by the following expressions

$$\begin{cases} q_0^*\left(t\right)=\arg\min\limits_{q\in Q}\left(\frac{SNR_0 d_q(t)}{T_S-SNR_0\sigma_m^2(q)(N(t)-1)}\right) \\ u_0^*\left(t\right)=\frac{SNR_0 d_{q_0^*(t)}(t)}{T_S-SNR_0\sigma_m^2\left(q_0^*(t)\right)(N(t)-1)} \\ \sigma_0^*\left(t_{c,i}^*\right)=\sigma_{q_0^*\left(t_{c,i}^*\right)} \qquad i=1,...,L_c^* \end{cases} \qquad \forall t\in T$$

where τ_c^* is the set of the "induced" controlled switching times, with cardinality L_c^* (see subsection 3.1). Note that the power-optimal hybrid strategy (σ_0^*,u_0^*) can be regarded as an output-feedback hybrid optimal control law.

4.1 Simulations

In order to evaluate the impact of cognition, the cognitive UWB network coexists with several narrowband interferers. The CNode is located at the centre of a circular area with radius $R=10\ m$. The area contains $N=10$ active nodes, not changing during the simulation time. The active users are continuously transmitting data towards the CNode during the whole duration of the simulation. At time t, the N active nodes adopt a generic waveform $w_q(t)$, that can be selected within a set of 6 different waveforms $w_1(t)$, . . . , $w_6(t)$, represented by the first six odd derivatives of the Gaussian pulse. Specifically, we assume that the CNode may order a change in the adopted waveform only at multiples of a given interval.

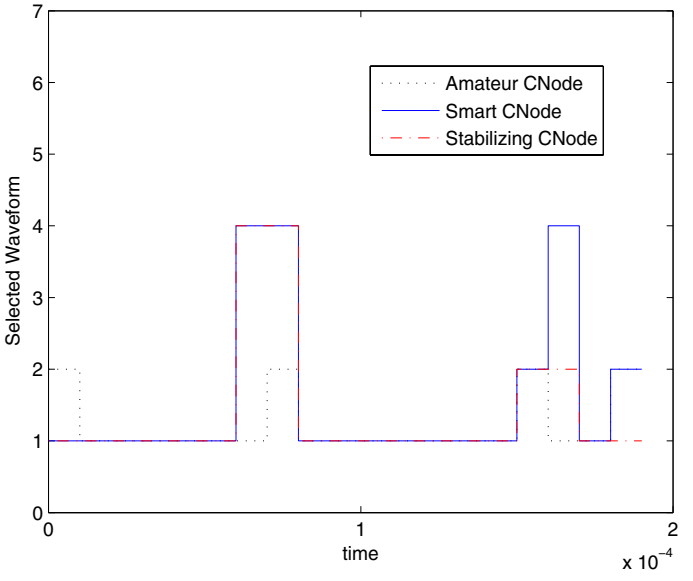


Fig. 2. Waveform Selection

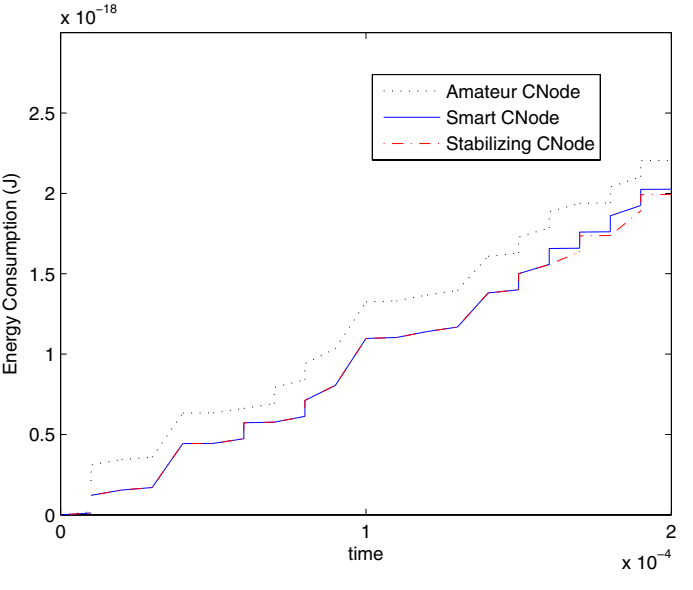


Fig. 3. Power Consumption

The synchronization threshold SNR0 is set to 3 dB, the emitted power of the interfering devices is equal to 10^{-3} W, the simulation is over 20 cognitive intervals, each of 10^{-5} seconds. The minimum dwell-time is set to 2 cognitive intervals and the switching cost is 10^{-19} $Joule$.

We assume that several narrowband interferers are present in the area, not transmitting continuously. In order to highlight the effect of cognition on network coexistence, we consider, for each simulation, four different CNodes that will be compared in terms of performance:

1. The Adaptive CNode (no cognition) initially selects a waveform and does not perform any further selection of the pulse shape during network lifetime.
2. The Amateur CNode (limited cognition) is capable to select the waveform in correspondence of a sub-set of the 6 available waveforms, consisting of the last used waveform and the two adjacent ones. Within this subset, it is capable of selecting the pulse shape minimizing the transmission power.
3. The Smart CNode (optimal cognition) is always capable of selecting the pulse shape that minimizes the transmission power for the active nodes of the network, i.e. it performs the power-optimal hybrid strategy (σ_0^*, u_0^*).
4. The Stabilizing CNode (stabilizing or sub-optimal cognition) guarantees the best trade-off between optimality and stability, performing the sub-optimal control strategy $(\sigma_\delta^*, u_\delta^*)$.

Figures 2 and 3 show the simulation results. The Adaptive CNode (not reported on the plot) chooses the waveform $w_2(t)$ and its network keeps transmitting using this pulse-shaper over the whole simulation. The result is totally unefficient because this choice leads to a power consumption that is about 7 times higher than in the optimal cognitive case. Limited and stabilizing cognitions are much better, requiring between 15% and 20% more power than the optimal CNode. The plots concerning power are not reported due to limited space.

Figure 2 shows that the stabilizing CNode makes only 4 waveform transitions, while the Amateur and the Smart one perform 6 jumps. This leads to a large saving of energy for the stabilizing CNode, such that its final energy consumption $x_2(t_{fin})$ is 2.7 % lower than the one performed by the power-optimal strategy (see Figure 3). This is an example in which Remark (3) holds, and the stabilizing strategy $(\sigma_\delta^*, u_\delta^*)$ is better than the power-optimal one (σ_0^*, u_0^*) in terms of total energy consumption.

5 Conclusions and Open Issues

In this paper, we focused on the hybrid modelling and optimal control of cognitive radio networks. At first, we provided a model of a general network of nodes operating under the cognitive radio paradigm, which abstracts from the physical transmission parameters of the network and focuses on the operation of the control module. Then, we proposed an optimal solution to the power minimization problem. We also introduced the notion of configuration stability and showed that this property is not ensured when the power-optimal control is applied. A control strategy achieving the best compromise between stability and optimality was then derived. Finally, we applied our results to the case of a cognitive network based on UWB technology.

The architecture we analyzed in this paper was centered on the Cognitive Node. An extension of this architecture would be one of multiple CNodes, with each CNode that is responsible for a cluster of nodes. The hierarchical distributed case will be the object of future investigations.

Acknowledgements

The authors would like to thank Daniele Domenicali for interesting discussions on cognitive radio networks and for the use of his UWB Simulator.

References

1. Mitola, J., Maguire, G.: Cognitive radio: making software radios more personal. IEEE Personal Communications 6(4), 13–18 (1999)
2. Mahmoud, Q.: Cognitive Networks: Towards Self-Aware Networks. Hardcover, John Wiley & Sons Inc. (2007)
3. Haykin, S.: Cognitive radio: brain-empowered wireless communications. IEEE Journal on Selected Areas in Communications 23(2), 201–220 (2005)
4. Thomas, R.W.: Cognitive Networks. PhD thesis, Virginia Polytechnic Institute and State University, Blacksburg, VA (September 2006)
5. Neel, J.O.: Analysis and Design of Cognitive Radio Networks and Distributed Radio Resource Management Algorithms. PhD thesis, Virginia Polytechnic Institute and State University, Blacksburg, VA (June 2007)
6. Maheswaran, R.T., Basar, T.: Nash equilibrium and decentralized negotiation in auctioning divisible resources. Group Decision and Negotiation 12, 361–395 (2003)
7. Famolari, D., Mandayam, N., Goodman, D., Shah, V.: A New Framework for Power Control in Wireless Data Networks: Games, Utility and Pricing. In: Ganesh, R., Pahlavan, K., Zvonar, Z. (eds.) Wireless Multimedia Network Technologies, ch. 1, pp. 289–310. Kluwer Academic Publishers, Dordrecht (1999)
8. Alpcan, T., Basar, T., Srikant, R., Altman, E.: CDMA uplink power control as a noncooperative game. Wireless Networks 8, 659–669 (2002)
9. Alpcan, T., Basar, T.: A hybrid systems model for power control in multicell wireless data networks. Performance Evaluation 57, 477–495 (2004)
10. Domenicali, D., De Nardis, L., Di Benedetto, M.-G.: UWB Network Coexistence and Coordination: a Cognitive Approach. In: Annual GTTI Meeting, June 18-20, Rome, Italy (2007)
11. Lygeros, J.: Lecture Notes on Hybrid Systems. Patras, Greece, Department of Electrical and Computer Engineering, University of Patras (2004)
12. Sontag, E.D.: Mathematical Control Theory: Deterministic Finite Dimensional Systems, 2nd edn. Springer, Heidelberg (1998)
13. Di Benedetto, M.-G., Di Benedetto, M.D., Giancola, G., De Santis, E.: Analysis of Cognitive Radio Dynamics. In: Bhargava, V., Hossain, E. (eds.) Cognitive Wireless Communications Networks. Springer, Heidelberg (2007)
14. Di Benedetto, M.-G., Giancola, G., Di Benedetto, M.D.: Introducing consciousness in UWB networks by Hybrid Modelling of Admission Control. ACM/Springer Journal on Mobile Networks and Applications, Special Issue on Ultra Wide Band for Sensor Networks 11(4), 521–534 (2006)
15. Bellman, R.: Dynamic Programming. Princeton University Press, Princeton (1957); Dover paperback edn. (2003)

Automatic Synthesis of Robust and Optimal Controllers – An Industrial Case Study*

Franck Cassez[1,**], Jan J. Jessen[2], Kim G. Larsen[2], Jean-François Raskin[3], and Pierre-Alain Reynier[4]

[1] National ICT Australia & CNRS, Sydney, Australia
[2] CISS, CS, Aalborg University, Denmark
[3] CS Department, Université Libre de Bruxelles, Belgium
[4] LIF, Aix-Marseille Universités & CNRS, UMR 6166, France

Abstract. In this paper, we show how to apply recent tools for the automatic synthesis of robust and near-optimal controllers for a real industrial case study. We show how to use three different classes of models and their supporting existing tools, UPPAAL-TIGA for synthesis, PHAVER for verification, and SIMULINK for simulation, in a complementary way. We believe that this case study shows that our tools have reached a level of maturity that allows us to tackle interesting and relevant industrial control problems.

1 Introduction

The design of controllers for embedded systems is a difficult engineering task. Controllers have to enforce properties like safety properties (e.g. "nothing bad will happen"), or reachability properties (e.g. "something good will happen"), and ideally they should do that in an efficient way, e.g. consume the least possible amount of energy. In this paper, we show how to use (in a systematic way) models and a chain of automatic tools for the synthesis, verification and simulation of a provably correct and near optimal controller for a real industrial equipment. This case study was provided to us by the HYDAC company in the context of a European research project Quasimodo*.

The system to be controlled is depicted in Fig. 1 and is composed of: (1) a machine which consumes oil, (2) a reservoir containing oil, (3) an accumulator containing oil and a fixed amount of gas in order to put the oil under pressure, and (4) a pump. When the system is operating, the machine consumes oil under pressure out of the accumulator. The level of the oil, and so the pressure within the accumulator (the amount of gas being constant), can be controlled using the pump to introduce additional oil in the accumulator (increasing the gas pressure). The control objective is twofold: first the

* Work supported by the projects: (i) Quasimodo: "Quantitative System Properties in Model-Driven-Design of Embedded", `http://www.quasimodo.aau.dk/`, (ii) Gasics: "Games for Analysis and Synthesis of Interactive Computational Systems", `http://www.ulb.ac.be/di/gasics/`, and (iii) Moves: "Fundamental Issues in Modelling, Verification and Evolution of Software", `http://moves.vub.ac.be`.

** This author is supported by a Marie Curie International Outgoing Fellowship within the 7th European Community Framework Programme.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 90–104, 2009.

level of oil into the accumulator (and so the gas pressure) can be controlled using the pump and must be maintained into a safe interval; second the controller should try to minimize the level of oil such that the accumulated energy in the system is kept minimal.

In a recent work [7], we have presented an approach for the synthesis of a correct controller for a timed system. It was based on the tool UPPAAL-TIGA [1] applied on a very abstract untimed game model for synthesis and on SIMULINK [8] for simulation. To solve the HYDAC control problem, we use three complementary tools for three different purposes: UPPAAL-TIGA for synthesis, PHAVER [5,4] for verification, and SIMULINK for simulation. For the synthesis phase, we show how to construct a (game) model of the case study which has the following properties:

- it is simple enough to be solved automatically using algorithmic methods implemented into UPPAAL-TIGA;
- it ensures that the synthesized controllers can be easily implemented.

To meet those two requirements, we consider an idealized version of the environment in which the controller is embedded, but we put additional constraints into the winning objective of the controller that ensure the robustness of winning strategies. As the winning strategies are obtained in a simplified model of the system, we show how to embed automatically the synthesized strategies into a more detailed model of the environment, and how to automatically prove their correctness using the tool PHAVER for analyzing hybrid systems. While the verification model allows us to establish correctness of the controller that is obtained automatically using UPPAAL-TIGA, it does not allow us to learn its expected performance in an environment where noise is not completely antagonist but follows some probabilistic rules. For this kind of analysis, we consider a third model of the environment and we analyze the performance of our synthesized controller using SIMULINK.

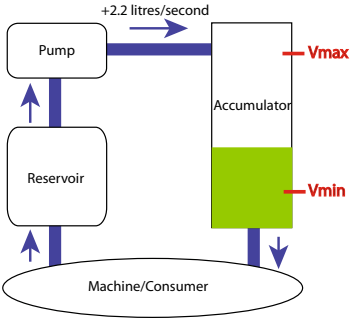


Fig. 1. Overview of the System

To show the advantages of our approach, we compare the performances of the controller we have automatically synthesized with two other control strategies. The first control strategy is a simple two-point control strategy where the pump is turned on when the volume of oil reaches a floor value and turned off when the volume of oil reaches a ceiling value. The second control strategy is a strategy designed by the engineers at HYDAC with the help of SIMULINK.

Structure of the paper. In section 2, we present the HYDAC control problem. In section 3, we present our construction of a suitable abstract model of the system, and the strategy we have obtained using the synthesis algorithm of UPPAAL-TIGA. In section 4, we embed the controllers into a continuous hybrid model of the environment and use the tool PHAVER to verify their correctness and robustness: we prove that strategies obtained using UPPAAL-TIGA are indeed correct and robust. In section 5, we analyze and compare the performances in term of mean volume of the three controllers using SIMULINK.

2 The Oil Pump Control Problem

In this section, we describe the components of the HYDAC case study using hybrid automata notations. Then we explain the control objectives for the system to design.

The Machine. The oil consumption of the machine is cyclic. One cycle of consumptions, as given by HYDAC, is depicted in Fig. 2(d). Each period of consumption is characterized by a rate of consumption m_r (expressed as a number of litres per second), a date of beginning, and a duration. We assume that the cycle is known *a priori*: we do not consider the problem of identifying the cycle (which can be performed as a pre-processing step). At time 2, the rate of the machine goes to $1.2l/s$ for two seconds. From 8 to 10 it is 1.2 again and from 10 to 12 it goes up to 2.5 (which is more than the maximal output of the pump). From 14 to 16 it is 1.7 and from 16 to 18 it is 0.5. Even if the consumption is cyclic and known in advance, the rate is subject to *noise*: if the mean consumption for a period is $c\ l/s$ (with $c > 0$), in reality it always lies within that period in the interval $[c - \epsilon, c + \epsilon]$, where ϵ is fixed to $0.1\ l/s$. This property is noted F.

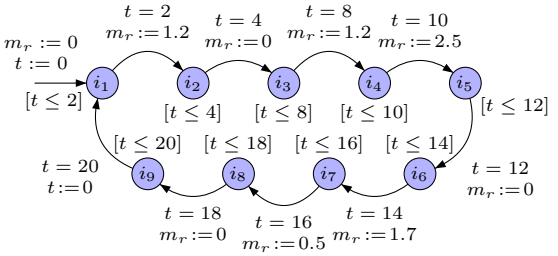
To model the machine, we use a timed automaton with 2 variables. The discrete variable m_r models the consumption rate of the machine, and the clock t is used to measure time within a cycle. The variable m_r is shared with the model of the accumulator. The timed automaton is given in Fig. 2(a). The noise on the rate of consumption is modeled in the model for the accumulator.

The Pump. The pump is either *On* or *Off*, and we assume it is initially *Off*. The operation of the pump must respect the following *latency* constraint: there must always be two seconds between any change of state of the pump, i.e. if it is turned *On* (respectively *Off*) at time t, it must stay *On* (respectively *Off*) at least until time $t + 2$: we note P_1 this property. When it is *On*, its *output* is equal to $2.2l/s$. We model the pump with a two states timed automaton given in Fig. 2(c) with two variables. The discrete variable p_r models the pumping rate of oil of the pump, and is shared with the accumulator. The clock z ensures that 2 t.u. have elapsed between two switches.

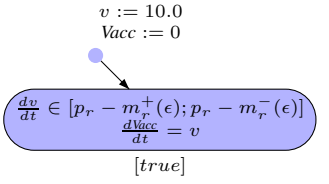
The Accumulator. To model the behavior of the accumulator, we use a one state hybrid automaton given in Fig. 2(b) that uses four variables. The variable v models the volume of oil within the accumulator, its evolution depends on the value of the variables m_r (the rate of consumption depending of the machine) and p_r (the rate of incoming oil from the pump). To model the imprecision on the rate of the consumption of the machine, the dynamics of the volume also depends on the parameter ϵ and is naturally given by the differential inclusion $dv/dt \in [p_r - m_r^-(\epsilon), p_r - m_r^+(\epsilon)]$ with $m_r^{\bowtie}(x) = m_r \bowtie x$ if $m_r > 0$ and m_r otherwise. The variable *Vacc* models the accumulated volume of oil along time in the accumulator. It is initially equal to 0 and its dynamic is naturally defined by the equation $dVacc/dt = v$.

The Control Problem. The controller must operate the pump (switch it *on* and *off*, respecting the latency constraint) to ensure the following two main requirements:

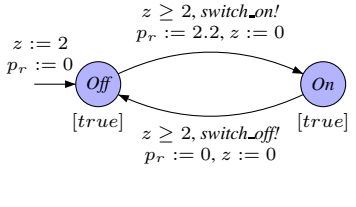
- (R_1): the level of oil $v(t)$ at time t (measured in litres) into the accumulator must always stay within two *safety* bounds $[V_{min}; V_{max}]$, in the sequel $V_{min} = 4.9l$ and $V_{max} = 25.1l$;



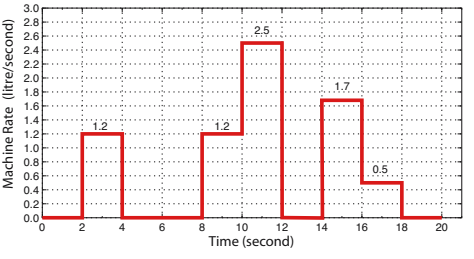
(a) The Machine



(b) The Accumulator



(c) The Pump



(d) Cycle of the Machine

Fig. 2. Hybrid Automaton Model of the System

– (R_2): a large amount of oil in the accumulator implies a high pressure of gas in the accumulator. This requires more energy from the pump to fill in the accumulator and also speeds up the wear of the machine. This is why the level of oil should be kept minimal during operation, in the sense that $\int_{t=0}^{t=T} v(t)dt$, that is $V_{acc}(T)$, is minimal for a given operation period T.

While (R_1) is a *safety requirement* and so must never be violated by any controller, (R_2) is an *optimality* requirement and will be used to compare different controllers.

Note that as the power of the pump is not always larger than the demand of the machine during one period of consumption (see Fig. 2(d) between 10 and 12), some extra amount of oil must be present in the accumulator before that period of consumption to ensure that the minimal amount of oil constraint (requirement R_1) is not violated[1].

Additional Requirements on the Controller. When designing a controller, we must decide what are the possible actions that the controller can take. Here are some considerations about that. First, as the consumptions are subject to noise, it is necessary to allow the controller to check periodically the level of oil in the accumulator (as it is not predictable in the long run). Second, as the consumption of the machine has a cyclic behavior, the controller should use this information to optimize the level of oil. So, it is natural to allow the controller to take control decisions at predefined instants during the cycle. Finally, we want a robust solution in the sense that if the controller has to turn *on* (or *off*) the pump at time t, it can do it a little before or after, that is at time $t \pm \Delta$

[1] It might be too late to switch the pump on when the volume reaches V_{min}.

for a small Δ without impairing safety. This robustness requirement will be taken into account in the synthesis and verification phases described later.

Two existing solutions. In the next sections, we will show how to use synthesis algorithms implemented in UPPAAL-TIGA to obtain a simple but still efficient controller for the oil pump. This controller will be compared to two other solutions that have been previously considered by the HYDAC company.

The first one is called the *Bang-Bang controller*. Using the sensor for oil volume in the accumulator, the Bang-Bang controller turns *on* the pump when a *floor* volume value V_1 is reached and turns *off* the pump when a *ceiling* volume value V_2 is reached. The Bang-Bang controller is thus a simple two-point controller, but it does not exploits the timing information about the consumption periods within a cycle.

To obtain better performances in term of energy consumption, engineers at HYDAC have designed a controller that exploit this timing. This second controller is called the *Smart controller*. This controller was designed by HYDAC, and works as follows [6]: in the first cycle the Bang-Bang controller is used and the volume $v(t)$ is measured and recorded every 10ms. According to the sampled values $v(t)$ computed in the initial cycle, an optimization procedure computes the points at which to start/stop the pump on the new cycle (this optimization procedure was given to us in the form of a C code executable into SIMULINK; unfortunately we do not have a mathematical specification of it). On this next cycle the values $v(t)$ are again recorded every 10ms which is the basis for the computation of the start/stop commands for the next cycle. If the volume leaves a predefined safety interval, the Bang-Bang controller is launched again. Though simulations of SIMULINK models developed by HYDAC reveal no unsafe behaviour, the engineers have not been able to verify its correctness and robustness. As we will see later, this strategy (we use the switching points in time obtained with SIMULINK when the C code is run) is not safe in the long run in presence of noise.

3 The UPPAAL-TIGA Model for Controller Synthesis

The hybrid automaton model presented in the previous section can be interpreted as a game in which the controller only supervises the pump. In this section, we show how to synthesize automatically, from a game model of the system and using UPPAAL-TIGA, an efficient controller for the Hydac case study. UPPAAL-TIGA is a recent extension of the tool UPPAAL which is able to solve timed games.

Game Models of Control Problems. While modeling control problems with games is very *natural* and *appealing*, we must keep in mind several important aspects. First, solving timed games is computationally hard, so we should aim at game models that are sufficiently abstract. Second, when modeling a system with a game model, we must also be careful about the information that is available to each player in the model. The current version of UPPAAL-TIGA offers *games of perfect information* (see [3] for steps towards games for imperfect information into UPPAAL-TIGA.) In games of perfect information, the two players have access to the full description of the state of the system. For simple objectives like safety or reachability, the strategies of the players are

functions from states to actions. To follow such strategies, the implementation of the controller must have access to the information contained in the states of the model. In practice, this information is acquired using sensors, timers, etc.

The UPPAAL-TIGA *Model.* We describe in the next paragraphs how we have obtained our game model for the hybrid automaton of the HYDAC case study. First, to keep the game model simple enough and to remain in a decidable framework[2], we have designed a model which: (a) considers one cycle of consumption; (b) uses an abstract model of the fluctuations of the rate; (c) uses a discretization of the dynamics within the system. Note that since the discretization impacts both the controller and the environment, it is neither an over- nor an under-approximation of the hybrid game model and thus we can not deduce directly the correctness of our controllers. However, our methodology includes a verification step based on PHAVER which allows us to prove this correctness. Second, to make sure that the winning strategies that will be computed by UPPAAL-TIGA are implementable, the states of our game model only contain the following information, which can be made available to an implementation:

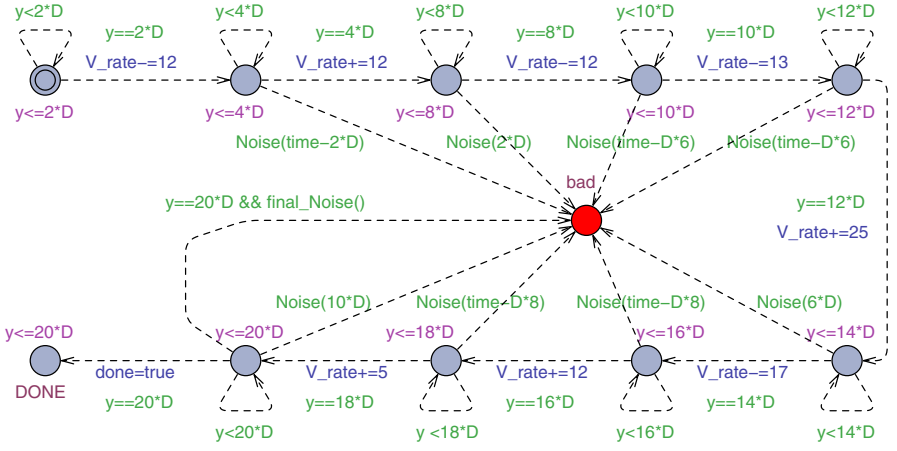
- the volume of oil at the beginning of the cycle; we thus only measure the oil once per cycle, leading to more simple controllers.
- the ideal volume as predicted by the consumption period in the cycle;
- the current time within the cycle;
- the state of the pump (*on* or *off*).

Third, to ensure robustness of our strategies, *i.e.* that their implementations are correct under imprecisions on measures of volume or time, we consider some margin parameter m which roughly represents how much the volume can deviate because of these imprecisions. We will consider values in range $[0.1; 0.4]l$.

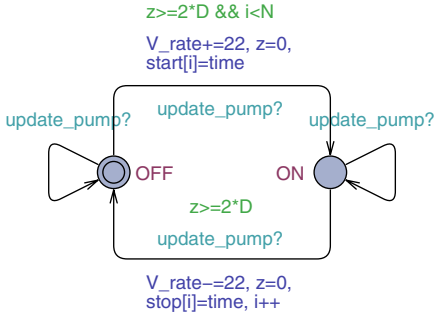
Global Variables. First, we discretize the time w.r.t. ratio stored in variable D, such that D time units represent one second. Second, we represent the current volume of oil by the variable V. We consider a precision of $0.1l$ and thus multiply the value of the volume by 10 to use integers. This volume evolves according to a rate stored in variable V_rate and the accumulated volume is stored in the variable V_acc[3]. Finally, we also use an integer variable time which measures the global time since the beginning of the cycle.

The Model of the Machine. The model for the behaviour of the machine is represented on Fig. 3(a). Note that all the transitions are uncontrollable (represented by dashed arrows). The construction of the nodes (except the middle one labelled bad) follows easily from the cyclic definition of the consumption of the machine. When a time at which the rate of consumption changes is reached, we simply update the value of the variable V_rate. The additional central node called bad is used to model the uncertainty on the value of V due to the fluctuations of the consumption. The function Noise (Fig. 4) checks whether the value of V, if modified by these fluctuations, may be outside the interval $[V_{min} + 0.1, V_{max} - 0.1]$[4]. The function final_Noise (Fig. 4) checks the

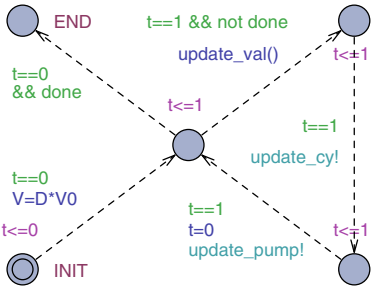
[2] The existence of winning strategies for timed games with costs in undecidable, see [2].
[3] To avoid integers divisions, we multiply all these values by D.
[4] For robustness, we restrain safety constraints of $0.1\ l$.



(a) Model of the cyclic consumption of the machine



(b) Model of the pump



(c) Model of the scheduler

Fig. 3. UPPAAL-TIGA models

same but for the volume obtained at the end of cycle and against the interval represented by V1F and V2F. Note that this modelling allows in some sense to perform partial observation using a tool for games of perfect information. Indeed, the natural modelling would modify at each step the actual value of the variable V and the strategies would then be aware of the amount of fluctuations. In our model the ideal value of V is predictable because it directly depends on the current time and from the point of view of the controller it does not give any information about the fluctuation.

The Model of the Pump. The model for the pump is represented on Fig. 3(b) and is very similar to the timed automaton given on Fig. 2(c). Note that the transitions are all controllable (plain arrows) and that we impose a bit more than P_1 as we require that 2 seconds have elapsed at the beginning of the cycle before switching on the pump. Moreover, an additional integer variable i is used to count how many times the pump has been started on. We use parameter N to bound this number of activations, which is set to 2 in the following. Note also that the time points of activation/deactivation of the pump are stored in two vectors start and stop.

```
bool Noise(int s){
// s is the number of t.u. of consumption
 return (V-s<(Vmin+1)*D|V+s>(Vmax-1)*D);}

bool final_Noise(){
// 10*D t.u. of consumption in 1 cycle
 return (V-10*D<V1F*D|V+10*D>V2F*D);}
```

```
void update_val(){
  int V_pred = V;
  time++;
  V+=V_rate;
  V_acc+=V+V_pred;
}
```

Fig. 4. Functions embedded in UppAal Tiga models

The Model of the Scheduler. We use a third automaton represented on Fig. 3(c) to schedule the composition. Initially it sets the value of the volume to V0 and then it repeats the following actions: it first updates the global variables V, V_acc and time through function update_val. Then the scheduling is performed using the two channels update_cy[5] and update_pump. When the end of the cycle of the machine is reached, the corresponding automaton sets the boolean variable done to true, which forces the scheduler to go to location END.

Composition. We denote by $\mathcal{A}$ the automaton obtained by the composition of the three automata described before. We consider as parameters the initial value of the volume, say V_0, and the target interval I_2, corresponding to V1F and V2F, and write $\mathcal{A}(V_0, I_2)$ the composed system.

Global Approach for Synthesis. Even if the game model that we consider is abstract and restricted to one cycle, note that our modelling enforces the constraints expressed in section 2. Indeed, R_1 is enforced through function Noise, F is handled through the two functions Noise and final_Noise, and P_1 is expressed explicitly in the model of the pump. To extend our analysis from one cycle to any number of cycles, and to optimize objective R_2, we formulate the following control objective (for some fixed margin $m \in \mathbb{Q}_{>0}$) :

Find some interval $I_1 = [V_1, V_2] \subseteq [4.9; 25.1]$ such that (Property $(*)$):

- (i) I_1 is m-*stable*: from all initial volume $V_0 \in I_1$, there exists a strategy for the controller to ensure that, whatever the fluctuations on the consumption, the value of the volume is always between 5 l and 25 l and the volume at the end of the cycle is within interval $I_2 = [V_1 + m, V_2 - m]$,
- (ii) I_1 is *optimal* among m-stable intervals: the supremum, over $V_0 \in I_1$ and over the strategies satisfying (i), of the accumulated volume is minimal.

The strategies that fulfill that control objective have a nice *inductive property*: as the value of the volume of oil at the end of the cycle is ensured to be within I_2, and $I_2 \subset I_1$ if $m > 0$, the strategies computed on our one cycle model can be safely repeated as many times as desired. Moreover, the choice of the margin parameter m will be done so as to ensure robustness. We will verify in PHAVER that even in presence of imprecisions, the final volume, if it does not belong to I_2, belongs to I_1: this is the reason why we fix a strict-subinterval of I_1 as a target in the synthesis phase.

[5] We did not represent this synchronization on Fig. 3(a) to ease the reading.

We now describe a procedure to compute an interval verifying Property $(*)$, and the associated strategies. We proceed as follows[6]:

1. For each $V_0 \in [4.9; 25.1]$, and target final interval $J \subseteq [4.9; 25.1]$, compute (by a binary search) the minimal accumulated volume $Score(V_0, J)$ that can be guaranteed. This value $Score(V_0, J)$ is
$$\min\{K \in \mathbb{N} \mid \mathcal{A}(V_0, J) \models \texttt{control: A<> Sched.END and V\_acc<=K}\}$$
2. Compute an interval $I_1 \subseteq [4.9; 25.1]$ such that, for $I_2 = [V_1 + m, V_2 - m]$:
   (a) $\forall V_0 \in I_1, \mathcal{A}(V_0, I_2) \models \texttt{control: A<> Sched.END}$
   (b) the value $Score(I_1) = \max\{Score(V_0, I_2) \mid V_0 \in I_1\}$ is minimal.
3. For each $V_0 \in I_1$, compute a control strategy $\mathcal{S}(V_0)$ for the control objective `A<> Sched.END and V_acc<=K` with `K` set to $Score(V_0, I_2)$. This strategy is defined by four dates of start/stop of the pump [7] and, by definition of $Score(V_0, I_2)$, minimizes the accumulated volume.

It is worth noticing that the value $Score$ is computed using the variable V_acc which is deduced from intermediary values of variable V. Since V corresponds to the value of the volume with no noise, V_acc represents the *mean value* of the accumulated volume for a given execution.

Results. For a margin $m = 0.4l$ and a granularity of 1 (D=1 in the UPPAAL-TIGA model), we obtain as optimal stable interval the interval $I_1 = [5.1, 10]$. The set of corresponding optimal strategies are represented on Fig. 5. For each value of the initial volume in the interval I_1, the corresponding period of activation of the pump is represented. We have represented volumes which share the same strategy in the same color. For the 50 initial possible values of volume, we obtain 10 different strategies (first row of Table 1). The overall strategy we synthesize thus measures the volume just once at the beginning of each cycle and play the corresponding "local strategy" until the beginning of next cycle.

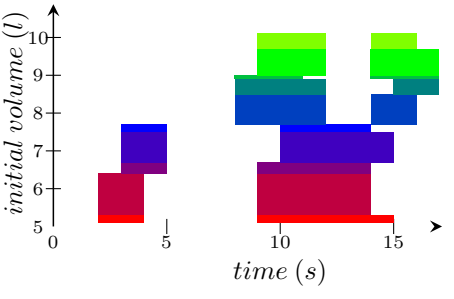


Fig. 5. Strategy for $D = 1$ and $m = 0.4\,\mathrm{l}$

Table 1 represents the results obtained for different granularities and margins. It gives the optimal stable interval I that is computed, (note that it is smaller if we allow a smaller margin or a finer granularity), the number of different local strategies, and the value of worst case mean volume which is obtained as $Score(I)/20$. These strategies are evaluated in sections 4 and 5.

4 Checking Correctness and Robustness of Controllers

In this section, we report on the results concerning the verification of the correctness robustness of the three solutions mentioned in the previous sections. To analyze the

[6] Control objectives are formulated as "control: P" following UPPAAL-TIGA syntax, where P is a TCTL formula specifying either a safety property `A[]`ϕ or a liveness property `A<>`ϕ.

[7] This is easy to obtain these times using the vectors start and stop of the pump.

Table 1. Main Characteristics of the Strategies Synthesized with UPPAAL-TIGA

Granularity	Margin	Stable interval	Number of strategies	Mean volume
1	4	$[5.1, 10]$	10	8.45
1	3	$[5.1, 9.8]$	10	8.35
1	2	$[5.1, 9.6]$	9	8.25
1	1	$[5.1, 9.4]$	9	8.2
2	4	$[5.1, 8.9]$	14	8.05
2	3	$[5.1, 8.7]$	14	7.95
2	2	$[5.1, 8.5]$	11	7.95
2	1	$[5.1, 8.3]$	11	7.95

correctness and the robustness of the three controllers, we use the tool PHAVER [4,5] for analysing hybrid systems. Robustness is checked according to the type of controller we use: for the Bang-Bang controller, it amounts to saying that the volume cannot be measured accurately and also that the rate fluctuates ($\pm 0.1 l/s$); for the Smart controller, robustness against rate fluctuation cannot be checked; for our synthesized controller, we take into account the rate fluctuation, the imprecision on the measure of the volume and the imprecision on the measure of time.

PHAVER allows us to consider a rich continuous time model of the system where we can take into account the fluctuations of consumption of the machine as well as adequate models of imprecisions inherent to any real implementation. The PHAVER models used in this paper are available in the extended version of this paper on the authors' webpages. This model takes into account the fluctuations in the consumption rate of the machine as well the imprecision on the measure of the volume. We now summarize the results for the three controllers.

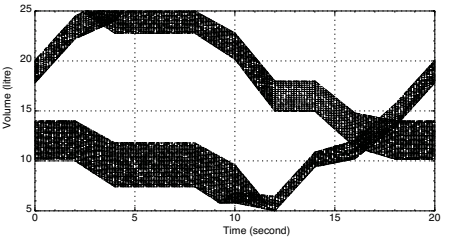


Fig. 6. Cyclic Behavior of the Bang-Bang controller with Noise

The Bang-Bang controller. To ensure robustness and implementability of this control strategy, we introduce imprecision in the measure of the oil volume: when the volume is read it may differ at most by $\epsilon = 0.06\ l$ from the actual value (precision of the sensor). Tuning this controller amounts to choose the tightest values for this floor and ceiling. In our experiment we found that 5.76 and 25.04 are the best margins we can expect.

With this PHAVER model and the previous margins[8], we are able to show that: (1) this control strategy enforces the safety requirement R_1, i.e. the volume of oil stays within the bounds $[4.9; 25.1]$; (2) the set of reachable states for initial volume equal to $10\ l$ can be computed and it is depicted in Fig. 6; this means that this controlled system is "cyclic" from the end of the first cycle on, and the same interval $[10.16; 14]$ (for the

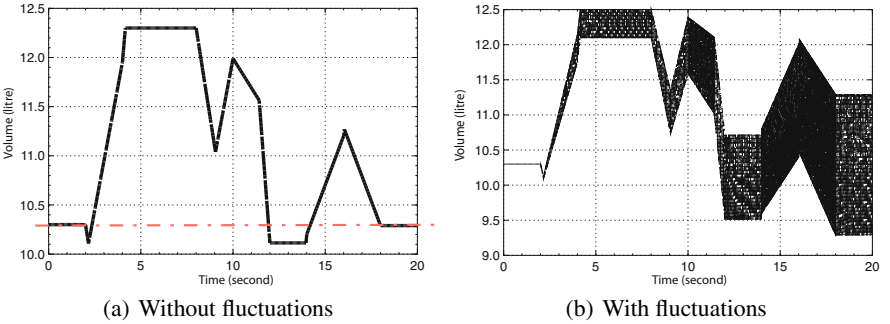
[8] And another suitable piece of PHAVER program for the computations.

volume) repeats every other cycle. It is thus possible to compute (with PHAVER) the interval of the accumulated volume over the two cycles: for this controller, the upper bound (worst case) is 307 and the mean volume is $307/20 = 15.35$.

The Smart Controller. The Smart Controller designed by HYDAC is specified by a 400 line C program and computes the start/stop dates for the next cycle according to what was observed in the previous cycle (see end of section 2). This controller requires to sample the plant every 10ms in order to compute the strategy to apply in the next cycle: although it is theoretically possible to specify this controller in PHAVER, this would require at least 100×20 discrete locations to store the sampled data in the previous cycle. It is thus not realistic to do this as PHAVER would not be able to complete an analysis of this model in a reasonable amount of time. Instead we have built the PHAVER controller that corresponds to the behaviour of the smart controller in a stationary regime, and in the absence of noise. It turns *on* and *off* so that the pump is active exactly during the three intervals $[2.16; 4.16]$, $[9.05; 11.42]$ and $[13.96; 16.04]$ during each cycle. Indeed using simulation, the engineers of HYDAC had discovered that the behavior of their controllers in the absence of noise was cyclic (stable on several cycles) if they started with an amount of oil equal to $10.3\,l$. This is confirmed by the simulations we report on at the end in Fig. 10 and by Fig. 7(a), obtained with PHAVER showing that the smart controller stabilizes with no fluctuations in the rate. However, our simplified version of the Smart controller (without imprecision on the dates of start and stop of the pump), is not robust against the fluctuations of the rate: the behavior of the system in the presence of noise is depicted in Fig. 7(b) and it can be shown with our PHAVER models that after four cycles, the safety requirement R_1 can be violated. Unfortunately, there is no way of proving the correctness of the *full* Smart controller with PHAVER, and SIMULINK only gives an average case. In this sense we cannot trust the Smart controller for ensuring the safety property.

The ideal Smart Controller (no noise on the rate) produces an average accumulated volume of around 221 per cycle i.e. an average volume of 11.05.

Controller Computed with UPPAAL-TIGA. We now study the correctness and robustness of the controller synthesized with UPPAAL-TIGA. This verification phase is



(a) Without fluctuations (b) With fluctuations

Fig. 7. Behavior of the HYDAC Smart Controller

necessary because during the synthesis phase we have used a very abstract model of the system and also discrete time. To force robustness and correctness, we have imposed additional requirements on the winning strategies (our inductive property together with the margin). But instead of proving by hand that the model and the objective are giving by construction robust and correct controller, it is more adequate to formally verify this. We summarize here the results of this verification phase. In the sequel we use the controller for granularity 2 and margin 4: this controller can be seen as 14 different local controllers, each one managing one of the 14 intervals in which the initial volume can be at the beginning of a cycle. We will focus on those strategies here but we have automated the process and the others may be treated along the same lines.

To make sure that our strategies are implementable, we have verified them in presence of fluctuations of the rate consumption and two types of imprecisions: on the date of start/stop of the pump (we use $\Delta = 0.01$ second), and on the measure of the initial volume, the imprecision being $0.06\ l$. Fig. 8 shows how the volume is controlled over 3 cycles: after the first one at $t = 20$, we measure the real volume with uncertainty $(0.06\ l)$ and use the corresponding controller from 20 to 40 and for 40 we again switch to another one.

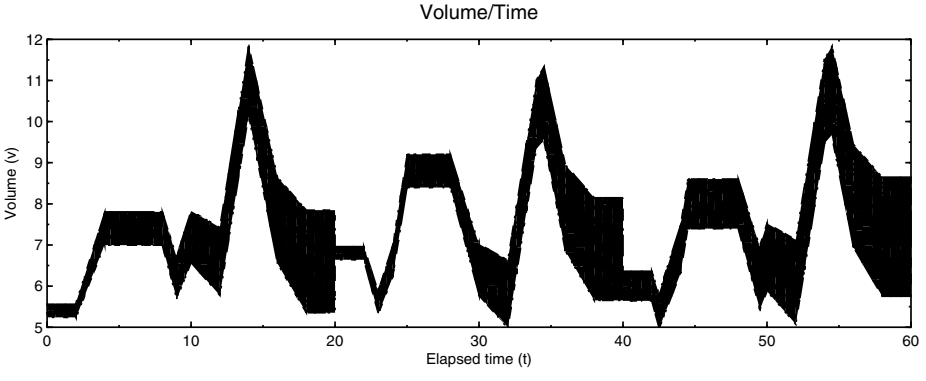


Fig. 8. The Pump Controlled over 3 Cycles with the UPPAAL-TIGA Controller

5 Simulation and Performances of the Controllers

In this section, we report on results obtained by simulating the three controller types in SIMULINK, with the purpose of evaluating their performance in terms of the accumulated volume of oil.

SIMULINK models of the *Bang-Bang* controller as well as of the *Smart* controller of HYDAC have been generously provided by the company. As for the eight controllers – differing in granularity and margin – synthesized by UPPAAL-TIGA, we have made a RUBY script which takes UPPAAL-TIGA strategies as input and transforms them into SIMULINK's *m*-format.

Fig. 9 shows the SIMULINK block diagram for simulation of the strategies synthesized by UPPAAL-TIGA. The diagram consist of built-in functions and four

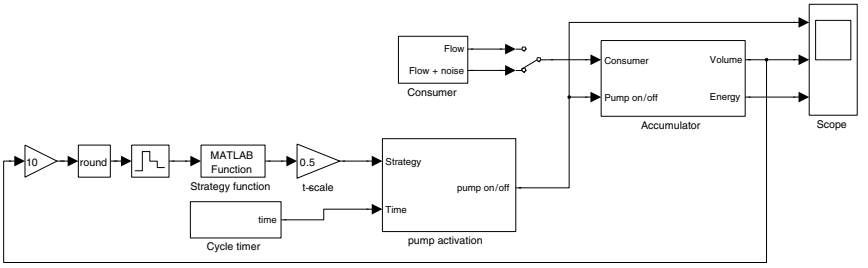
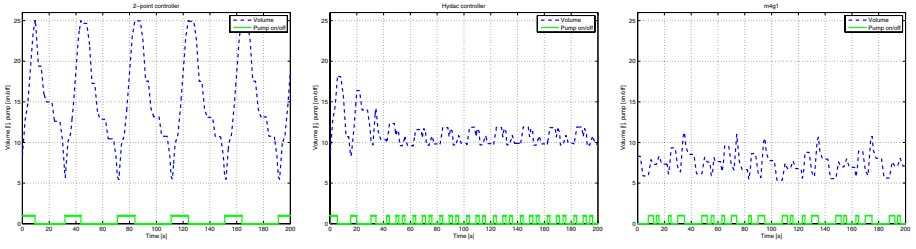


Fig. 9. The overall SIMULINK model



(a) Bang-Bang controller (b) Smart Controller (c) Synthesized Controller

Fig. 10. The three controller types with SIMULINK

subsystems: Consumer, Accumulator, Cycle timer and Pump activation (we omit the details of the subsystems). The Consumer subsystem defines the flow rates used by the machine with the addition of noise: here the choice of a uniform distribution on the interval $[-\epsilon, +\epsilon]$ with $\epsilon = 0.1l/s$ has been made. The Accumulator subsystem implements the continuous dynamics of the accumulator with a specified initial volume ($8.3l$ for the simulations). In order to use the synthesized strategies the volume is scaled with a factor 10, then rounded and feed into a zero-order hold function with a sample time of 20s. This ensures that the volume is kept constant during each cycle, which is feed into the strategy function. The Pump activation subsystem takes as input the on/off dates from the strategy (for the given input volume of the current cycle) and a Cycle timer, that holds the current time for each cycle.

Now, the plots in Fig. 10 are the result of SIMULINK simulations of the controllers, illustrating the volume of the accumulator as well as the state of the pump (on or off) for a duration of 200 s, i.e. 10 cycles. Though the simulations do not reveal the known violation of the safety requirement R_1 in the HYDAC Smart controller case, the simulations yield useful information concerning the performance of the controllers. In particular, the simulations indicate that the accumulated oil volume for all controllers grow linearly with time. Also, there is clear evidence that the strategies synthesized by UPPAAL-TIGA outperform the Smart controller of HYDAC – which is not robust – and also the Bang-Bang controller – which is robust but very non-optimal.

This is highlighted in Table 2, giving – for each of the ten strategies – the simulation results for the accumulated volume of oil , the corresponding mean volume as well as

Table 2. Performance characteristics based on SIMULINK simulations

Controller	Acc. volume	Mean volume	Mean volume (TIGA)
Bang-Bang	2689	13.45	-
HYDAC	2232	11.16	-
G1M4	1511	7.56	8.45
G1M3	1511	7.56	8.35
G1M2	1518	7.59	8.25
G1M1	1518	7.59	8, 2
G2M4	1527	7.64	8.05
G2M3	1513	7.57	7.95
G2M2	1500	7.5	7.95
G2M1	1489	7.44	7.95

the worst case mean volume according to synthesis of UPPAAL-TIGA. The table shows – as could be expected – that UPPAAL-TIGA's worst case mean volumes consistently are slightly more pessimistic that their simulation counter-parts. More interestingly, the simulation reveals that the performances of the synthesized controllers (e.g. G2M1) provide a vast improvement both of the Smart Controller of HYDAC (33%) and of the Bang-Bang Controller (45%).

6 Conclusion

In this paper we have presented a model-based methodology for the systematic development of robust and near-optimal controllers. The methodology applies a chain of tools for automatic synthesis (UPPAAL-TIGA), verification (PHAVER) and simulation (SIMULINK). Initially, sufficiently simple and abstract game models are used for synthesis. The correctness and robustness of the strategies are then verified using continuous hybrid models and – finally – the performance of the strategies are evaluated using simulation models.

Applied to the industrial case study provided by HYDAC, our method provides control strategies which outperforms the *Smart* controller as well as the simple *Bang-Bang* controller considered by the company. More important – whereas correctness and robustness of the Smart controller is unsettled – the strategies synthesized by our method are provably correct and robust. We believe that the case study demonstrates the maturity and industrial relevance of our tools.

Directions for further work include:

- Improve the performance of our controller further by optimizing over several cycles, and/or
- Improve the performance of our controller further by adding some predefined points when we can measure the volume (even with imprecision).
- Consideration of other imprecisions, e.g. with respect to the timing of consumer demands.

- Consideration of other optimization criteria. An interesting feature of the *Smart* controller of HYDAC seems to be that the oil volume is kept in a rather narrow interval, a feature which could possibly be beneficial for increasing the life-time of the Accumulator.
- Use the emerging version of UPPAAL-TIGA supporting synthesis under partial observability in order to allow more accurate initial game models.

References

1. Behrmann, G., Cougnard, A., David, A., Fleury, E., Larsen, K.G., Lime, D.: Uppaal-tiga: Time for playing games! In: Damm, W., Hermanns, H. (eds.) CAV 2007. LNCS, vol. 4590, pp. 121–125. Springer, Heidelberg (2007)
2. Brihaye, T., Bruyère, V., Raskin, J.-F.: On optimal timed strategies. In: Pettersson, P., Yi, W. (eds.) FORMATS 2005. LNCS, vol. 3829, pp. 49–64. Springer, Heidelberg (2005)
3. Cassez, F., David, A., Larsen, K.G., Lime, D., Raskin, J.-F.: Timed control with observation based and stuttering invariant strategies. In: Namjoshi, K.S., Yoneda, T., Higashino, T., Okamura, Y. (eds.) ATVA 2007. LNCS, vol. 4762, pp. 192–206. Springer, Heidelberg (2007)
4. Frehse, G.: Phaver: Algorithmic verification of hybrid systems past hytech. Int. Journal on Software Tools for Technology Transfer (STTT) 1(1–2) (1997); Extended and Revised version of [5]
5. Frehse, G.: Phaver: Algorithmic verification of hybrid systems past hytech. In: Morari, M., Thiele, L. (eds.) HSCC 2005. LNCS, vol. 3414, pp. 258–273. Springer, Heidelberg (2005)
6. Hermanns, H., Mittermüller, K.S., van Kuppeveld, T., Pedersen, J.S., Hougaard, P.: Preliminary descriptions of case studies. Quasimodo Deliverable D5.2, v1.0, Confidential Document (July 2008)
7. Jessen, J.J., Rasmussen, J.I., Larsen, K.G., David, A.: Guided controller synthesis for climate controller using Uppaal Tiga. In: Raskin, J.-F., Thiagarajan, P.S. (eds.) FORMATS 2007. LNCS, vol. 4763, pp. 227–240. Springer, Heidelberg (2007)
8. Simulink (2008), http://www.mathworks.com/products/simulink/

Local Identification of Piecewise Deterministic Models of Genetic Networks

Eugenio Cinquemani, Andreas Milias-Argeitis,
Sean Summers, and John Lygeros

Institut für Automatik, ETH Zurich, Switzerland
{cinquemani,milias,summers,lygeros}@control.ee.ethz.ch

Abstract. We address the identification of genetic networks under stationary conditions. A stochastic hybrid description of the genetic interactions is considered and an approximation of it in stationary conditions is derived. Contrary to traditional structure identification methods based on fitting deterministic models to several perturbed equilibria of the system, we set up an identification strategy which exploits randomness as an inherent perturbation of the system. Estimation of the dynamics of the system from sampled data under stability constraints is then formulated as a convex optimization problem. Numerical results are shown on an artificial genetic network model. While our methods are conceived for the identification of interaction networks, they can as well be applied in the study of general piecewise deterministic systems with randomly switching inputs.

Keywords: Piecewise deterministic systems, state-space identification, Markov processes, sampled systems, convex optimization.

1 Introduction

Genetic regulatory networks govern the synthesis of proteins in the living cell, and are thus responsible for fundamental cell functions such as metabolism, development and replication. Different approaches to genetic network modelling have been proposed in the literature and are conventionally classified into models with purely continuous dynamics and discrete event models [1]. However, it appears that certain systems are more naturally described by hybrid models that explicitly account for both continuous and discrete phenomena. This is witnessed by the number of researchers ([2,3,4,5,6], among others) who recently applied hybrid systems tools in this context. In addition, mounting experimental evidence suggests that gene expression, both in prokaryotes and eukaryotes, is an inherently stochastic process. Stochasticity can be attributed to the randomness of the transcription and translation processes (intrinsic noise), as well as to fluctuations in the amounts of molecular components that affect the expression of a certain gene (extrinsic noise), see [7,8,9,10]. In [11], stochastic modelling of genetic regulatory networks is reviewed along with numerical simulation methods and is compared to deterministic modelling. The authors have addressed

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 105–119, 2009.

stochastic hybrid modelling of genetic networks in [12]. A similar approach is taken in [13] for the analysis and numerical simulation of basic transcriptional network modules.

Recent works — [4,14,15,16,17] — have started to address the problem of learning genetic network models from experimental data. In particular, the literature on identification of stochastic regulatory network models is quite new [18,19,20,12]. A central problem in genetic network modelling is the identification of the network of interactions. Traditional approaches based on dynamic modelling rely on matching deterministic models to different equilibria corresponding to known perturbations of the system, see e.g. [21,22,23]. That is, one assumes that protein concentrations x evolve according to a kinetic model $\dot{x} = f(x, u)$, where u is a known perturbation input acting on the system. Then, a linearized model $\dot{x} = Ax + Bu$ is identified around several equilibrium points of the system corresponding to different constant values of u. This turns identification into a regression problem $0 = AX + BU$, where X is a matrix of observed equilibria and U is the matrix of the corresponding inputs. Matrix A carries information about the structure of the interaction network, hence the interest in its estimation. The main drawback of this approach is due to the assumption that A is the same at all equilibria. This implies that perturbations must be small. At the same time, several equilibria must be explored for the solution of the regression to be unique. The inherent random perturbations of the dynamics are not exploited in this case, in that the choice of deterministic modelling simply ignores this contribution.

In this paper we address identification of the structure of the network in a stochastic hybrid modelling framework. We start from the model described in [12] and consider a stochastic approximation of it around a stationary point of the system. Based on this, we borrow tools from the theory of identification of linear stochastic processes [24] to estimate the structure of the system. The conceptual difference with respect to traditional methods is that we make use of the randomness driving the system as a natural perturbation of the dynamics, with no further assumptions on the invariance of the dynamics. Artificial perturbations corresponding to several stationary conditions may be used to improve the estimation results and to separate different contributions, e.g. spontaneous degradation from regulatory effects. The identification procedure we propose relies on a local approximation of the stochastic hybrid model with a continuous stochastic model. This simplifies the identification problem but certain details of the network structure are lost in the approximation. The identification methods presented in [17,12], which build on the stochastic hybrid structure of the system, may then be used to recover the model in full detail.

The contribution of the paper is twofold. First, we introduce an approach to genetic network structure identification that accounts for and takes advantage of the inherent stochasticity of the systems. Of course, the approach requires that this randomness be reflected in the data. In view of the rapid progress of the protein level measurement techniques and of the advent of single-cell experiments [25,26], we believe that this approach is going to be applicable to

experimental data in the near future. Second, on a more theoretical level, we provide methods for the approximation and the identification of a family of stochastic hybrid models (namely the class of piecewise deterministic processes with switching inputs) that is relevant to a number of application scenarios.

The paper is organized as follows. In Section 2, we describe our stochastic hybrid framework for genetic network modelling. An approximate model of the stochastic hybrid dynamics under stationary conditions is derived in Section 3. Section 4 states the structure identification problem of our concern and describes a solution based on convex numerical optimization. A discussion of the method and of its possible extensions is developed in Section 5. The performance of our method is discussed in Section 6 by way of numerical experiments. Conclusions on and perspectives of our work are drawn in Section 7. Mathematical proofs are included in the appendix.

2 Piecewise Deterministic Models of Genetic Networks

A genetic network may be thought of as a collection of n proteins and of n corresponding genes along with their regulatory interactions. New molecules of a protein are synthesized when the gene that encodes it is expressed. The expression of a gene is regulated by one or more transcription factors (TFs). These are themselves proteins encoded by the genes of the network. In the simplest case, if a transcription factor is an activator (inhibitor), its binding to the promoter of the gene will activate (inhibit) a cascade of reactions that ultimately leads to the synthesis of new molecules of the protein encoded by that gene. In more generality, the simultaneous presence/absence of several transcription factors at the promoter site determines the status of the gene expression. We assume that changes in protein concentration due to synthesis and spontaneous degradation are well approximated by deterministic (kinetic) equations. On the other hand, the inherent randomness driving the binding/unbinding events and the presence of a limited number of binding sites leads us to model initiation and termination of gene expression as a stochastic process.

For a fixed $T \in \mathbb{R}_+$, let $\mathscr{T} = T \cdot \mathbb{N} = \{T, 2T, 3T, \ldots\}$ be a sequence of time instants. For $t \in \mathscr{T}$, let $x(t) \in \mathbb{R}_+^n$ be a continuous state vector of protein concentrations. Let $\ell(i) \subset \{1, \ldots, n\}$ denote the set of proteins acting as TFs on gene i. For each $k \in \ell(i)$ and $t \in \mathscr{T}$, let $u_{i,k}(t) \in \{0, 1\}$ be a discrete state variable that encodes the presence ($u_{i,k}(t) = 1$) or absence ($u_{i,k}(t) = 0$) of TF k at the promoter site of gene i. Therefore the activity of gene i is governed by a discrete state taking values in $\{0, 1\}^{|\ell(i)|}$, where $|\cdot|$ denotes set cardinality. Let $u(t) \in \{0, 1\}^m$, with $m = |\ell(1)| + \ldots + |\ell(n)|$, be a vector collecting all discrete variables $u_{i,k}(t)$, with $i = 1, \ldots, n$ and $k \in \ell(i)$. We model the evolution in time of the protein concentrations due to regulated synthesis and spontaneous degradation by the discrete-time dynamical equation

$$x(t + T) = Ax(t) + g\big(u(t + T)\big), \tag{1}$$

where $A \in \mathbb{R}_+^{n \times n}$ is a diagonal matrix of spontaneous degradation rates and $g : \mathbb{R}^m \to \mathbb{R}_+^n$ is a smooth function that quantifies the rate of synthesis of new

proteins in terms of the discrete state u. Typically, each component g_i of g takes the form

$$g_i(u) = \sum_j b_i^j \prod_{k \in \ell(i,j)} u_{i,k}, \tag{2}$$

where $b_i^j \in \mathbb{R}$ and the $\ell(i,j) \subseteq \{1,\dots,n\}$ are such that $\cup_j \ell(i,j) = \ell(i)$. To fix the ideas, each term of the summation corresponds to a different gene activation path, and b_i^j is the corresponding synthesis rate for protein i.

Stochasticity comes in the model by the description of the binding events, i.e. of the discrete transitions of u. Let $(\Omega, \mathscr{E}, \mathbb{P})$ be a probability space. For $t \in \mathscr{T}$, we describe the transitions of every $u_{i,k}$ as discrete random events with probabilities $\mathbb{P}[u_{i,k}(t+T)|u_{i,k}(t), x_k(t)]$ depending on the current protein concentrations $x_k(t)$ (e.g. the larger the concentration x_k, the larger the probability that a molecule of protein k binds to the promoter site of gene i). In light of this and Eq. (1), $u : \mathscr{T} \times \Omega \to \{0,1\}^m$ and $x : \mathscr{T} \times \Omega \to \mathbb{R}_+^n$ are two random processes defined on $(\Omega, \mathscr{E}, \mathbb{P})$. For simplicity we shall keep writing $x(t)$ and $u(t)$ in place of $x(t,\omega)$ and $u(t,\omega)$, where $\omega \in \Omega$. We impose the following two assumptions:

Assumption 1. *For every $t \in \mathscr{T}$, $u(t+T)$ and $x(t+T)$ are conditionally independent from the past history $x^-(t) = \{x(0), x(T), \dots, x(t-T)\}$ and $u^-(t) = \{u(0), u(T), \dots, u(t-T)\}$ given $x(t)$ and $u(t)$.*

Assumption 2. *For all $t \in \mathscr{T}$, the transition probability law*

$$p_{v,v'}(z) = \mathbb{P}[u(t+T) = v' | u(t) = v, x(t) = z], \qquad v, v' \in \{0,1\}^m,\ z \in \mathbb{R}_+^n, \tag{3}$$

is independent of t.

For a fixed initial condition $x(0) = x_0$ and an initial probability distribution $p_v^0 = \mathbb{P}[u(0) = v]$, the above completely specifies the probability laws of the joint process (x, u). A straightforward consequence of Assumptions 1 and 2 is

Proposition 1. *The joint process $\big(x(t), u(t)\big)$ is Markovian. For $t \in \mathscr{T}$,*

$$\begin{aligned}
&\mathbb{P}[x(t+T) = z', u(t+T) = v' | x(t) = z, u(t) = v, x^-(t), u^-(t)] = \\
&= \mathbb{P}[x(t+T) = z' | x(t) = z, u(t+T) = v'] \mathbb{P}[u(t+T) = v' | x(t) = z, u(t) = v] \\
&= \begin{cases} 0, & \text{if } z' \neq Az + g(v'), \\ p_{v,v'}(z), & \text{otherwise.} \end{cases}
\end{aligned}$$

For a given value of $u(t+T)$, Eq. (1) describes the transition of the continuous-valued process x from $x(t)$ to $x(t+T)$. We call $x(t)$ a piecewise deterministic process in that, as long as the value of u remains unchanged, the evolution of x is deterministic. On the other hand, for fixed values of $x(t)$ and $u(t)$, Eq. (3) determines the random outcome of the discrete-valued process $u(t+T)$. The joint process (x, u) resulting from the interconnection of the two processes is thus stochastic and hybrid. It is easy to recognize the class of processes defined above as a discrete-time variant of the Piecewise Deterministic Markov Processes introduced by [27].

To keep the analysis tractable we shall make a further assumption.

Assumption 3. *For every $t \in \mathscr{T}$, $u(t+T)$ is independent of $u(t)$ given $x(t)$.*

Biologically relevant conditions under which this assumption holds are discussed in [12,17]. Since $p_{v,v'}(z)$ is independent of v, we shall replace $p_{v,v'}(z)$ by $p_{v'}(z) = \mathbb{P}[u(t+T) = v' | x(t) = z]$.

3 Stochastic Approximation under Stationary Conditions

The stochastic hybrid structure of the genetic network model makes analysis and identification quite challenging. In [17] and [12] we proposed global methods for the identification of unknown model parameters that build on the stochastic hybrid model structure. The identification results are very good in that context since the full knowledge of the system structure was exploited. Yet, the associated optimization problems are nonconvex and generally hard to solve. Here we address the more difficult problem of structure identification and take an alternative approach to solve it. We approximate the stochastic hybrid dynamics locally by a continuous stochastic system and match the latter to the data. The resulting optimization problem is tractable, but the structure of the original stochastic hybrid model is partly obscured. In principle, the methods presented in [17,12] allow one to re-introduce the details of the network structure. This may be achieved via a series of heuristics which are currently being developed.

Assume that the joint process $\big(x(t), u(t)\big)$ has reached stationarity. Define $\bar{x} = \lim_{t\to\infty} \mathbb{E}[x(t)]$ and $\bar{u} = \lim_{t\to\infty} \mathbb{E}[u(t)]$, where $\mathbb{E}[\cdot]$ denotes expectation. Using the first-order expansion

$$g(u) = g\big(\bar{u}\big) + G_{\bar{u}}(u - \bar{u}) + o(u - \bar{u}) \simeq g(\bar{u}) + G_{\bar{u}} \cdot (u - \bar{u}),$$

where $G_{\bar{u}}$ is the Jacobian of g evaluated at $\bar{u}$, one may write

$$x(t+T) = Ax(t) + g\big(u(t+T)\big) \simeq Ax(t) + g(\bar{u}) + G_{\bar{u}} \cdot \big(u(t+T) - \bar{u}\big), \qquad (4)$$

the approximation being most accurate for small variance of $g\big(u(t)\big)$. Define $\tilde{x}(t) = x(t) - \bar{x}$ and $\tilde{u}(t) = u(t) - \bar{u}$.

Proposition 2. *Assume that (4) holds as an equality. Then $\bar{x} = A\bar{x} + g(\bar{u})$ and*

$$\tilde{x}(t+T) = A\tilde{x}(t) + G_{\bar{u}}\tilde{u}(t+T). \qquad (5)$$

We shall call (5) the approximate linear model for $\tilde{x}$. Of course, the model is not truly linear due to the dependence of $\tilde{u}$ on $\tilde{x}$. For any $\tilde{v} = v - \bar{u}$, with $v \in \{0,1\}^m$, and $\tilde{z} = z - \bar{x}$, with $z \in \mathbb{R}^n_+$, define $\tilde{p}_{\tilde{v}}(\tilde{z}) = \mathbb{P}[\tilde{u}(t+T) = \tilde{v} | \tilde{x}(t) = \tilde{z}]$.

Proposition 3. $\tilde{p}_{\tilde{v}}(\tilde{z}) = p_v(z)$.

Along with Eq. (5), this straightforward result provides locally an approximate model for the stochastic hybrid process $\big(\tilde{x}, \tilde{u}\big)$.

We are interested in the (approximate) second-order moments of the piecewise deterministic process $\tilde{x}(t)$. By definition, $\mathbb{E}[\tilde{x}(t)] = 0$. For $\ell \in \mathbb{Z}$, define the covariance function $\Sigma_x(\ell) = \mathbb{E}[\tilde{x}(t+\ell T)\tilde{x}(t)^T]$. By stationarity $\Sigma_x(\ell) = \Sigma_x(-\ell)^T$, and we may restrict our attention to $\ell \in \mathbb{N}$. Note that $\Sigma_x(0)$ is the covariance matrix of $\tilde{x}$.

Assumption 4. *There exists* $F_{\bar{x}} \in \mathbb{R}^{n\times n}$ *such that, for all* $t \in \mathbb{Z}$,

$$\mathbb{E}[\tilde{u}(t+T)|x(t)] = F_{\bar{x}}\tilde{x}(t). \tag{6}$$

Proposition 4. *For every* $\ell \in \mathbb{N}$ *it holds that*

$$\Sigma_x(\ell+1) = (A + G_{\bar{u}}F_{\bar{x}})\Sigma_x(\ell), \tag{7}$$

where $\Sigma_x(0)$ *is the solution of*

$$\Sigma_x(0) = (A + G_{\bar{u}}F_{\bar{x}})\Sigma_x(0)(A + G_{\bar{u}}F_{\bar{x}})^T + G_{\bar{u}}QG_{\bar{u}}^T. \tag{8}$$

In turn, $Q = \mathbb{E}\big[\mathrm{Var}\big(u(t+T)|x(t)\big)\big]$, *where* $\mathrm{Var}(\cdot|\cdot)$ *denotes conditional variance.*

Assumption 4 is met if $f(x) = \mathbb{E}[u(t+T)|x(t) = x]$ is linear. In practice, we shall assume that this is a valid approximation in a neighborhood of $\bar{x}$, i.e.

$$f(x) = f(\bar{x}) + F_{\bar{x}}\tilde{x} + o(\tilde{x}) \simeq f(\bar{x}) + F_{\bar{x}}\tilde{x}.$$

In this case, $\bar{u} = \mathbb{E}[u(t+T)] = \mathbb{E}\big[\mathbb{E}[u(t+T)|x(t)]\big] \simeq \mathbb{E}[f(\bar{x}) + F_{\bar{x}}\tilde{x}(t)] = f(\bar{x})$. Therefore, Assumption 4 is just a consequence of

$$\mathbb{E}[\tilde{u}(t+T)|x(t)] = \mathbb{E}[u(t+T)|x(t)] - \bar{u} \simeq \big(f(\bar{x}) + F_{\bar{x}}\tilde{x}(t)\big) - f(\bar{x}) = F_{\bar{x}}\tilde{x}(t).$$

Proposition 4 implies that the approximate second-order moments of the piecewise deterministic process $\tilde{x}$ are equal to those of a process described by the linear stationary state-space model

$$\tilde{x}(t+T) = \mathbb{A}_{\bar{x},\bar{u}}\tilde{x}(t) + G_{\bar{u}}w(t), \tag{9}$$

where $\mathbb{A}_{\bar{x},\bar{u}} = A + G_{\bar{u}}F_{\bar{x}}$ and $w(\cdot)$ is an i.i.d. process uncorrelated with $x^-(t)$ with mean zero and covariance matrix Q. Interestingly, this corresponds to replacing $\tilde{u}(t+T)$ in (5) with

$$\tilde{u}(t+T) = F_{\bar{x}}\tilde{x}(t) + w(t), \tag{10}$$

i.e. the gene regulation encoded by $\tilde{u}(t+T)$ may locally be thought of as a static linear state feedback with matrix gain $F_{\bar{x}}$ and additive noise w.

4 Constrained Identification of the Linearized Model

The approximation of the second-order moments of $\tilde{x}$ with those of (9) allows us to use concepts from the theory of linear stationary processes for the analysis and identification of piecewise deterministic systems. In view of the application to genetic network modelling, we are primarily interested in the estimation of the matrix $\mathbb{A}_{\bar{x},\bar{u}}$. This matrix combines spontaneous protein degradation (diagonal matrix A) with the effects of the regulatory interactions (matrix $G_{\bar{u}}F_{\bar{x}}$). In particular, $G_{\bar{u}}$ reflects the topology of the network, whereas $F_{\bar{x}}$ reflects the probability of each individual regulatory event near the stationary point $(\bar{x}, \bar{u})$. Note that the off-diagonal elements of $\mathbb{A}_{\bar{x},\bar{u}}$ only depend on the product $G_{\bar{u}}F_{\bar{x}}$.

As a result, the sign of each element $\left[\mathbb{A}_{\bar{x},\bar{u}}\right]_{i,k}$, with $i \neq k$, reveals the average (positive or negative) regulatory effect of protein k on the expression of gene i. A zero element, on the other hand, suggests that protein k is not involved in the regulation of gene i, at least around the stationary point $(\bar{x}, \bar{u})$. Therefore, the identification of $\mathbb{A}_{\bar{x},\bar{u}}$ provides information on the structure of the regulation network. A priori knowledge on the system (existing or non-existing interactions, for instance) should be accounted for at this stage. In this section we shall only constrain matrix $\mathbb{A}_{\bar{x},\bar{u}}$ to be stable. Local stability is a fundamental property of genetic regulatory networks near equilibria and is also central for the derivation of (5). If $\mathbb{A}$ were unstable, process (9), that is (5), would not be second-order stationary. From now on, we assume that $(\tilde{x}, \tilde{u})$ satisfies (5) and (6).

Assume that measurements y of (the protein concentrations) x are collected every $N > 0$ samples. This is captured by the following model:

$$y(\tau) = x(\tau) + n(\tau), \qquad \tau \in NT \cdot \mathbb{Z},$$

where n is a white noise process (not necessarily Gaussian), uncorrelated with x, with mean zero and covariance matrix $R = \mathbb{E}[nn^T]$. The identification problem is formulated as follows.

Problem 1. Given $M+1$ data points $\mathscr{Y} = \{y(t), y(t+NT), \ldots, y(t+MNT)\}$, compute an estimate $\widehat{\mathbb{A}}$ of $\mathbb{A}$ such that $\widehat{\mathbb{A}}$ is stable.

The case where $N > 1$ is especially relevant to genetic network identification. In this context, the discrete network events occur at a time scale T which is usually smaller than the time period that separates subsequent experimental measurements. Let $\bar{y} = \mathbb{E}[y(t)] = \mathbb{E}[x(t)] = \bar{x}$ be the mean of y and let $\tilde{y} = y - \bar{y}$. In practice, $\bar{y}$ can be estimated and removed from the data. For $\ell \in \mathbb{Z}$, define the covariance function $\Lambda(\ell N) \triangleq \mathbb{E}[\tilde{y}(t+\ell NT)\tilde{y}(t)^T]$. Note that $\Lambda(-\ell N) = \Lambda(\ell N)^T$ and that $\Lambda(0)$ is the covariance matrix of y.

Proposition 5. *$\Lambda(N) = \mathbb{A}^N\big(\Lambda(0) - R\big)$ and, for $\ell > 0$, $\Lambda(\ell N + N) = \mathbb{A}^N \Lambda(\ell N)$.*

For $\ell = 0, 1, \ldots, L$ with $L << M$, one may compute empirical estimates $\widehat{\Lambda}(\ell N)$ of $\Lambda(\ell N)$ as follows:

$$\widehat{\Lambda}(\ell N) = \frac{1}{M-\ell} \sum_{h=0}^{M-\ell} \tilde{y}(t+\ell NT+hNT)\tilde{y}(t+hNT)^T.$$

The approximation $\widehat{\Lambda}(\ell N) \simeq \Lambda(\ell N)$ is most accurate as $M \to \infty$. Assume for the time being that R is known. Define the $\mathbb{R}^{n \times nL}$ matrices

$$\widehat{\Lambda}_+ = \Big[\widehat{\Lambda}(N)\ \widehat{\Lambda}(2N) \cdots\ \widehat{\Lambda}(LN)\Big], \quad \widehat{\Lambda}_- = \Big[\big(\widehat{\Lambda}(0) - R\big)\ \widehat{\Lambda}(N) \cdots\ \widehat{\Lambda}\big((L-1)N\big)\Big].$$

In the light of Proposition 5, we address the identification Problem 1 by seeking a solution to the following optimization problem in the unknown matrix $\mathbb{A}$:

$$\text{minimize} \quad ||\widehat{\Lambda}_+ - \mathbb{A}^N \widehat{\Lambda}_-|| \qquad \text{subject to} \quad \mathbb{A} \text{ stable},$$

where $||\cdot||$ denotes the matrix spectral norm. In general, this problem is noncovex due to matrix exponentiation. To circumvent this issue, we use the fact that $\mathbb{A}$ is stable if and only if $\mathbb{A}^N$ is stable. We propose to solve the problem in two steps:

1. minimize $||\widehat{\Lambda}_+ - X\widehat{\Lambda}_-||$ subject to X stable. Denote the solution by $\widehat{X}$;
2. compute the matrix N-th root $\widehat{X}^{1/N}$.

Step 1. This amounts to matching the matrix $X = \mathbb{A}^N$ to the available covariance data in accordance with Proposition 5. By the Lyapunov theorem, the stability constraint on X is equivalent to the existence of a positive definite matrix P such that $XPX^T - P < 0$. Using Schur complement, this can be turned into the equivalent LMI

$$\begin{bmatrix} P & AP \\ PA^T & P \end{bmatrix} > 0,$$

with unknowns A and P. Define $Z = PA$. Using a series of standard transformations [28,29] based on the properties of the spectral norm, the problem can be reformulated in terms of the convex optimization

$$\text{minimize} \quad ||P\widehat{\Lambda}_+ - Z\widehat{\Lambda}_-|| \qquad \text{subject to} \quad \begin{bmatrix} P - \epsilon I & Z^T \\ Z & P \end{bmatrix} \geq 0, \quad P \geq I,$$

with unknowns Z and P. Here $\epsilon \in \mathbb{R}_+$ is a small design constant used to make the constraint set closed and to ensure the strict stability of the solution. This problem has a unique solution whenever the matrix $\widehat{\Lambda}_-$ has full row rank. Denote the solution with $\hat{Z}$ and $\hat{P}$. Then, setting $\widehat{X} = \hat{P}^{-1}\hat{Z}$ provides an approximate solution to the original problem.

Step 2. This requires the computation of the N-th root of a square matrix. The choice of the N-th matrix root is nonunique, see [30] for a detailed characterization of the solutions. However, provided the sampling time T is small enough, we expect that $\mathbb{A}$ is close to the identity, that is, all its eigenvalues should be located in a neighborhood of 1. Based on this consideration, we choose to compute the principal N-th root. By definition, this is the unique root matrix having all eigenvalues λ such that $\arg(\lambda) \in [-\pi/2N, \pi/2N]$, i.e. having all eigenvalues in the sector of the complex plain containing 1. Several algorithms for computing the principal root exist [30,31].

If R is unknown, we modify the problem by removing the leftmost $\mathbb{R}^{n\times n}$ element from $\widehat{\Lambda}_+$ and $\widehat{\Lambda}_-$. That is, we define the $\mathbb{R}^{n\times(L-1)n}$ matrices

$$\widehat{\Lambda}_+ = \Big[\widehat{\Lambda}(2N)\ \widehat{\Lambda}(3N) \cdots\ \widehat{\Lambda}(LN)\Big], \quad \widehat{\Lambda}_- = \Big[\big(\widehat{\Lambda}(N)\ \widehat{\Lambda}(2N) \cdots\ \widehat{\Lambda}\big((L-1)N\big)\Big]$$

and perform Steps 1 and 2 with these new matrices to get the estimate $\widehat{X}$. Provided $\widehat{X}$ is invertible, an estimate $\widehat{R}$ of R may be computed by solving $\widehat{\Lambda}(N) = \widehat{X}(\widehat{\Lambda}(0) - \widehat{R})$.

5 Discussion and Extensions

We mentioned above that the stability constraint can be equally imposed on $\mathbb{A}$ or on $\mathbb{A}^N$. To simplify the identification procedure, we decided to enforce this constraint on matrix $\mathbb{A}^N$ in the first identification step. In general, different information on the matrix $\mathbb{A}$, i.e. the sign of certain elements or the sparsity of the matrix, does not carry over to the matrix $\mathbb{A}^N$. If such prior knowledge on $\mathbb{A}$ is available, it is convenient to turn the identification scheme into a three-step procedure:

1′. minimize $||\widehat{\Lambda}_+ - X\widehat{\Lambda}_-||$ with respect to X, and name the solution $\widehat{X}$;
2′. compute the matrix N-th root $\widehat{X}^{1/N}$;
3′. minimize $||\widehat{X}^{1/N} - \mathbb{A}||$ subject to (stability and other) constraints on $\mathbb{A}$.

Step 1′ is an easy convex problem and serves the purpose of matching $X = \mathbb{A}^N$ to the data without constraints. If the spectral norm is replaced by the Frobenius norm, then the solution can be computed explicitly as $\widehat{X} = \widehat{\Lambda}_+\widehat{\Lambda}_-^R$, where $\widehat{\Lambda}_-^R = \widehat{\Lambda}_-^T\left(\widehat{\Lambda}_-\widehat{\Lambda}_-^T\right)^{-1}$ is the Moore-Pennrose pseudo-inverse of $\widehat{\Lambda}_-$. Step 2′ is the same as the former Step 2 but yields an unconstrained root matrix $\widehat{X}^{1/N}$. Finally, Step 3 seeks a constrained approximation of $\widehat{X}^{1/N}$ using the prior information on $\mathbb{A}$. Effective heuristics to solve this problem by convex optimization exist for many constraints of interest, see e.g. [21], and will not be further discussed here.

Given an estimate $\widehat{\mathbb{A}}_{\bar{x},\bar{u}}$, one cannot separate (the diagonal matrix) A from (the diagonal elements of) $G_{\bar{u}}F_{\bar{x}}$ and, in turn, $G_{\bar{u}}$ from $F_{\bar{x}}$. This is the same limitation of traditional methods. In these methods, however, perturbations of the system such as gene enhancement or knock-out are used to infer the overall system dynamics. In our setting, the overall dynamics $\mathbb{A}_{\bar{x},\bar{u}}$ are estimated based on a fixed experimental scenario, while system perturbations (i.e. estimates corresponding to different stationary points $(\bar{x},\bar{u})$) may be exploited to discern the individual contributions of A, $G_{\bar{u}}$ and $F_{\bar{x}}$.

In addition to the estimation of matrix $\mathbb{A}_{\bar{x},\bar{u}}$, our local approximation of the stochastic hybrid model can be used to learn the dimension of the system from the data. Consider for simplicity $N = 1$. It is well known that the rank of the block Hankel matrix

$$H = \begin{bmatrix} \Lambda(1) & \Lambda(2) & \Lambda(3) & \cdots \\ \Lambda(2) & \Lambda(3) & \Lambda(4) & \cdots \\ \Lambda(3) & \Lambda(4) & \Lambda(5) & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

associated with the linear system (9) is equal to the dimension of the state of a minimal realization of the system. Since the dimension of the state-space model (9) and of the piecewise deterministic model (1) are the same, the rank of H is indication of the dimension of the continuous state of (1). In the context of genetic network modelling, this type of analysis and other tools from the theory of realization/identification of linear stochastic processes [24] may help to develop methods for the estimation of the number of genes involved in the regulation of the observed proteins of the network.

6 Numerical Experiments

We consider an interaction network with four genes. The system is described by the stochastic hybrid model

$$\begin{aligned} x_1^+ &= \lambda_1 x_1 + b_1 u^-_{1,1}(1 - u^+_{1,2}u^-_{1,3}), & x_3^+ &= \lambda_3 x_3 + b^1_3 u^+_{3,1}u^+_{3,2}u^-_{3,3}, \\ x_2^+ &= \lambda_2 x_2 + b_2 u^-_{2,1}(1 - u^+_{2,2}u^-_{2,3}), & x_4^+ &= \lambda_4 x_4 + b^1_4 u^+_{4,1} + b^2_4. \end{aligned} \tag{11}$$

It is easy to verify that the protein synthesis rates are in the form (2). Processes $u^{\pm}_{i,k}(t+T)$ are independent given the current continuous state $x(t)$. The superscript $+$ or $-$ indicates whether the probability of $u_{i,k}(t+T)$ being equal to one is given by the sigmoidal function $\sigma^+_{i,k}(x_k) = x_k^d/(x_k^d + \theta^d)$ or by the complementary sigmoid $\sigma^-_{i,k}(x_k) = 1 - \sigma^+_{i,k}(x_k)$. Parameters $d \in \mathbb{R}_+$ and $\theta \in \mathbb{R}_+$ generally also depend on i, k. This model is in fact part of a larger model model for the nutrients stress response of bacterium *Escherichia coli*. The interested reader is deferred to [12] and references therein for a more detailed discussion.

Using biologically plausible parameter values and initial conditions [12], it can be observed by simulation that system (11) eventually reaches a stationary regime. Sample trajectories from the stationary regime are plotted in Fig. 1. We

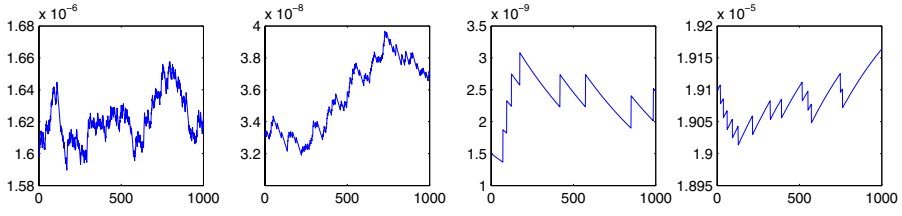


Fig. 1. Simulated trajectories of model (11) in stationary regime

perform local identification of the above model from simulated data. Estimation of the matrix $\mathbb{A}_{\bar{x},\bar{u}}$ is performed on the basis of a single trajectory $y(t)$, with $1 \le t \le 1000$. We considered four different experimental scenarios:

A. No measurement noise, no undersampling ($N = 1$)
B. With measurement noise, no undersampling ($N = 1$)
C. No measurement noise, with undersampling ($N = 10$)
D. With measurement noise, with undersampling ($N = 10$)

When $N = 1$, all 1000 data points $y(1), y(2), \ldots, y(1000)$ are used. When $N = 10$, only the 100 data points $y(10), y(20), \ldots, y(1000)$ are used. This simulates two biological experiments of the same duration but with different sampling rates. Such small size of the data sets reflects the typical experimental practice where a limited number of protein concentration measurements are collected sparsely in time by a single biological experiment. Noise, when applicable, is drawn from a normal distribution with mean zero and covariance matrix

$R = \mathrm{diag}(r_1^2, r_2^2, r_3^2, r_4^2)$, with $r_i = 0.01 \cdot \bar{x}_i$. In the identification process, the stationary mean value $\bar{y}$ is computed empirically and removed from the data y. Then, the two-step identification procedure (with $L = 2$ and R known) is applied to the data. For each of the four scenarios above, 100 estimates of the matrix $\mathbb{A}_{\bar{x},\bar{u}}$ are drawn from 100 random simulations on the model. For comparison, the true value of $\mathbb{A}_{\bar{x},\bar{u}}$ is computed from Eqn. (11), where the mean values $\bar{x}$ and $\bar{u}$ are computed empirically from the simulated trajectories. The mean value and the variance of the estimates of all elements of $\mathbb{A}_{\bar{x},\bar{u}}$ are reported in Fig. 2 along with the true values. In all cases, estimates are affected by very little or no bias which appears to be independent of the experimental conditions. The estimation variance is acceptable in almost all cases if one considers that a very limited data set

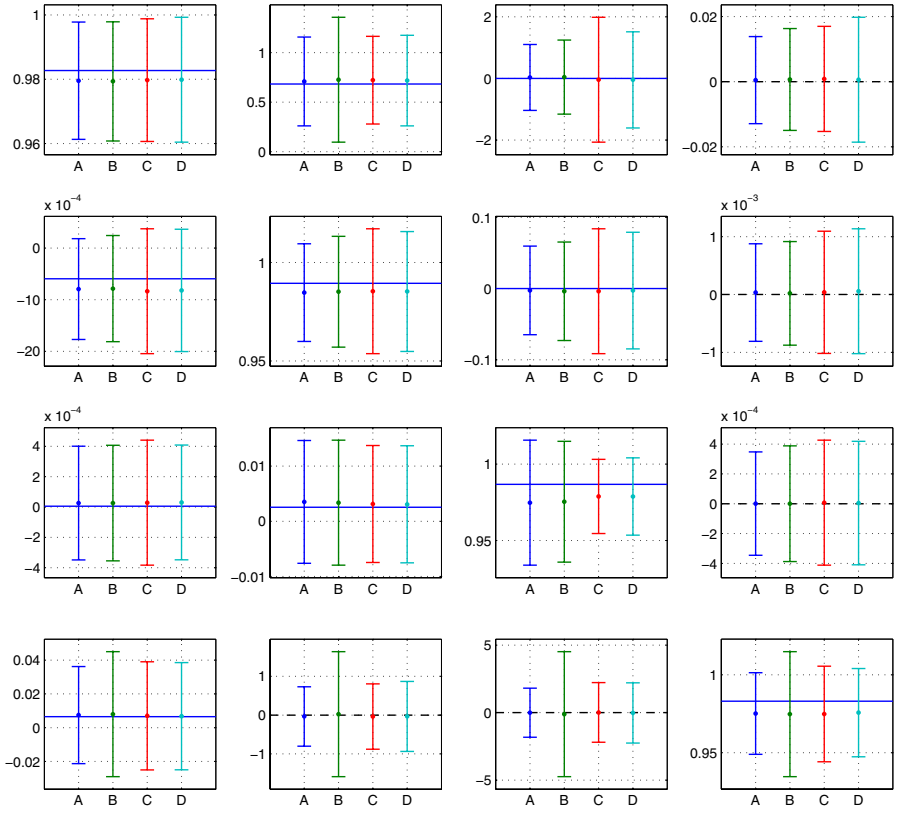


Fig. 2. For $r, c = 1, \ldots, 4$, the plot in the r-row and c-th column reports the estimation results for the (r, c)-th entry of $\mathbb{A}_{\bar{x},\bar{u}}$. In each plot, dots indicate the mean of the estimates and vertical bars correspond to 3 times the standard deviation of the estimates for the identification scenarios A (blue, left), B (green, second-left), C (red, second-right), D (cyan, right). Horizontal lines indicate the true entry values (dashed black lines for zeros, solid blue lines otherwise).

is used. As expected, the estimation variance generally increases with noise and is larger with larger values with N. Yet a 10-fold value of N is not detrimental for the estimation performance. For some elements of $\mathbb{A}_{\bar{x},\bar{u}}$, the estimates drawn in presence of measurement noise but with all data points available (scenario B) are by far the most uncertain. This exception is rather counterintuitive and deserves more investigation. Finally, unreported results comparing constrained and unconstrained estimation show that the stability constraint becomes active in roughly 10% of the estimation runs, the latter rate being larger in the presence of noise and undersampling.

7 Conclusions and Perspectives

We investigated the problem of genetic network structure identification in a stochastic hybrid modelling framework. We considered a piecewise deterministic model of genetic networks where protein synthesis is triggered by discrete random binding events and follows simple deterministic kinetics. We showed how to approximate the stochastic hybrid system locally via a linear stochastic system by considering the second order moments in stationarity. Using this approximation, we introduced an identification procedure that is based on matching the covariance function of the model to the data and provides an estimate of the average effect of each transcription factor on every gene. Extensions of the method were also discussed and include the estimation of the number of the genes in the network. Numerical results on simulated data witness the validity of the approach even in the presence of noisy and undersampled measurements. We are currently investigating on how to relax some of the assumptions mentioned in the paper and how to exploit system perturbations and experimental design to gain a more detailed insight into the structure of the network. In addition, we believe that our previous results on parameter estimation in stochastic hybrid models with known structure can be combined with the local structure identification procedure described in this paper to devise a full-blown stochastic hybrid model identification methodology. Heuristics for achieving this integration are currently under study.

Acknowledgements

This work was supported in part by the SystemsX.ch research consortium under the project YeastX.

References

1. de Jong, H.: Modeling and simulation of genetic regulatory systems: A literature review. Journal of Computational Biology 9(1), 69–105 (2002)
2. Alur, R., Belta, C., Ivancic, F., Kumar, V., Mintz, M., Pappas, G., Rubin, H., Schug, J.: Hybrid modeling and simulation of biological systems. In: Di Benedetto, M.D., Sangiovanni-Vincentelli, A.L. (eds.) HSCC 2001. LNCS, vol. 2034, pp. 19–32. Springer, Heidelberg (2001)

3. de Jong, H., Gouze, J.L., Hernandez, C., Page, M., Sari, T., Geiselmann, J.: Hybrid modeling and simulation of genetic regulatory networks: A qualitative approach. In: Maler, O., Pnueli, A. (eds.) HSCC 2003. LNCS, vol. 2623, pp. 267–282. Springer, Heidelberg (2003)
4. Drulhe, S., Ferrari-Trecate, G., de Jong, H., Viari, A.: Reconstruction of switching thresholds in piecewise-affine models of genetic regulatory networks. In: Hespanha, J.P., Tiwari, A. (eds.) HSCC 2006. LNCS, vol. 3927, pp. 184–199. Springer, Heidelberg (2006)
5. Batt, G., Ropers, D., de Jong, H., Geiselmann, J., Mateescu, R., Page, M., Schneider, D.: Validation of qualitative models of genetic regulatory networks by model checking: Analysis of the nutritional stress response in Escherichia coli. Bioinformatics 21(1), i19–i28 (2005)
6. Ghosh, R., Tomlin, C.: Symbolic reachable set computation of piecewise affine hybrid automata and its application to biological modeling: Delta-notch protein signaling. IET Systems Biology 1(1), 170–183 (2004)
7. Longo, D., Hasty, J.: Dynamics of single-cell gene expression. Molecular Systems Biology 2 (2006)
8. Elowitz, M.B., Levine, A.J., Siggia, E.D., Swain, P.S.: Stochastic gene expression in a single cell. Science 297(5584), 1183–1186 (2002)
9. McAdams, H.H., Arkin, A.: It's a noisy business! Genetic regulation at the nanomolar scale. Trends in Genetics 15(2), 65–69 (2002)
10. Paulsson, J.: Models of stochastic gene expression. Physics of Life Reviews 2(2), 157–175 (2005)
11. Samad, H.E., Khammash, M., Petzold, L., Gillespie, D.: Stochastic modeling of gene regulatory networks. International Journal of Robust and Nonlinear Control 15, 691–711 (2005)
12. Cinquemani, E., Milias-Argeitis, A., Summers, S., Lygeros, J.: Stochastic dynamics of genetic networks: modelling and parameter identification. Bioinformatics 24(23), 2748–2754 (2008)
13. Zeiser, S., Franz, U., Wittich, O., Liebscher, V.: Simulation of genetic networks modelled by piecewise deterministic markov processes. IET Systems Biology 2, 113–135 (2008)
14. Perkins, T., Hallett, M., Glass, L.: Inferring models of gene expression dynamics. Journal of Theoretical Biology 230(3), 289–299 (2004)
15. Fujarewicz, K., Kimmel, M., Swierniak, A.: On fitting of mathematical models of cell signaling pathways using adjoint systems. Mathematical Biosciences and Engineering 2(3), 527–534 (2005)
16. Dunlop, M., Franco, E., Murray, R.M.: A multi-model approach to identification of biosynthetic pathways. In: Proceedings of the 26th American Control Conference (2007)
17. Cinquemani, E., Porreca, R., Ferrari-Trecate, G., Lygeros, J.: Subtilin production by Bacillus subtilis: Stochastic hybrid models and parameter identification. IEEE Transactions on Automatic Control, Special Issue on Systems Biology 53, 38–50 (2008)
18. Reinker, S., Altman, R., Timmer, J.: Parameter estimation in stochastic biochemical reactions. IET Systems Biology 153, 168–178 (2006)
19. Tian, T., Xu, S., Gao, J., Burrage, K.: Simulated maximum likelihood method for estimating kinetic rates in gene expression. Bioinformatics 23(1), 84–91 (2007)
20. Golightly, A., Wilkinson, D.: Bayesian inference for stochastic kinetic models using a diffusion approximation. Biometrics (61), 781–788 (2005)

21. Zavlanos, M.M., Julius, A., Boyd, S.P., Pappas, G.J.: Identification of stable genetic networks using convex programming. In: Proceedings of the American Control Conference, Seattle, WA (June 2008)
22. Bansal, M., Belcastro, V., Ambesi-Impiombato, A., di Bernardo, D.: How to infer gene networks from expression profiles. Molecular Systems Biology 3(78)
23. Gardner, T.S., di Bernardo, D., Lorenz, D., Collins, J.J.: Inferring genetic networks and identifying compound mode of action via expression profiling. Science 301(5629), 102–105 (2003)
24. van Overschee, P., De Moor, B.L.: Subspace identification for linear systems: Theory - Implementation - Applications. Springer, Heidelberg (1996)
25. Golding, I., Paulsson, J., Zawilski, S.M., Cox, E.C.: Real-time kinetics of gene activity in individual bacteria. Cell 123(6), 1025–1036 (2005)
26. Cai, L., Friedman, N., Xie, X.S.: Stochastic protein expression in individual cells at the single molecule level. Nature 440, 358–362 (2006)
27. Davis, M.: Piecewise-deterministic Markov processes: A general class of non-diffusion stochastic models. Journal of the Royal Statistical Society B 46(3), 353–388 (1984)
28. Boyd, S.P., Vandenberghe, L.: Convex optimization. Cambridge University Press, Cambridge (2004)
29. Lacy, S.L., Bernstein, D.S.: Subspace identification with guaranteed stability using constrained optimization. IEEE Transactions on Automatic Control 48(7) (2003)
30. Smith, M.I.: A Schur algorithm for computing matrix pth roots. SIAM Journal on Matrix Analysis and Applications 24(4), 971–989 (2003)
31. Bini, D.A., Higham, N.J., Meini, B.: Algorithms for the matrix pth root. Numerical Algorithms 39, 349–378 (2005)

A Proofs

Proof of Proposition 2. The equation for $\bar{x} = \mathbb{E}[x(t)] = \mathbb{E}[x(t+T)]$ follows from $\mathbb{E}[x(t+T)] = \mathbb{E}\big[Ax(t) + g(\bar{u}) + G_{\bar{u}}\big(u(t+T) - \bar{u}\big)\big] = A\mathbb{E}[x(t)] + g(\bar{u})$. Using this equation and Eq. (4), $\tilde{x}(t+T) = Ax(t) + g(\bar{u}) + G_{\bar{u}}\tilde{u}(t+T) - \bar{x} = Ax(t) + G_{\bar{u}}\tilde{u}(t) - A\bar{x}(t)$, which is (5).

Proof of Proposition 4. *Without loss of generality, we shall prove the result for* $T = 1$. From Assumption 3, $\mathbb{E}[\tilde{u}(t+1)|x(t), x(t-\ell)] = \mathbb{E}[\tilde{u}(t+1)|x(t)]$ for all $\ell > 0$, where $\mathbb{E}[\cdot|\cdot]$ denotes conditional expectation. Eq. (7) is given by $\mathbb{E}[\tilde{x}(t+\ell+1)\tilde{x}(t)^T] = A\mathbb{E}[\tilde{x}(t+\ell)\tilde{x}(t)^T] + G_{\bar{u}}\mathbb{E}[\tilde{u}(t+\ell+1)\tilde{x}(t)^T]$ where

$$\begin{aligned}\mathbb{E}[\tilde{u}(t+\ell+1)\tilde{x}(t)^T] &= \mathbb{E}\big[\mathbb{E}[\tilde{u}(t+\ell+1)\tilde{x}(t)^T|x(t+\ell), x(t)]\big] = \\ &= \mathbb{E}\big[\mathbb{E}[\tilde{u}(t+\ell+1)|x(t+\ell)]\tilde{x}(t)^T\big] = F_{\bar{x}}\mathbb{E}[\tilde{x}(t+\ell)\tilde{x}(t)^T\big].\end{aligned}$$

To get Eq. (8), note that $\Sigma_x(0) = \mathbb{E}[\tilde{x}(t)\tilde{x}(t)^T] = \mathbb{E}[\tilde{x}(t+1)\tilde{x}(t+1)^T]$. Using (5) to expand the product in the latter expectation yields

$$\begin{aligned}\Sigma_x(0) = A\Sigma_x(0)A^T + G_{\bar{u}}\mathbb{E}[\tilde{u}(t+1)\tilde{x}(t)^T]A^T + A\mathbb{E}[\tilde{x}(t)\tilde{u}(t+1)^T]G_{\bar{u}}^T + \\ + G_{\bar{u}}\mathbb{E}[\tilde{u}(t+1)\tilde{u}(t+1)^T]G_{\bar{u}}^T. \quad (12)\end{aligned}$$

In turn, $\mathbb{E}[\tilde{u}(t+1)\tilde{x}(t)^T] = \mathbb{E}\big[\mathbb{E}[\tilde{u}(t+1)\tilde{x}(t)^T|x(t)]\big] = F_{\bar{x}}\mathbb{E}[\tilde{x}(t)\tilde{x}(t)^T] = F_{\bar{x}}\Sigma_x(0)$ and (writing $\tilde{u}$ for $\tilde{u}(t+T)$ and $\tilde{x}$ for $\tilde{x}(t)$)

$$\begin{aligned}
\mathbb{E}[\tilde{u}\tilde{u}^T] &= \mathbb{E}\Big[\big(u-\mathbb{E}[u|x]+\mathbb{E}[u|x]-\bar{u}\big)\big(u-\mathbb{E}[u|x]+\mathbb{E}[u|x]-\bar{u}\big)^T\Big] \\
&= \mathbb{E}\Big\{\mathbb{E}\Big[\big(u-\mathbb{E}[u|x]+\mathbb{E}[u|x]-\bar{u}\big)\big(u-\mathbb{E}[u|x]+\mathbb{E}[u|x]-\bar{u}\big)^T|x\Big]\Big\} \\
&\overset{*}{=} \mathbb{E}\Big\{\mathbb{E}\Big[\big(u-\mathbb{E}[u|x]\big)\big(u-\mathbb{E}[u|x]\big)^T|x\Big]+\mathbb{E}\Big[\big(\mathbb{E}[u|x]-\bar{u}\big)\big(\mathbb{E}[u|x]-\bar{u}\big)^T|x\Big]\Big\} \\
&= \mathbb{E}\big[\mathrm{Var}(u|x)\big] + \mathbb{E}\Big[\mathbb{E}\big[(F_{\bar{x}}\tilde{x})(F_{\bar{x}}\tilde{x})^T|x\big]\Big] = Q + F_{\bar{x}}\Sigma_x(0)F_{\bar{x}}^T.
\end{aligned}$$

(To verify "$*$", expand the product in the LHS and note that, since $\mathbb{E}[u|x] - \bar{u}$ is constant for given x and $\mathbb{E}\big[u-\mathbb{E}[u|x]|x\big] = 0$, the cross-product $\mathbb{E}\big[\big(u-\mathbb{E}[u|x]\big)\big(\mathbb{E}[u|x]-\bar{u}\big)^T|x\big]$ vanishes and so does its transpose.) Substituting these equations into (12) and rearranging the terms yields the result.

Proof of Proposition 5. The result can be deduced from the representation (9). *Without loss of generality, we may restrict to the case* $N = 1$. For the sake of simplicity let us also drop linearization points $\bar{x}$ and $\bar{u}$ from the notation. For $t \in \mathscr{T}$ and every $\ell \geq 1$,

$$\begin{aligned}
\mathbb{E}\big[\tilde{x}(t+\ell T+T)\tilde{y}(t)^T\big] &= \mathbb{E}\big[\big(\mathbb{A}\tilde{x}(t+\ell T)+Gw(t+\ell T)\big)\tilde{y}(t)^T\big] \\
&= \mathbb{A}\mathbb{E}\big[\tilde{x}(t+\ell T)\tilde{y}(t)^T\big].
\end{aligned}$$

On the other hand, $\mathbb{E}\big[\tilde{y}(t+\ell T)\tilde{y}(t)^T\big] = \mathbb{E}\big[\tilde{x}(t+\ell T)\tilde{y}(t)^T\big]$ because $n(t+\ell T)$ is uncorrelated with $\tilde{y}(t)$. Therefore $\Lambda(\ell+1) = \mathbb{A}\Lambda(\ell)$. For $\ell = 0$,

$$\mathbb{E}\big[\tilde{x}(t+T)\tilde{y}(t)^T\big] = \mathbb{E}\big[\big(\mathbb{A}\tilde{x}(t)+Gw(t)\big)\tilde{y}(t)^T\big] = \mathbb{A}\mathbb{E}\big[\tilde{x}(t)\tilde{y}(t)^T\big],$$

where $\mathbb{E}\big[\tilde{x}(t)\tilde{y}(t)^T\big] = \mathbb{E}\big[\big(\tilde{y}(t)-n(t)\big)\tilde{y}(t)^T\big] = \mathbb{E}\big[\tilde{y}(t)\tilde{y}(t)^T\big]-\mathbb{E}\big[n(t)n(t)^T\big]$, hence $\Lambda(1) = \mathbb{A}(\Lambda(0)-R)$.

Distributed Wombling by Robotic Sensor Networks

Jorge Cortés

Department of Mechanical and Aerospace Engineering
University of California, San Diego, CA 92093, USA
cortes@ucsd.edu

Abstract. This paper proposes a distributed coordination algorithm for robotic sensor networks to detect boundaries that separate areas of abrupt change of spatial phenomena. We consider an aggregate objective function, termed wombliness, that measures the change of the spatial field along the closed polygonal curve defined by the location of the sensors in the environment. We encode the network task as the optimization of the wombliness and characterize the smoothness properties of the objective function. In general, the complexity of the spatial phenomena makes the gradient flow cause self-intersections in the polygonal curve described by the network. Therefore, we design a distributed coordination algorithm that allows for network splitting and merging while guaranteeing the monotonic evolution of wombliness. The technical approach combines ideas from statistical estimation, dynamical systems, and hybrid modeling and design.

1 Introduction

Consider a network of mobile sensors moving in an environment with the objective of finding regions where large changes occur in a spatial phenomena of interest. Our aim is to design a distributed coordination algorithm that allows the group of sensors to determine boundaries that separate the areas with large differences in the spatial phenomena. The determination of such boundaries is relevant in multiple applications of robotic networks, including oceanographic surveys and weather forecasting. As an example, scientists are interested in determining regions of abrupt change in temperature fields over regions of the ocean, as they are related to upwelling and the food habits of fish.

The present work has connections with several scientific domains. In statistical estimation [1,2], wombling boundaries are curves that delimit areas of rapid change of some scientific phenomena of interest. Algorithms for detecting these boundaries based on point-referenced data are widely used for various applications, including biology [3], computational ecology [1], and medicine [4]. In computer vision [5,6], image segmentation and edge detection problems are encoded as optimization problems for a variety of objective functionals such as alignment, contrast, and geodesic active contour. These optimization problems are typically solved using PDE-based approaches that build on the variational information about the functionals. Finally, this work uses classical modeling and stabilization tools from hybrid systems theory [7,8,9,10] in the algorithm design.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 120–134, 2009.

The contributions of the paper are the following. We model the spatial phenomena as a deterministic spatial field. The wombliness of a non self-intersecting, closed curve is a measure of the alignment of the gradient of the spatial field along the normal direction to the curve. We use the notion of wombliness associated to a closed polygonal curve to formulate the network objective as a distributed optimization problem. We study the smoothness properties of the wombliness measure and provide an explicit expression for its gradient and a characterization of its critical points. If the network were to follow a gradient ascent law to optimize wombliness, then situations may arise where the polygonal curve described by the group of sensors becomes self-intersecting and the ensuing flow ill-posed. To prevent this from happening, we combine our analysis results with ideas from hybrid control design to synthesize a coordination algorithm for distributed wombliness optimization. The algorithm introduces the possibility of splitting and merging curves, and is guaranteed to monotonically optimize the wombliness measure associated to the network. Several simulations illustrate the results. For reasons of space, all proofs are omitted.

2 Preliminaries

Here, we gather some basic notions that will be frequently used along the paper. Let us start with some notation. We let $\mathrm{unit} : \mathbb{R}^2 \to \mathbb{R}^2$ denote the map defined by $\mathrm{unit}(x) = x/\|x\|$ for $x \neq 0$ and $\mathrm{unit}(0) = 0$. Given $n \in \mathbb{Z}_{>0}$ and $i, j \leq n$, let $\langle i, \dots, j \rangle$ be the set defined by $\langle i, \dots, j \rangle = \{i, \dots, j\}$ if $i \leq j$ and $\langle i, \dots, j \rangle = \{i, \dots, n, 1, \dots, j\}$ if $i > j$. Next, we introduce some useful geometric concepts.

2.1 Planar Geometric Notions

Given a vector $v = (v_1, v_2) \in \mathbb{R}^2$, we denote by $v^\perp = (v_2, -v_1) \in \mathbb{R}^2$ the vector perpendicular to v to the right, i.e., the 90 degree clockwise rotation of v. Given $p \neq q \in \mathbb{R}^2$, let $]p, q[$ and $[p, q]$ denote, respectively, the open and closed segments with end points p and q. We let $[p, q[$ denote the closed segment between p and q with the end point q excluded. We let $u_{[p,q]} = \mathrm{unit}(q - p)$ denote the unit vector in the direction from p to q and $n_{[p,q]} = u_{[p,q]}^\perp$ the unit normal vector to the right. In coordinates, if $p = (p_1, p_2)$ and $q = (q_1, q_2)$, then

$$u_{[p,q]} = \frac{1}{\|q - p\|}(q_1 - p_1, q_2 - p_2), \quad n_{[p,q]} = \frac{1}{\|q - p\|}(q_2 - p_2, p_1 - q_1).$$

We denote by $H^{\mathrm{out}}_{[p,q]} = \{z \in \mathbb{R}^2 \mid (z - p)^T n_{[p,q]} \geq 0\}$ the halfplane of points in the positive direction of the normal vector with respect to the closed segment $[p, q]$. Likewise, we denote $H^{\mathrm{in}}_{[p,q]} = \{z \in \mathbb{R}^2 \mid (z - p)^T n_{[p,q]} \leq 0\}$.

Given $p \in \mathbb{R}^2$ and $v \in \mathbb{R}^2$, we use the notation $\mathtt{ray}(p, v) = \{z \in \mathbb{R}^2 \mid z = p + tv,\ t \in \mathbb{R}_{\geq 0}\}$. The wedge $\mathtt{wedge}(p, (v_1, n_1), (v_2, n_2))$ is the cone with vertex p and axes $\mathtt{ray}(p, v_1)$ and $\mathtt{ray}(p, v_2)$. The interior of $\mathtt{wedge}(p, (v_1, n_1), (v_2, n_2))$ is the set of points towards which n_1 points along $\mathtt{ray}(p, v_1)$ and n_2 points along $\mathtt{ray}(p, v_2)$, see Figure 1 for an illustration. For the wedge to be well-defined, the normal vectors n_1 and n_2 need to specify the interior uniquely.

A domain $\mathcal{D} \subset \mathbb{R}^2$ is an open and simply connected set. Given $q \in \mathcal{D}$, let $T_q\mathcal{D}$ denote the set of all vectors tangent to $\mathcal{D}$ with origin at q. For $q \in \text{int}\,(\mathcal{D})$, $T_q\mathcal{D}$ is 2-dimensional and can be identified with $\mathbb{R}^2$. However, for $q \in \partial\mathcal{D}$, $T_q\mathcal{D}$ is one-dimensional and can be identified with $\mathbb{R}$. Let $T\mathcal{D}$ denote the collection $\cup\{T_q\mathcal{D} \mid q \in \mathcal{D}\}$ of all tangent vectors to $\mathcal{D}$. We let $\text{pr}_{T\mathcal{D}} : T_{\mathcal{D}}\mathbb{R}^2 \to T\mathcal{D}$ assign to each vector in $\mathbb{R}^2$ with origin at $q \in \mathcal{D}$ the orthogonal projection onto $T_q\mathcal{D}$. Any vector $v \in \mathbb{R}^2$ with origin in $\mathcal{D}$ has $\text{pr}_{T\mathcal{D}}(v) = v$.

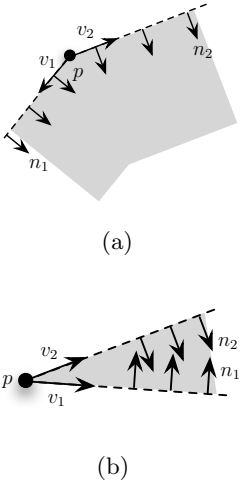


Fig. 1. Wedge determined by the point p and the pairs of vectors $(v_1.n_1)$ and (v_2, n_2)

2.2 Curve Parameterizations

A *curve* C in $\mathbb{R}^2$ is the image of a map $\gamma : [a, b] \to \mathbb{R}^2$. The map γ is called a *parametrization of* C. We often identify a curve with its parametrization. A curve C is *self-intersecting* if γ is not injective on (a, b). A curve C is *closed* if $\gamma(a) = \gamma(b)$. For a closed curve C, we let $n_C = \text{unit}(\dot{\gamma})^{\perp}$ denote the unit normal vector to C. A closed, not self-intersecting curve C partitions $\mathbb{R}^2$ into two disjoint open and connected sets, Inside_C and Outside_C, such that n_C along C points outside Inside_C and inside Outside_C, respectively. The orientation of C affects the definition of n_C and Inside_C, Outside_C, see Figure 2 for an illustration.

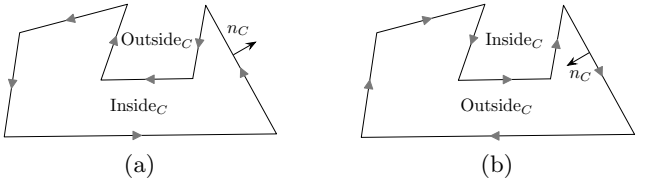


Fig. 2. Closed curve oriented in a (a) counterclockwise and (b) clockwise fashion

Given a curve C parametrized by a piecewise smooth map $\gamma : [a, b] \to C$, the line integral of a function $f : C \subset \mathbb{R}^2 \to \mathbb{R}$ over C is defined by

$$\int_C f = \int_C f(q)dq = \int_a^b f(\gamma(t))\,\|\dot{\gamma}(t)\|\,dt, \tag{1}$$

and it is independent of the selected parametrization.

3 Problem Statement

Let $Y : \mathbb{R}^2 \to \mathbb{R}$ be a twice continuously differentiable function modeling a planar spatial field. Consider a network of n mobile agents moving in a compact domain

$\mathcal{D} \subset \mathbb{R}^2$ with positions $p_1, \ldots, p_n$. Our objective is to find regions in $\mathcal{D}$ where large changes occur in the spatial field Y by determining their boundaries.

Let us start by defining a measure of how fast the field changes along a given curve. Let C be a non self-intersecting curve in $\mathbb{R}^2$ and define the *wombliness* or *alignment* of C by

$$\mathcal{H}(C) = \int_C \langle \nabla Y, n_C \rangle, \tag{2}$$

see e.g., [11,2]. The interpretation of the wombliness measure is as follows. At each point of the curve, we look at how much Y is changing along the normal direction to C (i.e., how much Y is "flowing through C"). The integral sums this change throughout the curve. We are interested in using the robotic network to find curves whose corresponding value of $\mathcal{H}$ is large.

For a closed non-self-intersecting curve, the wombling measure $\mathcal{H}$ can be rewritten, using the Gauss Divergence Theorem [12], as

$$\mathcal{H}(C) = \int_C \langle \nabla Y, n_C \rangle = \int_D \mathrm{div} \nabla Y = \int_D \Delta Y, \tag{3}$$

where D is the set in $\mathbb{R}^2$ whose boundary is C, and $\Delta Y = \frac{\partial^2 Y}{\partial x^2} + \frac{\partial^2 Y}{\partial y^2}$ denotes the Laplacian of Y. It is interesting to observe that, in general, that the level curves of the spatial field are not optimizers of $\mathcal{H}$.

In general, the optimization of (2) is an infinite-dimensional problem. Our approach here is to order counterclockwise the agents according to their unique identifier, and consider the closed polygonal curve that result from joining the positions of consecutive robots. In general, such curves may be self-intersecting. Therefore, we restrict our attention to the subset $\mathcal{S}_c$ of $\mathcal{D}^n$ defined as follows. For $(p_1, \ldots, p_n) \in \mathcal{D}^n$, let $\gamma_{\mathtt{cpc}}$ be the closed polygonal curve that results from the concatenation of the straight segments $[p_i, p_{i+1}]$, $i \in \{1, \ldots, n-1\}$ and $[p_n, p_1]$. Then, we define the following open subset of $\mathcal{D}^n$,

$$\mathcal{S}_c = \{(p_1, \ldots, p_n) \in \mathcal{D}^n \mid \gamma_{\mathtt{cpc}} \text{ is non-self-intersecting}\}.$$

Define the function $\mathcal{H}_c : \mathcal{S}_c \to \mathbb{R}$ by

$$\mathcal{H}_c(p_1, \ldots, p_n) = \mathcal{H}(\gamma_{\mathtt{cpc}}) = \sum_{i=1}^{n} \int_{[p_i, p_{i+1}]} \langle \nabla Y, n_{[p_i, p_{i+1}]} \rangle. \tag{4}$$

The optimization of (4) is now a finite-dimensional problem. Note that $\mathcal{H}_c$ can be expressed in terms of the polygon determined by the concatenated straight segments. If $\mathcal{P}(p_1, \ldots, p_n)$ denotes this polygon, then we have

$$\mathcal{H}_c(p_1, \ldots, p_n) = \int_{\mathcal{P}(p_1, \ldots, p_n)} \Delta Y. \tag{5}$$

For reasons that will become clear in the following sections, we assume that, at each network configuration, agent $i \in \{1, \ldots, n\}$ can measure the gradient ∇Y and the Laplacian ΔY along the segments $[p_{i-1}, p_i]$ and $[p_i, p_{i+1}]$.

4 Smoothness Analysis of the Wombliness Measure

In this section, we analyze the smoothness properties of the womblinness measure, provide explicit expressions for the gradient, and characterize the critical points. We start by stating the expression of the partial derivative of $\mathcal{H}_c$.

Proposition 1 (Gradient of $\mathcal{H}_c$). *The function $\mathcal{H}_c : \mathcal{S}_c \to \mathbb{R}$ is continuously differentiable. For each $i \in \{1, \dots, n\}$, the partial derivative of $\mathcal{H}_c$ with respect to p_i at $(p_1, \dots, p_n) \in \mathcal{S}_c$ is*

$$\frac{\partial \mathcal{H}_c}{\partial p_i} = \Big(\int_{[p_i, p_{i+1}]} \frac{\|p_{i+1} - q\|}{\|p_{i+1} - p_i\|} \Delta Y \Big) n_{[p_i, p_{i+1}]} + \Big(\int_{[p_{i-1}, p_i]} \frac{\|q - p_{i-1}\|}{\|p_i - p_{i-1}\|} \Delta Y \Big) n_{[p_{i-1}, p_i]}.$$

The proposition above implies in particular that the gradient of $\mathcal{H}_c$ is distributed over the ring graph: in other words, an agent i only needs to know about the location of its neighbors in the ring graph (agents $i-1$ and $i+1$) in order to be able to compute $\partial \mathcal{H}_c p_i$.

Using Proposition 1, we can characterize the critical configurations of $\mathcal{H}_c$.

Corollary 2 (Critical points of $\mathcal{H}_c$). *With a slight abuse of notation, let $\mathcal{H}_c : \overline{\mathcal{S}_c} \to \mathbb{R}$ denote the extension by continuity of $\mathcal{H}_c$ to $\overline{\mathcal{S}_c}$. Let $(p_1, \dots, p_n) \in \mathcal{S}_c$ be a critical configuration of $\mathcal{H}_c$. Then, for $i \in \{1, \dots, n\}$,*

$$\mathrm{pr}_{T\mathcal{D}} \Big(\frac{\partial \mathcal{H}_c}{\partial p_i} \Big) = 0.$$

Moreover, if $(p_1, \dots, p_n) \in \operatorname{int}(\mathcal{D}^n)$ and no three consecutive agents are aligned, this characterization can be alternatively described by, for $i \in \{1, \dots, n\}$,

$$\int_{[p_i, p_{i+1}]} \|p_{i+1} - q\| \Delta Y = 0, \qquad \int_{[p_i, p_{i+1}]} \|q - p_i\| \Delta Y = 0. \tag{6}$$

Remark 3 (Characterization of critical points of $\mathcal{H}_c$). The characterization (6) of the critical configurations of $\mathcal{H}_c$ in the interior of $\mathcal{D}$ has the following interpretation. For each $i \in \{1, \dots, n\}$, define the map $G_i : [p_i, p_{i+1}] \to \mathbb{R}$ by

$$z \mapsto G_i(z) = \int_{[p_i, z]} \Delta Y.$$

Note that $G(p_i) = 0$ by definition. Moreover, after some manipulations, one can show that equations (6) are equivalent to

$$G_i(p_{i+1}) = 0, \qquad \int_{[p_i, p_{i+1}]} G_i(z) dz = 0. \tag{7}$$

Using the fact that $\Delta Y = \operatorname{div}(\nabla Y)$, we can interpret the first equation in (7) as follows: on a critical configuration, there is no net average change of the gradient ∇Y along the segment $[p_i, p_{i+1}]$. However, even if this condition holds true, ∇Y might exhibit a preferred orientation with respect to $[p_i, p_{i+1}]$. It is precisely the second equation in (7) that takes care of ensuring that there is no bias in the orientation of ∇Y with respect to $[p_i, p_{i+1}]$. •

5 Distributed Hybrid Design for Wombliness Optimization

Our approach to find boundaries that delimit areas where the spatial field changes abruptly consists of starting with an initial network configuration and optimizing the magnitude of the wombliness of the closed polygonal boundary defined by the network. To maximize $\mathcal{H}_c$, we implement the distributed gradient flow of this function, cf. Proposition 1, that is,

$$\dot{p}_i = \operatorname{sgn}(\mathcal{H}_c(P_0))\, \operatorname{pr}_{T\mathcal{D}}\Big(\frac{\partial \mathcal{H}_c}{\partial p_i}\Big), \quad i \in \{1, \dots, n\}. \tag{8}$$

However, in general, the set $\overline{\mathcal{S}_c}$ is not invariant under (8). In other words, evolutions under (8) of the closed polygonal curve γ_{cpc} defined by the points $p_1, \dots, p_n$ become self-intersecting. To address this problem, we propose the following switching design, which is inspired on the interplay between the geometry of the polygonal curve γ_{cpc} and the value of the wombliness function $\mathcal{H}_c$.

5.1 Curve Self-intersection

Let γ_{cpc} be the closed polygonal curve defined by the segments $\{[p_i, p_{i+1}] \mid i \in \{1, \dots, n-1\}\} \cup [p_n, p_1]$. Assume $(p_1, \dots, p_n) \in \overline{\mathcal{S}_c}$, i.e., the curve γ_{cpc} is self-intersecting. Note that when a self-intersection occurs, either $\text{Inside}_{\gamma_{\mathsf{cpc}}}$ becomes disconnected or $\text{Outside}_{\gamma_{\mathsf{cpc}}}$ becomes disconnected. We refer to these two cases as *inside* and *outside* self-intersections, respectively. Figure 3 presents an illustration. We further distinguish between whether the self-intersection occurs at an open segment or at a point's location.

Self-intersection at an open segment. For each $i \neq j \in \{1, \dots, n\}$ such that $p_i \in]p_j, p_{j+1}[$, define $\lambda \in [0, 1)$ by $p_i = (1-\lambda)p_j + \lambda p_{j+1}$ and consider

$$v_i = (1-\lambda)u_j + \lambda u_{j+1}, \quad u_k = \operatorname{sgn}(\mathcal{H}_c(P))\, \operatorname{pr}_{T\mathcal{D}}\Big(\frac{\partial \mathcal{H}_c}{\partial p_k}\Big),$$

where $k \in \{i, j, j+1\}$. The guards depend upon the type of self-intersection.

Inside self-intersection. If the self-intersection is of inside type, it is because the segment $[p_i, p_{i+1}]$ belongs to $H^{\text{in}}_{[p_j, p_{j+1}]}$ and there exists the possibility of p_i crossing from $H^{\text{in}}_{[p_j, p_{j+1}]}$ to $H^{\text{out}}_{[p_j, p_{j+1}]}$, see Figure 3(a). The criterium to identify if a transition is needed in the network configuration is as follows. If

$$(u_i - v_i)^T n_{[p_j, p_{j+1}]} \leq 0,$$

then p_i does not cross, and the curve stays in $\overline{\mathcal{S}_c}$. If

$$(u_i - v_i)^T n_{[p_j, p_{j+1}]} > 0,$$

then the curve will move into $\mathcal{D}^n \setminus \overline{\mathcal{S}_c}$ unless the self-intersection is resolved.

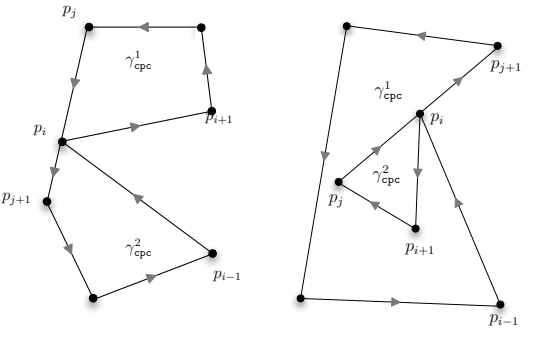
Outside self-intersection. If the self-intersection is of outside type, it is because the segment $[p_i, p_{i+1}]$ belongs to $H^{\text{out}}_{[p_j,p_{j+1}]}$ and there exists the possibility of p_i crossing from $H^{\text{out}}_{[p_j,p_{j+1}]}$ to $H^{\text{in}}_{[p_j,p_{j+1}]}$, see Figure 3(b). The criterium to identify if a transition is needed in the network configuration is as follows. If

$$(u_i - v_i)^T n_{[p_j,p_{j+1}]} \geq 0,$$

then p_i does not cross, and the curve stays in $\overline{\mathcal{S}_c}$. If

$$(u_i - v_i)^T n_{[p_j,p_{j+1}]} < 0,$$

the curve will move into $\mathcal{D}^n \setminus \overline{\mathcal{S}_c}$ unless the self-intersection is resolved.



(a) inside (b) outside

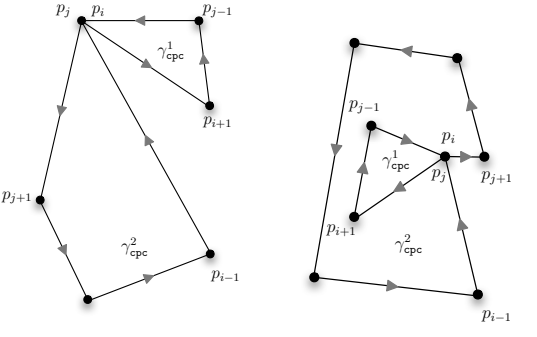
Fig. 3. The curve γ_{cpc} defined by $p_1, \ldots, p_n$ is self-intersecting at an open segment. (a) shows an inside self-intersection and (b) shows an outside self-intersection. In both cases, γ_{cpc} can be decomposed into two non-self-intersecting curves γ^1_{cpc} and γ^2_{cpc}.

Self-intersection at a point. For each $i \neq j \in \{1, \ldots, n\}$ such that $p_i = p_j$, consider the vectors

$$u_i = \text{sgn}(\mathcal{H}_c(P)) \frac{\partial \mathcal{H}_c}{\partial p_i},$$

$$u_j = \text{sgn}(\mathcal{H}_c(P)) \, \text{pr}_{T\mathcal{D}} \left(\frac{\partial \mathcal{H}_c}{\partial p_j} \right).$$

The guards depend upon the type of self-intersection.



(a) inside (b) outside

Fig. 4. The curve γ_{cpc} defined by $p_1, \ldots, p_n$ is self-intersecting at a point's location. (a) shows an inside self-intersection and (b) shows an outside self-intersection. In both cases, γ_{cpc} can be decomposed into two non-self-intersecting curves γ^1_{cpc} and γ^2_{cpc}.

Inside self-intersection. If the self-intersection is of inside type, see Figure 4(a), define the vectors

$$v_1 = \begin{cases} u_{[p_{i-1},p_i]} & \text{if } [p_{j-1}, p_j] \subset H^{\text{in}}_{[p_{i-1},p_i]}, \\ u_{[p_j,p_{j-1}]} & \text{if } [p_{j-1}, p_j] \not\subset H^{\text{in}}_{[p_{i-1},p_i]}, \end{cases}$$

$$v_2 = \begin{cases} u_{[p_{i+1},p_i]} & \text{if } [p_j, p_{j+1}] \subset H^{\text{in}}_{[p_i,p_{i+1}]}, \\ u_{[p_j,p_{j+1}]} & \text{if } [p_j, p_{j+1}] \not\subset H^{\text{in}}_{[p_i,p_{i+1}]}. \end{cases}$$

The criterium to identify if a transition is needed in the network configuration is as follows. If

$$u_i - u_j \in \texttt{wedge}(p_j, (v_1, v_1^{\perp}), (v_2, -v_2^{\perp})),$$

then the relative motion of p_i and p_j is such that the curve stays in $\overline{\mathcal{S}_c}$. If

$$u_i - u_j \notin \texttt{wedge}(p_j, (v_1, v_1^{\perp}), (v_2, -v_2^{\perp})),$$

then the curve will move into $\mathcal{D}^n \setminus \overline{\mathcal{S}_c}$ unless the self-intersection is resolved.

Outside self-intersection : If the self-intersection is of outside type, see Figure 4(b), define the vectors

$$v_1 = \begin{cases} u_{[p_j, p_{j-1}]} & \text{if } [p_{j-1}, p_j] \subset H^{\text{in}}_{[p_{i-1}, p_i]}, \\ u_{[p_{i-1}, p_i]} & \text{if } [p_{j-1}, p_j] \not\subset H^{\text{in}}_{[p_{i-1}, p_i]}, \end{cases}$$

$$v_2 = \begin{cases} u_{[p_j, p_{j+1}]} & \text{if } [p_j, p_{j+1}] \subset H^{\text{in}}_{[p_i, p_{i+1}]}, \\ u_{[p_{i+1}, p_i]} & \text{if } [p_j, p_{j+1}] \not\subset H^{\text{in}}_{[p_i, p_{i+1}]}. \end{cases}$$

The criterium to identify if a transition is needed in the network configuration is as follows. If

$$u_i - u_j \in \texttt{wedge}(p_j, (v_1, -v_1^{\perp}), (v_2, v_2^{\perp})),$$

then the relative motion of p_i and p_j is such that the curve stays in $\overline{\mathcal{S}_c}$. If

$$u_i - u_j \notin \texttt{wedge}(p_j, (v_1, -v_1^{\perp}), (v_2, v_2^{\perp})),$$

then the curve will move into $\mathcal{D}^n \setminus \overline{\mathcal{S}_c}$ unless the self-intersection is resolved.

State transition. We have encountered above the need to deal with self-intersections in $\gamma_{\texttt{cpc}}$ to prevent it from stepping into $\mathcal{D}^n \setminus \overline{\mathcal{S}_c}$. Next, we deal with these situations. For simplicity, we begin by considering the case where there is only one agent causing the self-intersection. If this is the case, then $\gamma_{\texttt{cpc}}$ can be decomposed into two polygonal curves $\gamma^1_{\texttt{cpc}}$ and $\gamma^2_{\texttt{cpc}}$, see Figures 3 and 4. The curve $\gamma^1_{\texttt{cpc}}$ is defined by the concatenation of the segments $\{[p_k, p_{k+1}] \mid k \in \langle i, \ldots, j-1 \rangle\} \cup [p_j, p_i]$, if $p_i \in]p_j, p_{j+1}[$, and $\{[p_k, p_{k+1}] \mid k \in \langle i+1, \ldots, j-1 \rangle\} \cup [p_j, p_{i+1}]$, if $p_i = p_j$. The curve $\gamma^2_{\texttt{cpc}}$ is defined in an analogous way as the concatenation of the segments $\{[p_k, p_{k+1}] \mid k \in \langle j+1, \ldots, i-1 \rangle\} \cup [p_i, p_{j+1}]$, if $p_i \in]p_j, p_{j+1}[$, and $\{[p_k, p_{k+1}] \mid k \in \langle j+1, \ldots, i-1 \rangle\} \cup [p_i, p_{j+1}]$, if $p_i = p_j$. Observe that $\gamma^2_{\texttt{cpc}}$ might not be oriented in a counterclockwise fashion. Moreover, if we are dealing with a self-intersection at an open segment, i.e., p_i belongs to $]p_j, p_{j+1}[$, note that p_i appears in the definition of both $\gamma^1_{\texttt{cpc}}$ and $\gamma^2_{\texttt{cpc}}$. The wombliness of $\gamma_{\texttt{cpc}}$ is split between $\gamma^1_{\texttt{cpc}}$ and $\gamma^2_{\texttt{cpc}}$ according to

$$\mathcal{H}(\gamma_{\texttt{cpc}}) = \mathcal{H}(\gamma^1_{\texttt{cpc}}) + \mathcal{H}(\gamma^2_{\texttt{cpc}}).$$

We are now ready to detail the two possible outcomes if a self-intersection occurs:

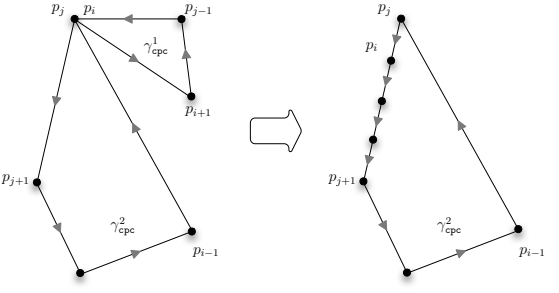


Fig. 5. Agent re-positioning. Agents in the curve $\gamma^1_{\texttt{cpc}}$ get re-positioned onto the curve $\gamma^2_{\texttt{cpc}}$.

Agent re-positioning. If $\mathcal{H}(\gamma^1_{\mathsf{cpc}})$ and $\mathcal{H}(\gamma^2_{\mathsf{cpc}})$ have different signs, we only keep the curve whose wombliness has the same sign as γ_{cpc}. Without loss of generality, assume the curve we keep is γ^2_{cpc}. Then, we re-position the agents in γ^1_{cpc} along the boundary of γ^2_{cpc}. This process does not affect the value of the wombliness of γ^2_{cpc}, and can be made in an arbitrary way. Note that the absolute value of the wombliness of the resulting non-self-intersecting curve is strictly larger than the value of the wombliness of the original self-intersecting curve γ_{cpc}. This transition is illustrated in Figure 5.

Curve splitting. If $\mathcal{H}(\gamma^1_{\mathsf{cpc}})$ and $\mathcal{H}(\gamma^2_{\mathsf{cpc}})$ have the same sign as $\mathcal{H}(\gamma_{\mathsf{cpc}})$, then choosing only one curve would lead to a decrease in the value of the wombliness. Therefore, we consider both. If the self-intersection occurs on an open segment, we need to add one more agent to the network at the intersection location, according to the definition of γ^1_{cpc} and γ^2_{cpc} above. After the split, each curve evolves independently according to (8). This transition is illustrated in Figure 6.

If multiple self-intersections occur at different locations, then the state transitions corresponding to each one of them can be executed simultaneously. If multiple self-intersections occur at the same location, then the curve γ_{cpc} can be decomposed into 3 or more non self-intersecting curves, and the state transition as described above can be conveniently modified to jointly consider the wombliness of each individual curve.

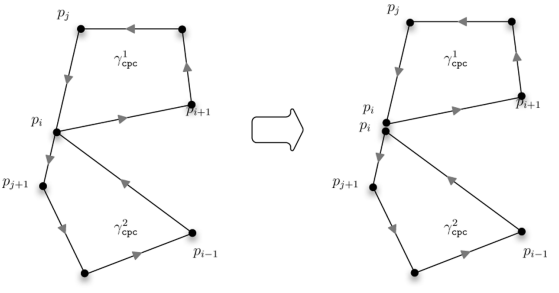


Fig. 6. Curve splitting. The curve γ_{cpc} is split into γ^1_{cpc} and γ^2_{cpc}, and these curves evolve independently afterwards.

5.2 Intersection between Curves

As a result of the curve splitting transition described in Section 5.1, there might be more than one curve moving in $\mathcal{D}$. It is therefore conceivable that along the ensuing evolution two of these curves intersect each other. Let us consider this situation. For simplicity, we only treat the case where there are two curves evolving in $\mathcal{D}$. The case with more than two curves can be treated in an analogous way. Let $\gamma^\alpha_{\mathsf{cpc}}$ be a closed polygonal curve determined by n_1 agents at positions $P^\alpha = (p^\alpha_1, \dots, p^\alpha_{n_1})$ and wombliness $\mathcal{H}^\alpha_c(P^\alpha) = \mathcal{H}(\gamma^\alpha_{\mathsf{cpc}})$, and let $\gamma^\beta_{\mathsf{cpc}}$ be a closed polygonal curve determined by n_2 agents at positions $P^\beta = (p^\beta_1, \dots, p^\beta_{n_2})$ and wombliness $\mathcal{H}^\beta_c(P^\beta) = \mathcal{H}(\gamma^\beta_{\mathsf{cpc}})$. Note that the orientation of the curves is not necessarily counterclockwise. When a intersection occurs between the two curves, either $\text{Inside}_{\gamma^\alpha_{\mathsf{cpc}}} \cap \text{Inside}_{\gamma^\beta_{\mathsf{cpc}}}$ is connected or $\text{Outside}_{\gamma^\alpha_{\mathsf{cpc}}} \cap \text{Outside}_{\gamma^\beta_{\mathsf{cpc}}}$ is connected. We refer to these two cases as *inside* and *outside* intersections, respectively. Figure 7 presents an illustration of these notions.

We further distinguish between whether the intersection occurs at an open segment or at a point's location.

Intersection at an open segment. For each $i \in \{1, \ldots, n_1\}$ such that $p_i^\alpha \in]p_j^\beta, p_{j+1}^\beta[$ for some $j \in \{1, \ldots, n_2\}$, define $\lambda \in [0, 1)$ by $p_i^\alpha = (1-\lambda)p_j^\beta + \lambda p_{j+1}^\beta$ and consider the vectors

$$v_i = (1-\lambda)u_j + \lambda u_{j+1},$$
$$u_i = \mathrm{sgn}(\mathcal{H}_c^\alpha(P^\alpha))\,\mathrm{pr}_{T\mathcal{D}}\left(\frac{\partial \mathcal{H}_c^\alpha}{\partial p_i^\alpha}\right), \quad u_k = \mathrm{sgn}(\mathcal{H}_c^\beta(P^\beta))\,\mathrm{pr}_{T\mathcal{D}}\left(\frac{\partial \mathcal{H}_c^\beta}{\partial p_k^\beta}\right),$$

where $k \in \{j, j+1\}$. The guards depend upon the type of self-intersection.

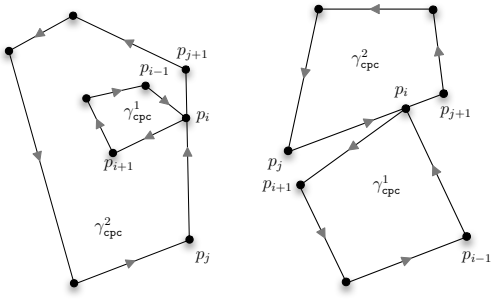
Inside intersection. If the intersection is of inside type, it is because the segment $[p_i^\alpha, p_{i+1}^\alpha]$ belongs to $H^{\mathrm{in}}_{[p_j^\beta, p_{j+1}^\beta]}$ and there exists the possibility of p_i^α crossing from $H^{\mathrm{in}}_{[p_j^\beta, p_{j+1}^\beta]}$ to $H^{\mathrm{out}}_{[p_j^\beta, p_{j+1}^\beta]}$, see Figure 7(a). The criterium to identify if a transition is needed in the network configuration is as follows. If

$$(u_i - v_i)^T n_{[p_j^\beta, p_{j+1}^\beta]} \leq 0,$$

then p_i^α does not cross. If

$$(u_i - v_i)^T n_{[p_j^\beta, p_{j+1}^\beta]} > 0,$$

then p_i^α will cross unless the intersection is resolved.



(a) inside (b) outside

Fig. 7. The curves $\gamma^\alpha_{\mathsf{cpc}}$ and $\gamma^\beta_{\mathsf{cpc}}$ intersect at an open segment. (a) shows an inside intersection and (b) shows an outside intersection. In both cases, the curves $\gamma^\alpha_{\mathsf{cpc}}$ and $\gamma^\beta_{\mathsf{cpc}}$ can be merged into a new self-intersecting curve γ_{cpc}.

Outside intersection. If the intersection is of outside type, it is because the segment $[p_i^\alpha, p_{i+1}^\alpha]$ belongs to $H^{\mathrm{out}}_{[p_j^\beta, p_{j+1}^\beta]}$ and there exists the possibility of p_i^α crossing from $H^{\mathrm{out}}_{[p_j^\beta, p_{j+1}^\beta]}$ to $H^{\mathrm{in}}_{[p_j^\beta, p_{j+1}^\beta]}$, see Figure 7(b). The criterium to identify if a transition is needed in the network configuration is as follows. If

$$(u_i - v_i)^T n_{[p_j^\beta, p_{j+1}^\beta]} \geq 0,$$

then p_i^α does not cross. If

$$(u_i - v_i)^T n_{[p_j^\beta, p_{j+1}^\beta]} < 0,$$

then p_i^α will cross unless the intersection is resolved.

Intersection at a point. For each $i \in \{1,\dots,n_1\}$ and $j \in \{1,\dots,n_2\}$ such that $p_i^\alpha = p_j^\beta$, consider the vectors

$$u_i = \operatorname{sgn}(\mathcal{H}_c(P^\alpha))\operatorname{pr}_{T\mathcal{D}}\left(\frac{\partial \mathcal{H}_c^\alpha}{\partial p_i^\alpha}\right), \quad u_j = \operatorname{sgn}(\mathcal{H}_c(P^\beta))\operatorname{pr}_{T\mathcal{D}}\left(\frac{\partial \mathcal{H}_c^\beta}{\partial p_j^\beta}\right).$$

The guards depend upon the type of intersection.

Inside intersection. If the intersection is of inside type, see Figure 8(a), define

$$v_1 = \begin{cases} u_{[p_{i-1}^\alpha, p_i^\alpha]} & \text{if } [p_{j-1}^\beta, p_j^\beta] \subset H^{\text{in}}_{[p_{i-1}^\alpha, p_i^\alpha]}, \\ u_{[p_j^\beta, p_{j-1}^\beta]} & \text{if } [p_{j-1}^\beta, p_j^\beta] \not\subset H^{\text{in}}_{[p_{i-1}^\alpha, p_i^\alpha]}, \end{cases}$$

$$v_2 = \begin{cases} u_{[p_{i+1}^\alpha, p_i^\alpha]} & \text{if } [p_j^\beta, p_{j+1}^\beta] \subset H^{\text{in}}_{[p_i^\alpha, p_{i+1}^\alpha]}, \\ u_{[p_j^\beta, p_{j+1}^\beta]} & \text{if } [p_j^\beta, p_{j+1}^\beta] \not\subset H^{\text{in}}_{[p_i^\alpha, p_{i+1}^\alpha]}. \end{cases}$$

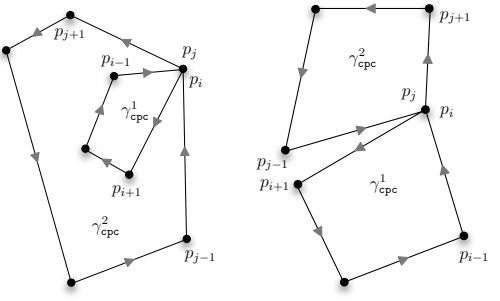
The criterium to identify if a transition is needed is as follows. If

$$u_i - u_j \in \texttt{wedge}(p_j^\beta, (v_1, v_1^\perp), (v_2, -v_2^\perp)),$$

then the relative motion of p_i^α and p_j^β is such that the curves $\gamma_{\texttt{cpc}}^\alpha$ and $\gamma_{\texttt{cpc}}^\beta$ evolve without "crossing each other." If

$$u_i - u_j \notin \texttt{wedge}(p_j^\beta, (v_1, v_1^\perp), (v_2, -v_2^\perp)),$$

then the intersection needs to be resolved.



(a) inside (b) outside

Fig. 8. The curves $\gamma_{\texttt{cpc}}^\alpha$ and $\gamma_{\texttt{cpc}}^\beta$ intersect at a point's location. (a) shows an inside intersection and (b) shows an outside intersection. In both cases, the curves $\gamma_{\texttt{cpc}}^\alpha$ and $\gamma_{\texttt{cpc}}^\beta$ can be merged into a new self-intersecting curve $\gamma_{\texttt{cpc}}$.

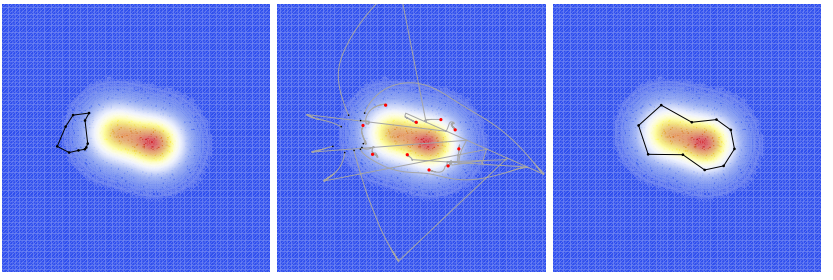
Outside intersection. If the intersection is of outside type, see Figure 8(b), define

$$v_1 = \begin{cases} u_{[p_j^\beta, p_{j-1}^\beta]} & \text{if } [p_{j-1}^\beta, p_j^\beta] \subset H^{\text{in}}_{[p_{i-1}^\alpha, p_i^\alpha]}, \\ u_{[p_{i-1}^\alpha, p_i^\alpha]} & \text{if } [p_{j-1}^\beta, p_j^\beta] \not\subset H^{\text{in}}_{[p_{i-1}^\alpha, p_i^\alpha]}, \end{cases}$$

$$v_2 = \begin{cases} u_{[p_j^\beta, p_{j+1}^\beta]} & \text{if } [p_j^\beta, p_{j+1}^\beta] \subset H^{\text{in}}_{[p_i^\alpha, p_{i+1}^\alpha]}, \\ u_{[p_{i+1}^\alpha, p_i^\alpha]} & \text{if } [p_j^\beta, p_{j+1}^\beta] \not\subset H^{\text{in}}_{[p_i^\alpha, p_{i+1}^\alpha]}. \end{cases}$$

The criterium to identify if a transition is needed is as follows. If

$$u_i - u_j \in \texttt{wedge}(p_j^\beta, (v_1, -v_1^\perp), (v_2, v_2^\perp)),$$



(a) (b) (c)

Fig. 9. Robotic network of 10 agents evolving under the wombling coordination algorithm. (a) shows the initial configuration, (b) shows the robot trajectories, and (c) shows the final configuration. The spatial field is $Y(x_1, x_2) = 1.25e^{-(x_1+.75)^2-(x_2-.2)^2} + 1.75e^{-(x_1-.75)^2-(x_2+.2)^2}$. The gradient flow (8) first triggers 1 outside self-intersection and then 2 inside self-intersections. All transitions result in agent re-positionings.

then the relative motion of p_i^α and p_j^β is such that the curves $\gamma_{\mathtt{cpc}}^\alpha$ and $\gamma_{\mathtt{cpc}}^\beta$ evolve without "crossing each other." If

$$u_i - u_j \notin \mathtt{wedge}(p_j^\beta, (v_1, -v_1^\perp), (v_2, v_2^\perp)),$$

then the intersection needs to be resolved.

State transition. We have encountered above the necessity to deal with intersections between the curves $\gamma_{\mathtt{cpc}}^1$ and $\gamma_{\mathtt{cpc}}^2$. For simplicity, we begin by considering the case where there is only one agent causing the intersection. If this is the case, then the two curves can be merged into a single one, see Figures 7 and 8. The closed polygonal curve $\gamma_{\mathtt{cpc}}$ is defined by the concatenation of the segments

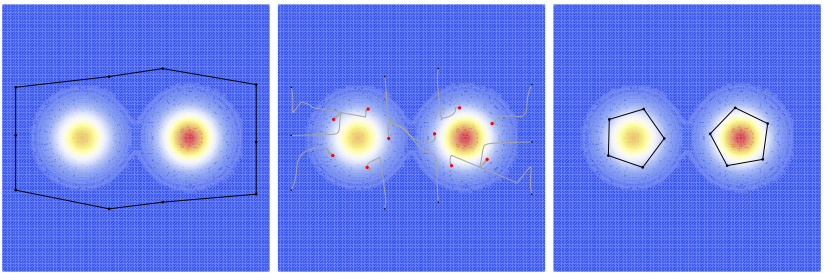
$$\{[p_k^\alpha, p_{k+1}^\alpha] \mid k \in \langle i, \ldots, i-1\rangle\} \cup [p_i^\alpha, p_{j+1}^\beta] \\ \cup \{[p_k^\beta, p_{k+1}^\beta] \mid k \in \langle j+1, \ldots, j-1\rangle\} \cup [p_j^\beta, p_i^\alpha],$$

if $p_i^\alpha \in]p_j^\beta, p_{j+1}^\beta[$, and $\{[p_k^\alpha, p_{k+1}^\alpha] \mid k \in \langle i, \ldots, i-1\rangle\} \cup \{[p_k^\beta, p_{k+1}^\beta] \mid k \in \langle j, \ldots, j-1\rangle\}$, if $p_i^\alpha = p_j^\beta$. Observe that if we are dealing with a curve intersection at an open segment, i.e., p_i^α belongs to $]p_j^\beta, p_{j+1}^\beta[$, then p_i^α appears twice in the definition of $\gamma_{\mathtt{cpc}}$. The wombliness of $\gamma_{\mathtt{cpc}}^\alpha$ and $\gamma_{\mathtt{cpc}}^\beta$ is summed up according to

$$\mathcal{H}(\gamma_{\mathtt{cpc}}) = \mathcal{H}(\gamma_{\mathtt{cpc}}^\alpha) + \mathcal{H}(\gamma_{\mathtt{cpc}}^\beta).$$

We are now ready to detail the two possible outcomes of a curve intersection:

Agent re-positioning. If $\mathcal{H}(\gamma_{\mathtt{cpc}}^\alpha)$ and $\mathcal{H}(\gamma_{\mathtt{cpc}}^\beta)$ have different signs, we only keep the curve whose wombliness is larger in absolute value. Without loss of generality, assume the curve we keep is $\gamma_{\mathtt{cpc}}^2$. Then, we re-position the agents in $\gamma_{\mathtt{cpc}}^1$ along the boundary of $\gamma_{\mathtt{cpc}}^2$. This process does not affect the value of the wombliness



(a) (b) (c)

Fig. 10. Robotic network of 10 agents evolving under the wombling coordination algorithm. (a) shows the initial configuration, (b) shows the robot trajectories, and (c) shows the final configuration. The spatial field is $Y(x_1, x_2) = e^{-(x_1+2)^2-x_2^2} + 1.25e^{-(x_1-2)^2-x_2^2}$. The gradient flow (8) first triggers an inside self-intersection that results in a curve splitting. After this, each of the new curves undergoes an inside self-intersection that result in agent re-positionings.

of γ^2_{cpc}, and can be made in an arbitrary way. Note that the absolute value of the wombliness of the resulting non-self-intersecting curve is strictly larger than the value of the wombliness of γ_{cpc}.

Curve merging. If $\mathcal{H}(\gamma^\alpha_{\mathsf{cpc}})$ and $\mathcal{H}(\gamma^\beta_{\mathsf{cpc}})$ have the same sign, then choosing only one curve would lead to a decrease in the value of the wombliness. Therefore, we consider their merge into the curve γ_{cpc}. If the intersection occurs on an open segment, we need to add one more agent to the network at the intersection location, according to the definition of γ_{cpc} above. After the merge, the curve γ_{cpc} evolves according to (8).

The case when multiple intersections occur at the same time can be dealt with in a similar fashion to the discussion in Section 5.1.

5.3 Convergence Analysis

We refer to the distributed hybrid control design described in Sections 5.1 and 5.2 as the *wombling coordination algorithm*. The next result follows from a simple application of LaSalle's Invariance principle [13].

Proposition 4. *Any network trajectory evolving under the wombling coordination algorithm that does not undergo curve-splitting or curve-merging transitions converges to a critical configuration of $\mathcal{H}_c$ while monotonically optimizing the total wombliness.*

From Proposition 4, we can deduce the following result for network trajectories that undergo curve-splitting and curve-merging transitions.

Corollary 5. *A network trajectory that undergoes a finite number of curve-splitting and curve-merging transitions monotonically optimizes the total wombliness. Moreover, the subnetworks that result after these transitions have taken place each converge to a critical configuration of $\mathcal{H}_c$.*

Remark 6. Note that no conditions are imposed in Corollary 5 on the number of agent re-positioning transitions. A similar result could be established for network trajectories that undergo an infinite number of curve-splitting and curve-merging transitions but are non-Zeno executions of the hybrid system [7,14]. •

Figures 9 and 10 present illustrations of the execution of the wombling coordination algorithm. The domain in all plots is $\mathcal{D} = [-4, 4] \times [-4, 4]$.

6 Conclusions

We have proposed a distributed coordination algorithm for robotic sensor networks that seek to detect areas of abrupt change of a spatial phenomena of interest. Our algorithm design has combined notions borrowed from statistical estimation and computer vision with tools from hybrid systems theory. The proposed algorithm allows for network splitting and re-grouping, and is guaranteed to monotonically increase the wombliness of the overall ensemble.

In order to make the proposed hybrid control design more amenable to implementation in practical scenarios, future work will address two limitations of the present approach. We need to move beyond the assumption that individual agents have gradient and Laplacian information on the spatial field along their immediate counterclockwise and clockwise boundary. When a curve merging or splitting occurs, the addition of an agent to the network can be done in a number of ways - e.g., individual agents might carry several smaller, lighter agents that can be deployed if needed. However, we need to better understand the number of switchings that can occur along the evolution, and provide conditions for their finiteness. We also plan to extend the present approach to open polygonal curves to detect "fronts" of abrupt change in the spatial phenomena.

Acknowledgments

This research was supported in part by NSF CAREER Award ECS-0546871.

References

1. Fagan, W.F., Fortin, M.J., Soykan, C.: Integrating edge detection and dynamic modeling in quantitative analyses of ecological boundaries. BioScience 53, 730–738 (2003)
2. Banerjee, S., Gelfand, A.E.: Bayesian wombling: Curvilinear gradient assessment under spatial process models. Journal of the American Statistical Association 101, 1487–1501 (2006)
3. Barbujani, G., Oden, N.L., Sokal, R.R.: Detecting areas of abrupt change in maps of biological variables. Systematic Zoology 38, 376–389 (1989)
4. Jacquez, G.M., Greiling, D.A.: Geographic boundaries in breast, lung, and colorectal cancers in relation to exposure to air toxins in long island, new york. International Journal of Health Geographics 2, 1–22 (2003)
5. Osher, S., Paragios, N. (eds.): Geometric Level Set Methods in Imaging, Vision, and Graphics. Springer, New York (2003)

6. Paragios, N., Chen, Y., Faugeras, O. (eds.): Handbook of Mathematical Models in Computer Vision. Springer, New York (2005)
7. van der Schaft, A.J., Schumacher, H.: An Introduction to Hybrid Dynamical Systems. Lecture Notes in Control and Information Sciences, vol. 251. Springer, Heidelberg (2000)
8. Liberzon, D.: Switching in Systems and Control. In: Systems & Control: Foundations & Applications, Birkhäuser, Basel (2003)
9. Hespanha, J.: Stabilization through hybrid control. In: Unbehauen, H. (ed.) Control Systems, Robotics, and Automation. Eolss Publishers, Oxford (2004)
10. Sanfelice, R.G., Goebel, R., Teel, A.R.: Invariance principles for hybrid systems with connections to detectability and asymptotic stability. IEEE Transactions on Automatic Control 52, 2282–2297 (2007)
11. Kimmel, R.: Fast edge integration. In: Osher, S., Paragios, N. (eds.) Geometric Level Set Methods in Imaging, Vision, and Graphics, pp. 59–78. Springer, Heidelberg (2003)
12. Courant, R., John, F.: Introduction to Calculus and Analysis II/2. Classics in Mathematics. Springer, New York (1999)
13. Khalil, H.K.: Nonlinear Systems. Prentice-Hall, Englewood Cliffs (2002)
14. Johansson, K.J., Egerstedt, M., Lygeros, J., Sastry, S.S.: On the regularization of Zeno hybrid automata. Systems & Control Letters 38, 141–150 (1999)

Epsilon-Tubes and Generalized Skorokhod Metrics for Hybrid Paths Spaces

J.M. Davoren

Department of Electrical & Electronic Engineering
The University of Melbourne, VIC 3010 Australia
davoren@unimelb.edu.au

Abstract. We develop several generalized Skorokhod pseudo-metrics for hybrid path spaces, cast in a quite general setting, where the basic open sets are epsilon-tubes around paths that, intuitively, allow for some "wiggle room" in both time and space via set-valued retiming maps between the time domains of paths. We then determine necessary and sufficient conditions under which these topologies are Hausdorff and their distance functions are metrics. On spaces of paths with closed time domains, our metric topology of generalized Skorokhod uniform convergence on finite prefixes is equivalent to the implicit topology of graphical convergence of hybrid paths, currently used extensively by Teel and co-workers.

1 Introduction

A basic problem in the foundations of hybrid systems is that of giving useful quantitative measures of closeness between trajectories that may differ in their time domains, in virtue of variations in timing of discrete transition events, or in their way of letting time "run to infinity"; for example, how do we compare a Zeno trajectory with one that exhibits finite-escape time after finitely-many discrete transitions? Topological – and preferably metric – structure on spaces of hybrid trajectories, and on spaces of paths discretely simulating or approximating hybrid trajectories, is a necessary prelude to addressing questions of robustness, or of the accuracy of discrete simulations or approximations.

One approach addressing several of these issues (proposed independently by Teel and co-workers in [1] and by Collins in [2], and employed in [3,4,5,6]) is to model the time domain of a hybrid path as a linearly-ordered subset of the partially-ordered structure $\mathbb{R} \times \mathbb{Z}$; the coordinate in $\mathbb{R}$ gives the "normal" time and the coordinate in $\mathbb{Z}$ is incremented with each discrete transition.[1] In developing topological structure on hybrid path spaces, the papers [1,3,4,6], and also [2,5], take an indirect route: the convergence of a sequence of hybrid paths is formulated in terms of the set-convergence of the graphs of those paths as subsets of $\mathbb{R} \times \mathbb{Z} \times \mathbb{R}^n$, with set-convergence as in [16]. A more direct approach is taken in [17,18] and also in [19], which use variants of the *Skorokhod metric* (originally from stochastic processes with right-continuous sample paths

[1] This approach is equivalent to the so-called "hybrid time trajectories" used in [7,8]. Two-dimensional time structures linearly-ordered by the lexicographic order are also used in earlier work on hybrid trajectories in the context of logics and formal methods for hybrid systems in [9,10,11,12,13], and in behavioural systems approaches to hybrid systems, in [14,15].

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 135–149, 2009.

[20]) to structure the space of infinite non-Zeno hybrid trajectories modeled as functions with real-time domain $\mathbb{R}^+ = [0, \infty)$. These Skorokhod-type metrics accommodate trajectories with different transition times by using retiming maps which are strictly order-preserving, bijective functions from one time domain to the other. Intuitively, Skorokhod-type metrics allow us to "wiggle space and time a bit" – in contrast with the topology of uniform convergence of continuous functions over a common time domain, which only allows us to "wiggle space a bit". However, a significant limitation of the original Skorokhod-type metrics (discussed in [19]) is that strictly order-preserving, single-valued retiming maps are too inflexible and restrictive in a hybrid setting.

The first contribution of the present paper is to develop generalized Skorokhod pseudo-metrics for hybrid path spaces in a quite general setting, and to determine necessary and sufficient conditions under which these topologies are Hausdorff and their distance functions are metrics. We start with spaces of finite-length paths (including those with finite-escape time), where the key notion is that of ε-tolerance relations which pair finite-length paths that can be viewed as ε-close via a set-valued retiming map between their domains; the generalized Skorokhod distance between two paths is then the infimum of all such ε for that pair of paths. We then extend up to spaces of arbitrary-length paths by considering the ε-closeness of longer and longer finite prefixes. The generalized Skorokhod distance between two arbitrary-length paths is given as an infinite sum weighted by 2^{-n} of the distances between length-n finite prefixes. For arbitrary-length paths, we identify two distinct topologies, that of generalized Skorokhod uniform convergence, and that of generalized Skorokhod uniform convergence on finite prefixes, and determine distinct metrics for them. For the Hausdorff property, we give an easy-to-satisfy sufficient condition, as well as a more technical necessary and sufficient condition to mark the limits of metrizability. We also show that, restricted to spaces of arbitrary-length paths with closed time domains, the implicit topology of graph-convergence for hybrid paths from [1,3,4] is equivalent to the weaker of the two generalized Skorokhod metrics. The metric and convergence notions developed here are illustrated on spaces of solution paths of hybrid systems, under the standing assumptions used by Teel and co. in [1,3,4] in addressing questions of asymptotic stability.

The paper is a substantial advance on [21], which introduces set-valued retiming maps in order to accommodate various hybrid phenomena, and uses them in developing several (2- and 3-parameter) uniform topologies on hybrid path spaces, but without developing a pseudo-metric or characterizing the Hausdorff property, as is done here.

On notation: we write $R\colon X \rightsquigarrow Y$ to mean R is a set-valued map, with (possibly empty) values $R(x) \subseteq Y$; *domain* $\mathrm{dom}(R) := \{x \in X \mid R(x) \neq \emptyset\}$; *inverse* $R^{-1}\colon Y \rightsquigarrow X$ with $x \in R^{-1}(y)$ iff $y \in R(x)$; and *range* $\mathrm{ran}(R) := \mathrm{dom}(R^{-1})$. We do not distinguish between a set-valued map and a relation/set of ordered pairs $R \subseteq X \times Y$. For any set $A \subseteq X$, the direct or post-image is the set $R(A) := \{y \in Y \mid R^{-1}(y) \cap A \neq \emptyset\}$. If a map R is a *partial function*, we write $R\colon X \dashrightarrow Y$ to mean R is single-valued on its domain, with values $R(x) = y$ (rather than $R(x) = \{y\}$). As usual, $R\colon X \to Y$ means R is single-valued with $\mathrm{dom}(R) = X$ and $\mathrm{ran}(R) \subseteq Y$. We write $\mathbb{R}^+$ for $[0, \infty)$, $\mathbb{R}^{>0}$ for $(0, \infty)$, $\mathbb{R}^{+\infty}$ for $\mathbb{R}^+ \cup \{\infty\}$, and $\mathbb{N}^{>0}$ for $\{n \in \mathbb{N} \mid n > 0\}$.

2 Time Structures and Their Topologies

A structure $(S, \leqslant, 0, +, -)$ is an *partially-ordered abelian group* [22] if $(S, \leqslant)$ is a partial order, $(S, 0, +, -)$ is an abelian group, and the strict ordering $<$ is *shift-invariant*: $s < t$ implies $s + r < t + r$, for all $s, t, r \in S$. An element $u > 0$ is called an *order-unit* for the partially-ordered group S if for every $s \in S$, there exists an $m \in \mathbb{N}^+$ (depending on s) such that $s \leqslant m\,u$, where integer multiplication is just iterated addition. An order-unit uniquely determines a *pseudo-norm* $\|\cdot\| \colon S \to \mathbb{R}^+$ that assigns $\|u\| = 1$ and is such that for all $s, t \in S$, if $t \geq 0$ and $-t \leqslant s \leqslant t$ then $\|s\| \leq \|t\|$. As first identified by Stone [23], the order-unit pseudo-norm $\|\cdot\|$ from u has the explicit description:

$$(\forall s \in S) \quad \|s\| \;:=\; \inf\left\{ \frac{m}{n} \in \mathbb{Q}^+ \;\middle|\; m, n \in \mathbb{N}^+ \;\wedge\; -m\,u \leqslant n\,s \leqslant m\,u \right\} . \qquad (1)$$

The pseudo-norm $\|\cdot\|$ is a norm (satisfying $\|s\| = 0$ iff $s = 0$, for all $s \in S$) when S is *archimedean*, which means that if $ks \leqslant t$ for all $k \in \mathbb{N}$, then $s \leqslant 0$.

Definition 1. [Time structures [21]]
A time structure $(S, \leqslant, 0, +, -, u)$ *is an archimedean partially-ordered abelian group with a distinguished order-unit $u > 0$ that determines an order-unit norm $\|\cdot\|$. A* future time structure *T is the positive cone of a time structure, so $T = S^+ := \{s \in S \mid 0 \leqslant s\}$ for some S. A time structure S is* finite-dimensional *iff for some integer $n \geq 1$, S is isomorphic with a partially-ordered abelian sub-group of $(\mathbb{R}^n, \mathbf{1}^n)$ with order-unit $\mathbf{1}^n = (1, 1, \ldots, 1)$ (hence S is lattice-ordered), where the embedding is a strictly order-preserving group isomorphism that is a continuous function w.r.t. the norm topologies and maps order-unit to order-unit and positive elements to positive elements.*

The continuous time structure $\mathbb{R}$ and the discrete time structure $\mathbb{Z}$ are both linearly-ordered abelian groups, and both are Dedekind-complete and archimedean; taking 1 as the order-unit gives the usual absolute-value $\|s\| = |s| = \max\{s, -s\}$. The basic hybrid time structure $\mathbb{Z}\times\mathbb{R}$ is a 2-dimensional abelian group with pair-wise addition and group identity $(0, 0)$, partially-ordered by the product order, $(i, t) \leqslant (i', t')$ iff $i \leq i'$ and $t \leq t'$; it is also Dedekind-complete and archimedean. The basic hybrid future time structure $\mathbb{H} := \mathbb{N} \times \mathbb{R}^+$ is the positive cone (and positive quadrant) of $\mathbb{Z} \times \mathbb{R}$. For the order-unit, we can take $u = (1, 1)$, and the Stone order-unit-norm is $\|(i, t)\| = \max\{|i|, |t|\}$. An equivalent norm, implicitly used in [3,4], is $\|(i, t)\|' := \frac{1}{2}(|i| + |t|)$, which satisfies $\frac{1}{2}\|(i, t)\| \leq \|(i, t)\|' \leq \|(i, t)\|$. For modeling and analysis of discrete-time simulations of hybrid systems, one uses $\mathbb{Z} \times \mathbb{Z}$, with future cone $\mathbb{N} \times \mathbb{N}$.

For each $r \in S$ in a time structure, the *r-shift function* $\sigma^r \colon S \to S$ is strictly order-preserving, where $\sigma^r(s) := s + r$ for all $s \in S$. In partial orders (as in linear orders) the basic sets are the *intervals* between points: sets $[a, b] := \{s \in S \mid a \leqslant s \leqslant b\}$ and $(a, b) := \{s \in S \mid a < s < b\}$; the *up-sets* above a given point: $[a\uparrow) := \{s \in S \mid a \leqslant s\}$ and $(a\uparrow) := \{s \in S \mid a < s\}$; symmetrically, the *down-sets* $(\downarrow a]$ and $(\downarrow a)$; and the *incomparability set*: $(a\perp) := S \setminus ([a\uparrow) \cup (a\downarrow])$, which is empty for all $a \in S$ iff the ordering is linear. In general, intervals, up-sets and down-sets are only partially-ordered.

In a time structure S with order-unit u, the *unit interval* is $[0, u]$, and the *granularity* of the norm $\|\cdot\|$ is defined by $\mathsf{gr}(S) := \inf\{\|s\| \in \mathbb{R}^+ \mid s \in (0, u]\}$. A time structure S

is *discrete* iff $\mathsf{gr}(S) > 0$, and is *dense* iff $\mathsf{gr}(S) = 0$. For example, $\mathbb{R}$, $\mathbb{Z} \times \mathbb{R}$, and $\mathbb{Q}_{\mathbb{B}} \times \mathbb{R}$ all have granularity 0, while $\mathbb{Z}$ and $\mathbb{Z} \times \mathbb{Z}$ have granularity 1.

On a time structure S, let $\mathcal{T}_{\leqslant}$ be the order topology on S which has as a basis the family $\mathcal{B}_{\leqslant}$ of all strict up-sets and down-sets, and their intersections, the strict open intervals. Let $\mathcal{T}_{\text{norm}}$ be the norm topology on S determined by $\|\cdot\|$ which has as a basis the family $\mathcal{B}_{\text{norm}}$ of all norm-balls $B_\delta(s) := \{t \in S \mid \|t - s\| < \delta\}$, for $s \in S$ and real $\delta > 0$; $\mathcal{T}_{\text{norm}}$ is also the coarsest topology on S w.r.t. which $\|\cdot\| : S \to \mathbb{R}^+$ is continuous. From [21], some key properties of finite-dimensional time structures S are as follows:

(1) The norm topology is refined by the order topology; that is: $\mathcal{T}_{\text{norm}} \subseteq \mathcal{T}_{\leqslant}$, with $\mathcal{T}_{\text{norm}} = \mathcal{T}_{\leqslant}$ if $\leqslant$ is a linear-ordering.

(2) For all $s, t \in S$, intervals $[s, t]$, up-sets $[s\uparrow)$, and down-sets $(s\downarrow]$, are closed in $\mathcal{T}_{\text{norm}}$; if $s \leqslant t$, then $[s, t]$ is compact in $\mathcal{T}_{\text{norm}}$.

(3) For any subset $A \subseteq S$, A is norm-bounded iff A is order-bounded; if S is also Dedekind-complete, then A is compact in $\mathcal{T}_{\text{norm}}$ iff A is closed and bounded in $\mathcal{T}_{\text{norm}}$.

3 Compact Paths and Their Maximal Extensions

Definition 2. [Compact time domains [21]]
Given a time structure S with future time T, let $\mathsf{Lin}(T)$ *be the set of all non-empty linearly-ordered subsets L of T; i.e. the partial-order $\leqslant$ restricted to L is a linear-order. A* compact time domain *in T is any set $L \in \mathsf{Lin}(T)$ such that $0 \in L$ and L is compact in $\mathcal{T}_{\text{norm}}$. Let* $\mathsf{CD}(T)$ *be the set of all compact time domains in T.*

If S is finite-dimensional and Dedekind-complete and $L \in \mathsf{Lin}(T)$, then $L \in \mathsf{CD}(T)$ iff L contains 0, $L \subset [0, t]$ for some $t \in T$ and L is closed $\mathcal{T}_{\text{norm}}$. As a special case, all finite sample-time sets $L = \{0, t_1, \ldots, t_{N-1}\}$ are compact time domains. For any L in $\mathsf{CD}(T)$, either L is a single linearly-ordered and densely-ordered subset of T (including the one-point set $\{0\} = [0, 0]$), or else there exist one or more pairs of *discrete-successor points* $t_i, t_i' \in L$ such that $t_i < t_i'$ and $(t_i, t_i') \cap L = \varnothing$.

Definition 3. [Compact continuous paths [21]]
Given a time structure S with future time T, let the signal value-space be a non-empty metric space (X, d_X). Define the set $\mathsf{CP}(T, X)$ *of* compact continuous T-paths in X *by:*

$$\mathsf{CP}(T, X) := \{\gamma \colon T \dashrightarrow X \mid \mathrm{dom}(\gamma) \in \mathsf{CD}(T) \ \wedge \ \gamma \textit{ is continuous on } \mathrm{dom}(\gamma)\}.$$

For $\gamma \in \mathsf{CP}(T, X)$, define the end-time *of γ by $b_\gamma := \max(\mathrm{dom}(\gamma))$, and the* length *of γ by $\mathsf{len}(\gamma) := \|b_\gamma\|_T$. Define a partial-ordering on $\mathsf{CP}(T, X)$ using subset-inclusion (on sets of ordered pairs) and the partial-ordering on T: $\gamma < \gamma'$ iff $\gamma \subset \gamma'$ and $t < t'$ for all $t \in \mathrm{dom}(\gamma)$ and all $t' \in \mathrm{dom}(\gamma') \setminus \mathrm{dom}(\gamma)$, in which case the path γ' is a* strict extension *of γ, and γ is a* strict prefix *of γ'; as usual, $\gamma \leqslant \gamma'$ iff $\gamma < \gamma'$ or $\gamma = \gamma'$.*

Being continuous on compact domains, all paths $\gamma \in \mathsf{CP}(T, X)$ are uniformly continuous. From [21] (differing slightly from [12,13]), the following three operations on paths are well-defined partial functions on $\mathsf{CP}(T, X)$: for $\gamma \in \mathsf{CP}(T, X)$, $t \in T$ and $b_\gamma = \max(\mathrm{dom}(\gamma))$:

- the *t-prefix* $\gamma|_t$, with $\mathrm{dom}(\gamma|_t) := [0, t] \cap \mathrm{dom}(\gamma)$ and $\gamma|_t(s) := \gamma(s)$ for all $s \in \mathrm{dom}(\gamma|_t)$;
- the *t-suffix* ${}_t|\gamma$, which is defined only when $t \in \mathrm{dom}(\gamma)$, with $\mathrm{dom}({}_t|\gamma) := [0, b_\gamma - t] \cap \sigma^{-t}(\mathrm{dom}(\gamma))$ where ${}_t|\gamma(s) := \gamma(s + t)$ for all $s \in \mathrm{dom}({}_t|\gamma)$;

- the *t-fusion* $\gamma *_t \gamma'$, which is defined only when $t \in \mathrm{dom}(\gamma)$ and $\gamma(t) = \gamma'(0)$, and which has $\mathrm{dom}(\gamma *_t \gamma') := \mathrm{dom}(\gamma|_t) \cup \sigma^{+t}(\mathrm{dom}(\gamma'))$ and $(\gamma *_t \gamma')(s) := \gamma(s)$ if $s \in \mathrm{dom}(\gamma|_t)$ and $(\gamma *_t \gamma')(s) := \gamma'(s-t)$ if $s \in \sigma^{+t}(\mathrm{dom}(\gamma'))$.

This prefix operation is well-defined for *all* times $t \in T$, not just $t \in \mathrm{dom}(\gamma)$, and $\gamma|_t \preccurlyeq \gamma$ for all $t \in T$; in particular, $\gamma|_t \prec \gamma$ if $t \not\geqslant b_\gamma$, while $\gamma|_t = \gamma$ if $t \geqslant b_\gamma$. Moreover, for any $t \in T$, if $\|t\|_T > \mathsf{len}(\gamma)$, then $\gamma|_t = \gamma$. For all $t \in T$ and compact γ, the set $[0,t] \cap \mathrm{dom}(\gamma) = \mathrm{dom}(\gamma|_t)$ is compact and linearly-ordered, with maximum $t_0 = \max\{\, s \in \mathrm{dom}(\gamma) \mid s \leqslant t \,\}$. A set $P \subseteq \mathsf{CP}(T,X)$ is *prefix-closed* iff for all $\gamma \in P$ and all $t \in T$, the path $\gamma|_t \in P$. A set $P \subseteq \mathsf{CP}(T,X)$ is *deadlock-free* iff for all $\gamma \in P$, there exists $\gamma' \in P$ such that $\gamma \prec \gamma'$. From [12,13], a *general flow system* is a set-valued map $\Phi\colon X \rightsquigarrow \mathsf{CP}(T,X)$ such that for all $x \in \mathrm{dom}(\Phi)$, for all $\gamma \in \Phi(x)$, and all $t \in \mathrm{dom}(\gamma)$: (GF0) $x = \gamma(0)$; (GF1) suffix-closure ${}_t|\gamma \in \Phi(\gamma(t))$; and (GF2) fusion-closure $(\gamma *_t \gamma') \in \Phi(x)$ for all $\gamma' \in \Phi(\gamma(t))$.

We take finite-length compact paths as the basic objects precisely because in multi-dimensional time structures, there are multiple ways of "letting time go to infinity". However, for the asymptotic analysis of dynamics, as well as for the semantics of temporal logics of such systems [12,13] we do need to determine the *maximal extensions* of compact paths. When S is finite-dimensional, any $L \in \mathsf{Lin}(T)$ will have cardinality at most that of the reals, so we only need to consider extending sequences of paths of ordinal length at most ω_1, the first uncountable ordinal. Let CLO be the set of all countable limit ordinals ν with $\omega \leq \nu < \omega_1$, where ω is the ordinal length of $\mathbb{N}$. Given any set $P \subseteq \mathsf{CP}(T,X)$, and a $\nu \in \mathrm{CLO}$, a ν-length sequence $\{\gamma_m\}_{m<\nu}$ is a *P-chain* if $\gamma_m \prec \gamma_{m'}$ for all $m < m' < \nu$. The *asymptotic limit* of a P-chain is the partial function $\eta\colon T \dashrightarrow X$ such that $\eta = \bigcup_{m<\nu} \gamma_m$ (considered as sets of ordered-pairs), with the *length* $\mathsf{len}(\eta) := \sup_{m<\nu} \mathsf{len}(\gamma_m)$, possibly infinite.

Definition 4. [Limit extension and maximal extension of path sets [12,13]]
Let T be the future of a finite-dimensional time structure. For any set $P \subseteq \mathsf{CP}(T,X)$ of compact paths, define the limit extension $\mathsf{L}(P)$*, the* maximal extension $\mathsf{M}(P) \subseteq \mathsf{L}(P)$*, and the* maximal infinite-length extension $\mathsf{M}^\infty(P) \subseteq \mathsf{M}(P)$*, as follows:*

$$\mathsf{L}(P) := \{\, \eta \in [T \dashrightarrow X] \mid (\exists \nu \in \mathrm{CLO})\left(\exists\, \overline{\boldsymbol{\gamma}} \in [\nu \to \mathsf{CP}(T,X)]\right)(\forall m < \nu)$$
$$\gamma_m := \overline{\boldsymbol{\gamma}}(m) \in P \ \wedge\ (\forall m' < \nu)\,(m < m' \ \Rightarrow\ \gamma_m \prec \gamma_{m'}) \ \wedge\ \eta = \textstyle\bigcup_{m<\nu} \gamma_m \,\};$$
$$\mathsf{M}(P) := \{\, \eta \in \mathsf{L}(P) \mid (\forall \gamma \in P)\ \eta \not\prec \gamma \,\} \quad \textit{and} \quad \mathsf{M}^\infty(P) := \{\, \eta \in \mathsf{M}(P) \mid \mathsf{len}(\eta) = \infty \,\}.$$

A set of compact paths P is called maximally-extendible *iff for all $\gamma \in P$, there exists $\eta \in \mathsf{M}(P)$ such that $\gamma \prec \eta$, and P is* forward-complete *iff P is maximally-extendible and $\mathsf{M}(P) = \mathsf{M}^\infty(P)$. Set $\mathsf{LCP}(T,X) := \mathsf{L}(\mathsf{CP}(T,X))$.*

The extension partial order on compact paths readily extends to limit paths: $\eta \prec \eta'$ iff $\eta \subset \eta'$ and $t \prec t'$ for all $t \in \mathrm{dom}(\eta)$ and $t' \in \mathrm{dom}(\eta') \setminus \mathrm{dom}(\eta)$. The prefix, suffix and fusion operations also extend to limit paths in the straight-forward way, with the strict prefix of a limit path always a compact path. It is also readily established that every limit path $\eta \in \mathsf{LCP}(T,X)$ is continuous (but may fail to be uniformly continuous).

Given a general flow system $\Phi\colon X \rightsquigarrow \mathsf{CP}(T,X)$, the *maximal extension* of Φ is the set-valued map $\mathsf{M}\Phi\colon X \rightsquigarrow \mathsf{LCP}(T,X)$ given by $\mathsf{M}\Phi(x) := \mathsf{M}(\Phi(x))$ for all $x \in X$,

with $\mathsf{M}\Phi(x) = \emptyset$ if $x \notin \mathrm{dom}(\Phi)$. A general flow Φ is *maximally-extendible* (*forward-complete*) iff for all $x \in \mathrm{dom}(\Phi)$, the path set $\Phi(x)$ is maximally-extendible (forward-complete). From [12,13], a core result (using the Axiom of Choice/Zorn's Lemma) is that a set of paths $P \subseteq \mathsf{CP}(T,X)$ is maximally-extendible iff P is deadlock-free, and hence, for general flow $\Phi\colon X \rightsquigarrow \mathsf{CP}(T,X)$, Φ is maximally-extendible iff Φ is deadlock-free.

We will subsequently be interested in: $\mathsf{CP}^\star(T,X) := \mathsf{CP}(T,X) \cup \mathsf{LCP}(T,X)$, the combined path set of both compact and limit continuous paths under the path-extension ordering, of finite or infinite length, and also the distinguished subsets:

$$\begin{aligned}\mathsf{CP}^\star_{\mathrm{cl}}(T,X) &:= \mathsf{CP}(T,X) \;\cup\; \{\,\eta \in \mathsf{LCP}(T,X) \mid \mathrm{dom}(\eta) \textit{ is norm-closed in } T\,\}\\ \mathsf{CP}^\star_{\mathrm{fin}}(T,X) &:= \{\,\eta \in \mathsf{CP}^\star(T,X) \mid \mathsf{len}(\eta) < \infty\,\}\\ &= \mathsf{CP}(T,X) \;\cup\; (\mathsf{CP}^\star(T,X) \setminus \mathsf{CP}^\star_{\mathrm{cl}}(T,X))\,.\end{aligned}$$

The basic fact being used here is that a path $\eta \in \mathsf{CP}^\star(T,X) \setminus \mathsf{CP}^\star_{\mathrm{cl}}(T,X)$ exactly when η is a limit path with $\mathrm{dom}(\eta)$ failing to be norm-closed, which is the case if and only if $\mathrm{dom}(\eta)$ is norm-bounded with finite length. Given a set of compact paths $P \subseteq \mathsf{CP}(T,X)$, we say a limit path $\eta \in \mathsf{L}(P)$ has *finite-escape time* w.r.t. P iff $\eta \in \mathsf{M}(P)$ and $\mathsf{len}(\eta) < \infty$, and so $\eta \notin \mathsf{M}^\infty(P)$, and $\mathrm{dom}(\eta)$ will be norm-bounded but not norm-closed in T.

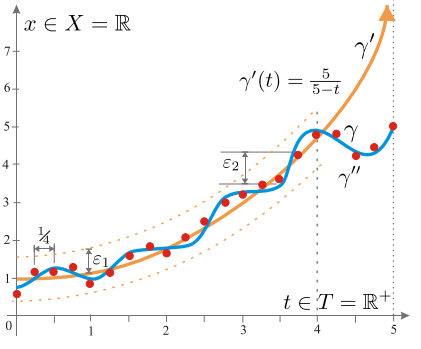


Fig. 1. Three finite-length real-time paths, with differing time domains

Example 1. Consider three finite-length paths $\gamma, \gamma', \gamma'' \in Z \subseteq \mathsf{CP}^\star_{\mathrm{fin}}(T,X)$ in Figure 1 where $T = \mathbb{R}^+$ and $X = \mathbb{R}$, and P the set of all $\gamma \in \mathsf{CP}(T,X)$ with either $\mathrm{dom}(\gamma) = [0,b]$, or $\mathrm{dom}(\gamma) = \{0, s_1, s_2, \ldots, s_N\}$, and $Z = (P \cup \mathsf{M}(P)) \cap \mathsf{CP}^\star_{\mathrm{fin}}(T,X)$. Here, γ is a compact path with $\mathrm{dom}(\gamma) = [0,5]$, while γ' is a limit path in $\mathsf{M}(P)$ having $\mathrm{dom}(\gamma') = [0,5)$ and $\gamma'(t) = \frac{5}{5-t}$ for all $t \in [0,5)$, with escape to infinity at time 5. The third path γ'' is also compact (and uniformly continuous!), with $\mathrm{dom}(\gamma'') = \{\,\frac{k}{4} \mid 0 \le k \le 20\,\}$, giving a time-sampling of the interval $[0,5]$ with a (rather coarse) sampling period $d = \frac{1}{4}$.

4 Path Spaces and General Flows of Hybrid Systems

For a fixed metric space X, let $P_{\mathrm{hyb}}(X) \subset \mathsf{CP}(\mathbb{H},X)$ be the set of regular compact hybrid paths γ whose time domains within $\mathbb{H} = \mathbb{N} \times \mathbb{R}^+$ are finite unions of the form

$\mathrm{dom}(\gamma) = \bigcup_{i<m} \{i\} \times [s_i, s_{i+1}] \cup \{m\} \times [s_m, b_\gamma]$, where $m \in \mathbb{N}$ in the number of discrete transitions, $s_0 := 0$ and for each $i < m$, $s_{i+1} \in \mathbb{R}^+$ is the real time at the $(i+1)^{\text{st}}$ switching or discrete transition, with $s_{i+1} \geq s_i$. For maximal paths $\eta \in \mathsf{M}(P_{\text{hyb}}(X))$, we have $\mathsf{len}(\eta) < \infty$ iff $\mathrm{dom}(\eta)$ fails to be norm-closed and $\mathrm{dom}(\eta) \subset [(0,0),(i,c))$ for some $(i,c) \in \mathbb{H}$, which will be the case exactly when the last continuous time evolution has finite-escape time. A hybrid limit path $\eta \in \mathsf{M}^\infty(P_{\text{hyb}}(X))$ is *Zeno* iff $\mathsf{len}(\eta) = \infty$ and $\mathrm{dom}(\eta) \subset \mathbb{N} \times [0,c)$ for some $c < \infty$, in which case the length of η is infinite but the total real-time duration is finite and bounded by c. The non-Zeno infinite-length hybrid paths are those of infinite real-time duration, and such paths $\eta \in \mathsf{M}^\infty(P_{\text{hyb}}(X))$ may have either an infinite or a finite number of discrete transitions; in the latter case, $\mathrm{dom}(\eta) = \bigcup_{i<m}\{i\}\times[s_i, s_{i+1}] \cup \{m\} \times [s_m, \infty)$ for some $m \in \mathbb{N}^{>0}$, while in the former case, $\mathrm{dom}(\eta) = \bigcup_{i\in\mathbb{N}}\{i\} \times [s_i, s_{i+1}]$.

Formulated within the framework of differential and difference inclusions [3,7], a *hybrid system* is a structure $H = (X, F, G, C, D)$ where:

- $X \subseteq \mathbb{R}^n$ is the state space, with $(C \cup D) \subseteq X$;
- $F\colon X \rightsquigarrow \mathbb{R}^n$ describes the continuous dynamics $\dot{x} \in F(x)$;
- $G\colon X \rightsquigarrow X$ describes the discrete dynamics $x' \in G(x)$;
- $C \subseteq (X \cap \mathrm{dom}(F))$ is the region of continuous flow; and
- $D \subseteq (X \cap \mathrm{dom}(G))$ is the discrete switching, jump or transition guard region.

The trajectories of H determine a prefix-closed general flow $\Phi_H\colon X \rightsquigarrow \mathsf{CP}(\mathbb{H}, X)$ such that a compact-domain hybrid path $\gamma \in \Phi_H(x)$ exactly when: (i) $x \in \mathrm{dom}(\Phi_H) := C \cup D$, and $x = \gamma(0,0)$; (ii) $\gamma \in P_{\text{hyb}}(X)$ is a regular hybrid path, with end-time $(m, b_\gamma) := \max(\mathrm{dom}(\gamma))$, and switching times $\{s_{i+1}\}_{i\leq m}$ with $s_0 = 0$; (iii) for each $(i,t) \in \mathrm{dom}(\gamma)$, (a) if $t = s_{i+1}$, a switching time, then $\gamma(i,t) \in D$, and $\gamma(i+1,t) \in G(\gamma(i,t))$, and (b) if $i < m$ and $t \in [s_i, s_{i+1}]$, or if $i = m$ and $t \in [s_m, b_\gamma]$, then $\gamma(i,t) \in C$ and $\frac{d}{d\tau}\gamma(i,\tau) \in F(\gamma(i,\tau))$ for almost all $\tau \in [s_i, s_{i+1}]$, taking $s_{m+1} := b_\gamma$ when $i = m$, where the real-time curve segment $\xi_i\colon [s_i, s_{i+1}] \to X$ given by $\xi_i(\tau) := \eta(i,\tau)$ for all $\tau \in [s_i, s_{i+1}]$, is absolutely continuous on the interval $[s_i, s_{i+1}]$.

If any of the vector coordinates, say x_1 of $x \in X$, is designated *discrete*, as is the case for the locations in *hybrid automata*, then the first component $F_1\colon X \rightsquigarrow \mathbb{R}$ has $F_1(x) = \{0\}$ for all $x \in C$, and $x_1 \in Q$ for all $x \in C \cup D$, with Q a finite subset of $\mathbb{R}$, so that x_1 only changes value under G. If x_j is an (accurate) *clock*, then $F_j(x) = \{1\}$.

5 Generalized Skorokhod Topologies and Metrics on Path Spaces

When two paths η and η' in $\mathsf{CP}(T,X)$ or $\mathsf{LCP}(T,X)$ have *the same time domain*, we can use the metric d_x on X to determine if they spatially ε-close for their whole length by taking $d_\infty(\eta,\eta') := \sup_{t\in L} d_x(\eta(t),\eta'(t))$ for $L = \mathrm{dom}(\eta) = \mathrm{dom}(\eta')$. The infinity-metric d_∞ intuitively allows for some "wiggle in space" between the paths η and η'. In order to compare paths with *different time domains*, we need a notion of *retimings* between the time domains of paths, that allow for some "wiggle in time" as well as in space.

The *Skorokhod metric* allows for the comparison of real-time piecewise-continuous signals with differing points of discontinuity by using retimings that are *strictly order-preserving* functions between the time domains. Let $\mathsf{SRet}(\mathbb{R}^+)$ be the set of all strictly order-preserving and surjective $\rho : [0,b] \to [0,b']$, for $b, b' \in \mathbb{R}^+$, and for each $\rho \in$

$\mathsf{SRet}(\mathbb{R}^+)$, the *temporal deviation* is $\mathsf{dev}(\rho) := \sup_{t\in\mathrm{dom}(\rho)} |\, t - \rho(t)\,|$, possibly infinite, and applied to two signals with $\mathrm{dom}(\eta) = [0, b]$ and $\mathrm{dom}(\eta') = [0, b']$, the *spatial variation* is $\mathsf{var}(\eta, \eta', \rho) := \sup_{t\in\mathrm{dom}(\rho)} d_X(\eta(t), \eta'(\rho(t)))$, possibly infinite. For two finite-length interval-domain paths $\eta, \eta' : \mathbb{R}^+ \dashrightarrow X$ (one of several variants of) the Skorokhod distance between them is:

$$d_{\mathrm{Skor}}(\eta, \eta') := \begin{cases} \infty & \text{if there does not exist } \varepsilon \in \mathbb{R}^{>0} \text{ and } \rho \in \mathsf{SRet}(\mathbb{R}^+) \\ & \quad \text{such that } \mathsf{dev}(\rho) < \varepsilon \ \wedge\ \mathsf{var}(\eta, \eta', \rho) < \varepsilon\,, \\ \inf\{\,\varepsilon > 0 \mid (\exists \rho \in \mathsf{SRet}(\mathbb{R}^+))\ \mathsf{dev}(\rho) < \varepsilon \ \wedge\ \mathsf{var}(\eta, \eta', \rho) < \varepsilon\,\} & \\ & \quad \text{otherwise.} \end{cases} \tag{2}$$

The limitation of the Skorokhod metric when time T is hybrid is that too often, there will not be any strictly order-preserving functions between the domains of "close" paths. Non-strictly order-preserving single-valued maps are not invertible, so symmetry is lost. This motivates our relaxation to retiming maps that are order-preserving in a set-valued sense, are readily invertible and composable (like bijections), and include all order-preserving single-valued maps, strict and non-strict (the latter with set-valued inverses).

Definition 5. [The earlier-than relation on linearly-ordered sets, and retimings [21]] *Given a time structure S with future time T, the* earlier-than *relation $\trianglelefteq$ on the set $\mathsf{Lin}(T)$ of non-empty linearly-ordered subsets of T, is defined by:*

$$L \trianglelefteq L' \quad \Leftrightarrow \quad (\forall t \in (L \setminus L'))(\forall t' \in L')\ \ t < t' \ \wedge\ (\forall t \in L)(\forall t' \in (L' \setminus L))\ \ t < t'\,.$$

for all $L, L' \in \mathsf{Lin}(T)$. A set-valued map $\rho\colon T \rightsquigarrow T$ will be called order-preserving *iff $t_1 < t_2$ implies $\rho(t_1) \trianglelefteq \rho(t_2)$, for all $t_1, t_2 \in \mathrm{dom}(\rho)$. Given sets $L, L' \in \mathsf{Lin}(T)$, a set-valued map $\rho\colon T \rightsquigarrow T$ will be called a* retiming *from L to L' iff the following hold:*

(i) $\mathrm{dom}(\rho) = L$ *and* $\mathrm{ran}(\rho) = L'$*;*
(ii) *for all $t \in L$, $\rho(t) \in \mathsf{Lin}(T)$, and for all $t' \in L'$, $\rho^{-1}(t') \in \mathsf{Lin}(T)$; and*
(iii) *ρ and ρ^{-1} are both order-preserving.*

For a retiming $\rho\colon L \rightsquigarrow L'$, define the deviation $\mathsf{dev}(\rho) \in \mathbb{R}^{+\infty}$ *as follows:*

$$\mathsf{dev}(\rho) \ := \ \sup\{\,\|t - s\| \in \mathbb{R}^+ \mid t \in \mathrm{dom}(\rho) \ \wedge\ s \in \rho(t)\,\}\,.$$

Let $\mathsf{Ret}(L, L')$ denote the set of all retimings $\rho\colon L \rightsquigarrow L'$ together with all retimings $\rho'\colon L' \rightsquigarrow L$, so that $\mathsf{Ret}(L, L') = \mathsf{Ret}(L', L)$.

The key facts from [21] are: $\trianglelefteq$ is a partial-order on $\mathsf{Lin}(T)$; $\mathsf{Ret}(T)$ is closed under relational inverses and compositions of retimings, with $\mathsf{dev}(\rho^{-1}) = \mathsf{dev}(\rho)$ and $\mathsf{dev}(\rho \circ \rho') \leq \mathsf{dev}(\rho) + \mathsf{dev}(\rho')$. In [21], we worked with a finer 2-parameter uniform structure on the space $\mathsf{CP}(T, X)$, with one parameter $\delta \in \mathbb{R}^{>0}$ bounding the temporal deviation $\mathsf{dev}(\rho)$ and the second $\varepsilon \in \mathbb{R}^{>0}$ bounding the spatial variation $\mathsf{var}(\gamma, \gamma', \rho)$. Here, we work on the larger space $\mathsf{CP}^\star_{\mathrm{fin}}(T, X)$ of all finite-length continuous paths, and with a view to developing a pseudo-metric and metric, we combine those two parameters into one by effectively taking their maximum. For the rest of the paper, we assume the time structure S is finite-dimensional, and (X, d_X) is a metric space.

Definition 6. [Uniform relations and generalized Skorokhod distance: finite-length] *Let $Z \subseteq \mathsf{CP}^\star_{\mathrm{fin}}(T, X)$ be any set of finite-length paths. For each pair $(\gamma, \gamma') \in Z \times Z$, let $\mathsf{Ret}(\gamma, \gamma') := \mathsf{Ret}(\mathrm{dom}(\gamma), \mathrm{dom}(\gamma'))$, and let $\mathsf{Ret}(Z)$ be the union of all $\mathsf{Ret}(\gamma, \gamma')$ for*

$\gamma, \gamma' \in Z$. *Then define the* variation function $\mathsf{var}\colon (Z \times Z \times \mathsf{Ret}(Z)) \to \mathbb{R}^{+\infty}$ *such that* $\mathsf{var}(\gamma, \gamma', \rho) := \infty$ *if* $\rho \notin \mathsf{Ret}(\gamma, \gamma')$, *and otherwise,* $\mathsf{var}(\gamma', \gamma, \rho^{-1}) = \mathsf{var}(\gamma, \gamma', \rho)$, *and assuming that* $\mathrm{dom}(\rho) = \mathrm{dom}(\gamma)$ *and* $\mathrm{ran}(\rho) = \mathrm{dom}(\gamma')$, *we have:*

$$\mathsf{var}(\gamma, \gamma', \rho) \;:=\; \sup\{\, d_x(\gamma(t), \gamma'(t')) \mid t \in \mathrm{dom}(\gamma) \wedge t' \in \mathrm{dom}(\gamma') \wedge (t, t') \in \rho \,\}.$$

For each strictly positive real $\varepsilon \in \mathbb{R}^{>0}$, *define the relation* $V_\varepsilon\colon Z \rightsquigarrow Z$ *as follows:*

$$V_\varepsilon \;:=\; \{\, (\gamma, \gamma') \in Z \times Z \mid (\exists \rho \in \mathsf{Ret}(\gamma, \gamma'))\ \mathsf{dev}(\rho) < \varepsilon \ \wedge\ \mathsf{var}(\gamma, \gamma', \rho) < \varepsilon \,\}.$$

The finite-length-paths generalized Skorokhod distance *function* $d_{\mathrm{fgS}} : Z \times Z \to \mathbb{R}^{+\infty}$ *is defined for all* $\gamma, \gamma' \in Z$ *by:*

$$d_{\mathrm{fgS}}(\gamma, \gamma') \;:=\; \begin{cases} \inf\{\, \varepsilon \in \mathbb{R}^{>0} \mid (\gamma, \gamma') \in V_\varepsilon \,\} & \text{if } (\exists \varepsilon \in \mathbb{R}^{>0})\ (\gamma, \gamma') \in V_\varepsilon \\ \infty & \text{otherwise.} \end{cases} \tag{3}$$

As with the original Skorokhod metric $\mathsf{var}(\gamma, \gamma', \rho)$ bounds the "wiggle in space" variation between γ and γ' under a retiming ρ, while $\mathsf{dev}(\rho)$ bounds the "wiggle in time" allowed by ρ. The (reflexive, symmetric) relation V_ε is one of *ε-tolerance* between paths γ and γ', and the *ε-tube* $V_\varepsilon(\gamma)$ around γ is the set of all paths $\gamma' \in Z$ that are ε-close, and contains only paths of length within ε of that of γ. For brevity, we will usually write "gS-" for the adjectival phrase "generalized Skorokhod", and "fgS-" for "finite-length-paths generalized Skorokhod".

Proposition 1. [Generalized Skorokhod uniform topology on finite-length paths] *Let* $Z \subseteq \mathsf{CP}^\star_{\mathrm{fin}}(T, X)$ *be any set of finite-length continuous paths. For all* $\varepsilon, \varepsilon_1, \varepsilon_2 \in \mathbb{R}^{>0}$*:*

$$V_{\varepsilon_1} \subseteq V_{\varepsilon_2} \text{ when } \varepsilon_1 \le \varepsilon_2 \qquad V_\varepsilon \subseteq V_{\varepsilon_1} \cap V_{\varepsilon_2} \text{ when } \varepsilon \le \min\{\varepsilon_1, \varepsilon_2\}$$

$$V_{\varepsilon_1} \circ V_{\varepsilon_2} \subseteq V_\varepsilon \text{ when } \varepsilon_1 + \varepsilon_2 \le \varepsilon \qquad V_\varepsilon \circ V_\varepsilon \subseteq V_{\varepsilon_1} \text{ when } \varepsilon \le \tfrac{1}{2}\varepsilon_1\,,$$

and for all $\gamma, \gamma' \in Z$, $d_{\mathrm{fgS}}(\gamma, \gamma') < \varepsilon$ *iff* $(\gamma, \gamma') \in V_\varepsilon$.
Hence the family $\mathcal{V}_{\mathrm{fgS}} := \{\, V_\varepsilon\colon Z \rightsquigarrow Z \mid \varepsilon \in \mathbb{R}^{>0} \,\}$ *constitutes a basis for a uniformity on the path set Z, and the* fgS-uniform topology $\mathcal{T}_{\mathrm{fgS}}$ *generated by* $\mathcal{V}_{\mathrm{fgS}}$ *has as its basic open sets the ε-tubes* $V_\varepsilon(\gamma)$ *around paths* $\gamma \in Z$. *Furthermore, the fgS-distance function* $d_{\mathrm{fgS}} : Z \times Z \to \mathbb{R}^{+\infty}$ *is a pseudo-metric, and the topology generated by* d_{fgS} *is the same as the uniform topology* $\mathcal{T}_{\mathrm{fgS}}$.

Example 1 revisited. (See Fig. 1) For the example of the compact path γ with $\mathrm{dom}(\gamma) = [0, 5]$ and the spatially-unbounded limit path γ' with $\mathrm{dom}(\gamma') = [0, 5)$, the fgS-distance d_{fgS} comes out as $d_{\mathrm{fgS}}(\gamma, \gamma') = \infty$ because the distance $d_x(\gamma(5), \gamma'(t'))$ becomes arbitrarily large as $t' \to 5$, so no retiming of finite variation exists. However, from Fig. 1, the prefixes $\gamma|_4$ and $\gamma'|_4$ are quite close, with $d_{\mathrm{fgS}}(\gamma|_4, \gamma'|_4) < \varepsilon_1$ witnessed by the identity retiming; to be concrete, take $\varepsilon_1 \le 0.65$. To determine the fgS-distance between the (coarsely) sampled+quantized path γ'', and the original γ, three quantities come into play: (a) the sampling period, here $d = 0.25$; (b) the quantization error, here bounded by 0.2; and (c) the quantity labeled ε_2 in Fig. 1 from the uniform continuity of γ, such that for all $t, s \in \mathrm{dom}(\gamma)$, if $|t - s| \le 0.25$ then $d_{\mathbb{R}}(\gamma(t), \gamma(s)) \le \varepsilon_2$. Say $\varepsilon_2 \le 0.75$. The sampling retiming map $\rho_d : \mathrm{dom}(\gamma) \rightsquigarrow \mathrm{dom}(\gamma'')$ is given by $\rho_d(0) := \{0\}$ and $\rho_d(t) := \{\frac{k+1}{4}\}$ for all $t \in (\frac{k}{4}, \frac{k+1}{4}]$ and $k < 20$, so that $\mathsf{dev}(\rho_d) = d = 0.25$. Via the triangle inequality, the retiming ρ_d gives $d_{\mathrm{fgS}}(\gamma, \gamma'') \le \varepsilon_2 + 0.2 \le 0.95$.

Having established we have a uniform topology generated by ε-tubes, the further, more substantial task, is to identify sets $Z \subseteq \mathsf{CP}^\star_{\mathrm{fin}}(T, X)$ of finite-length paths for which

this uniform topology is Hausdorff, as in this case, the fgS-pseudo-metric d_{fgS} is actually a metric. For a set $Z \subseteq \mathsf{CP}^\star(T,X)$ of arbitrary-length paths, we call Z *highly discerning* iff $Z \subseteq (P \cup \mathsf{M}(P))$ for some set $P \subseteq \mathsf{CP}(T,X)$ of compact paths. In particular, Z contains no limit paths $\gamma \in \mathsf{L}(P) \setminus \mathsf{M}(P)$ that are not maximal w.r.t. P. We will show that the highly discerning property is sufficient for the Hausdorff property. In seeking a necessary and sufficient characterization of the Hausdorff property, we weaken the condition on path sets $Z \subseteq \mathsf{CP}^\star(T,X)$ to isolate the problem cases. Call a set Z *discerning* iff for all paths $\gamma, \gamma' \in Z$, if $\gamma < \gamma'$ and $\gamma \notin \mathsf{CP}(T,X)$, then the set difference $\mathrm{dom}(\gamma') \setminus \mathrm{dom}(\gamma)$ is not a singleton set. The fact that, for a set Z of finite-length paths, highly discerning implies discerning, will be a corollary of the following main result. Note that both properties are trivially satisfied by all sets $Z \subseteq \mathsf{CP}^\star(T,X) = \mathsf{CP}(T,X) \cup \mathsf{M}^\infty(\mathsf{CP}(T,X))$ if T is discrete, and by all sets $Z \subseteq \mathsf{CP}^\star_{\text{cl}}(T,X)$, for arbitrary T.

Proposition 2. [Properties of fgS-uniform topology and pseudo-metric]
Let $Z \subseteq \mathsf{CP}^\star_{\text{fin}}(T,X)$ be equipped with the uniform topology $\mathcal{T}_{\text{fgS}}$.

1. *The topology $\mathcal{T}_{\text{fgS}}$ on Z has a countable sub-basis for its uniformity.*
2. *The topology $\mathcal{T}_{\text{fgS}}$ on Z is Hausdorff if Z is highly discerning.*
3. *The topology $\mathcal{T}_{\text{fgS}}$ on Z is Hausdorff if and only if Z is discerning.*
4. *The topology $\mathcal{T}_{\text{fgS}}$ on Z is Hausdorff if and only if the fgS-pseudo-metric d_{fgS} is a an extended-valued metric on Z.*
5. *If the topology $\mathcal{T}_{\text{fgS}}$ on Z is Hausdorff, then for all sequences $\{\gamma_k\}_{k\in\mathbb{N}}$ in Z and all paths $\gamma \in Z$, $\gamma = \lim_{k\to\infty} \gamma_k$ iff $\lim_{k\to\infty} d_{\text{fgS}}(\eta, \eta_k) = 0$.*
6. *Restricted to the subset $P := Z \cap \mathsf{CP}(T,X)$ of compact paths, the uniform topology $\mathcal{T}_{\text{fgS}}$ on P is always Hausdorff, and the fgS-metric is always finite-valued.*

The difficult part of the proof of Proposition 2 is *Part 3*, in establishing that the discerning property is sufficient for the topology to be Hausdorff. Most parts of the proof make essential use of the paths being continuous on their domains.

In "lifting up" the uniform structure of the V_ε relations on finite-length paths, in order to use it on spaces $Z \subseteq \mathsf{CP}^\star(T,X) = \mathsf{CP}(T,X) \cup \mathsf{LCP}(T,X)$ of paths of arbitrary length, the key idea is that since a limit path is just the union of a chain of longer and longer compact prexes, we should look at closeness of longer and longer compact prexes. This motivates the introduction of a second parameter $\nu \in \mathbb{R}^+$ which references the length up to which two paths are ε-close. (In [21], we used a time position parameter $t \in T$, which turned out to be sub-optimal when looking for a metric). As parameter sets, let $A_2 := \mathbb{R}^{>0} \times \mathbb{R}^+$ and $A_2^\infty = \mathbb{R}^{>0} \times \mathbb{R}^{+\infty} = A_2 \cup \{(\varepsilon, \infty) \mid \varepsilon \in \mathbb{R}^{>0}\}$. We need the following key technical result.

Proposition 3. [Path operations within fgS-uniform topology]
For any paths $\gamma, \gamma' \in \mathsf{CP}^\star_{\text{fin}}(T,X)$ and for any parameter $\varepsilon \in \mathbb{R}^{>0}$,
if $d_{\text{fgS}}(\gamma,\gamma') < \varepsilon$ with witness $\rho \in \mathsf{Ret}(\gamma,\gamma')$ with $\mathsf{dev}(\rho) < \varepsilon$ and $\mathsf{var}(\gamma,\gamma',\rho) < \varepsilon$,
then for all pairs of time points $(t,t') \in \rho$ related by ρ, we have $d_{\text{fgS}}(\gamma|_t, \gamma'|_{t'}) < \varepsilon$, $d_{\text{fgS}}({}_t|\gamma, {}_{t'}|\gamma') < \varepsilon$, and for all $\eta, \eta' \in \mathsf{CP}^\star_{\text{fin}}(T,X)$, $d_{\text{fgS}}(\gamma *_t \eta, \gamma' *_{t'} \eta') < \varepsilon$ if $\gamma(t) = \eta(0)$, $\gamma'(t') = \eta'(0)$ and $d_{\text{fgS}}(\eta,\eta') < \varepsilon$.

Definition 7. [Uniform relations and gS-distances: arbitrary-length]
Let $Z \subseteq \mathsf{CP}^\star(T,X)$ be any set of continuous paths of arbitrary length, and for each pair $(\varepsilon,\nu) \in A_2$, let $U_{\varepsilon,\nu} : Z \rightsquigarrow Z$ be the relation defined as follows:

$$U_{\varepsilon,v} := \{\, (\eta,\eta') \in Z \times Z \mid \Big(\max\{\,\mathsf{len}(\eta),\, \mathsf{len}(\eta')\,\} \le v+\varepsilon \;\wedge\; d_{\mathrm{fgS}}(\eta,\eta') < \varepsilon\Big) \\ \vee\; ((\exists\, t \in \mathrm{dom}(\eta))(\exists\, t' \in \mathrm{dom}(\eta')) \\ \min\{\,\|\,t\,\|_T,\, \|\,t'\,\|_T\,\} \ge v \;\wedge\; d_{\mathrm{fgS}}(\eta|_t, \eta'|_{t'}) < \varepsilon\Big)\Big\}$$

and for each $\varepsilon \in \mathbb{R}^{>0}$, *let* $U_{\varepsilon,\infty} \colon Z \rightsquigarrow Z$ *be the relation defined by:*

$$U_{\varepsilon,\infty} \;:=\; \textstyle\bigcap_{v\in\mathbb{R}^+} U_{\varepsilon,v} \;=\; \{\, (\eta,\eta') \in Z\times Z \mid (\forall v \in \mathbb{R}^+)\;\; (\eta,\eta') \in U_{\varepsilon,v}\,\}\,.$$

For each $v \in \mathbb{R}^{+\infty}$, *define the* length-$v$ gS-distance *function* $d^v_{\mathrm{gS}} \colon (Z\times Z) \to \mathbb{R}^{+\infty}$ *by:*

$$d^v_{\mathrm{gS}}(\eta,\eta') \;:=\; \begin{cases} \infty & \text{if } (\forall \varepsilon \in \mathbb{R}^{>0})\;\; (\eta,\eta') \notin U_{\varepsilon,v} \\ \inf\{\,\varepsilon \in \mathbb{R}^{>0} \mid (\eta,\eta') \in U_{\varepsilon,v}\,\} & \text{otherwise}\,. \end{cases} \tag{4}$$

Define the weak gS-distance $d_{\mathrm{wgS}} : Z \times Z \to [0,1]$, *for all* $\eta,\eta' \in Z$, *by:*

$$d_{\mathrm{wgS}}(\eta,\eta') := \sum_{n=1}^{\infty} 2^{-n} \min\{\,1,\, d^n_{\mathrm{gS}}(\eta,\eta')\,\}\,, \tag{5}$$

and the gS-distance $d_{\mathrm{gS}} : Z \times Z \to [0,1]$, *for all* $\eta,\eta' \in Z$, *by:*

$$d_{\mathrm{gS}}(\eta,\eta') := \tfrac{1}{2}\Big(\,\min\{\,1,\, d^\infty_{\mathrm{gS}}(\eta,\eta')\,\} \;+\; d_{\mathrm{wgS}}(\eta,\eta')\,\Big) \tag{6}$$

In defining the length-v tolerance relation $U_{\varepsilon,v}$ and, from that, the length-v gS-distance d^v_{gS} in equation (4), *either* the paths η and η' are both of length less than $v+\varepsilon$, and they are ε-close in the fgS metric, *or else* there is a pair of time points $(t,t') \in \mathrm{dom}(\eta)\times\mathrm{dom}(\eta')$ with both of *at least* length v and the compact prefixes $\eta|_t$ and $\eta'|_{t'}$ are ε-close in the fgS metric; the latter entails that $\|\,t-t'\,\| < \varepsilon$ from the witnessing retiming, without requiring the overly-strong condition that $t' = t$.

Proposition 4. [gS-uniform topologies and pseudo-metrics on arbitrary-length paths] *Let* $Z \subseteq \mathsf{CP}^\star(T,X)$ *be any set of continuous paths. Then for all* $\varepsilon \in \mathbb{R}^{>0}$ *and for all paths* $\eta,\eta' \in Z$, *and all* $v \in \mathbb{R}^{+\infty}$,

$$d^v_{\mathrm{gS}}(\eta,\eta') < \varepsilon \quad \textit{iff} \quad (\eta,\eta') \in U_{\varepsilon,v}\,;$$

and

$$U_{\varepsilon,\infty}(\eta) = V_\varepsilon(\eta) \;\; \textit{iff} \;\; \mathsf{len}(\eta) < \infty\,; \quad \textit{and} \quad U_{\varepsilon,v}(\eta) = V_\varepsilon(\eta) \;\; \textit{if} \;\; \mathsf{len}(\eta) < v\,.$$

Each of the length-v *distance functions* d^v_{gS} *are pseudo-metrics on* Z, *as are both the gS-distance* d_{gS} *and the weak gS-distance* d_{wgS}, *and both families:*

$$\mathcal{U}_{\mathrm{gS}} \;:=\; \{\, U_{\varepsilon,v} \colon Z \rightsquigarrow Z \mid (\varepsilon,v) \in A_2^\infty\,\} \quad \textit{and} \quad \mathcal{U}_{\mathrm{wgS}} \;:=\; \{\, U_{\varepsilon,v} \colon Z \rightsquigarrow Z \mid (\varepsilon,v) \in A_2\,\}$$

constitute bases for uniformities on the path set Z. *The uniform topology* $\mathcal{T}_{\mathrm{wgS}}$ *on* Z *generated by* $\mathcal{U}_{\mathrm{wgS}}$ *has as its basic opens the* (ε,v)*-tubes* $U_{\varepsilon,v}(\eta)$ *for all finite pairs* $(\varepsilon,v) \in A_2$, *and is equivalently described by the family* $\{\, d^v_{\mathrm{gS}} \mid v \in \mathbb{R}^+\,\}$ *of pseudo-metrics. The uniform topology* $\mathcal{T}_{\mathrm{gS}}$ *on* Z *generated by* $\mathcal{U}_{\mathrm{gS}}$ *has as its basic opens the* (ε,v)*-tubes* $U_{\varepsilon,v}(\eta)$ *around* $\eta \in Z$, *for all* $(\varepsilon,v) \in A_2^\infty$; *it is equivalently described by the family* $\{\, d^v_{\mathrm{gS}} \mid v \in \mathbb{R}^{+\infty}\,\}$ *of pseudo-metrics; and it contains* $\mathcal{T}_{\mathrm{fgS}}$ *and* $\mathcal{T}_{\mathrm{wgS}}$ *as sub-topologies.*

We call $\mathcal{T}_{\mathrm{wgS}}$ the *topology of weak gS-uniform convergence*, and $\mathcal{T}_{\mathrm{gS}}$ the *topology of gS-uniform convergence*. For the Hausdorff property and metricizability, we can re-use the same notions developed for finite-length paths: the *discerning* and *highly discerning* properties. As for Proposition 2, by far the hardest part of Proposition 5 is that the discerning property implies the topology is Hausdorff. Verifying the equivalence of metric convergence and convergence in the uniform structures also takes some effort.

Proposition 5. [Properties of the 2-parameter gS-uniform topologies]
Let $Z \subseteq \mathsf{CP}^\star(T,X)$ be any set of continuous paths, of finite or infinite length.

1. *The uniform topologies $\mathcal{T}_{\mathrm{gS}}$ and $\mathcal{T}_{\mathrm{wgS}}$ on Z both have countable sub-bases.*
2. *The topologies $\mathcal{T}_{\mathrm{gS}}$ and $\mathcal{T}_{\mathrm{wgS}}$ on Z are both Hausdorff if Z is highly discerning.*
3. *The following five conditions are equivalent:*
   - *the path set Z is discerning;*
   - *the topology $\mathcal{T}_{\mathrm{gS}}$ on Z is Hausdorff;*
   - *the generalized Skorokhod pseudo-metric d_{gS} is a metric on Z;*
   - *the topology $\mathcal{T}_{\mathrm{wgS}}$ on Z is Hausdorff;*
   - *the weak generalized Skorokhod pseudo-metric d_{wgS} is a metric on Z.*
4. *If the path set Z is discerning, then for all sequences $\{\eta_k\}_{k\in\mathbb{N}}$ in Z and $\eta \in Z$:*
   (a) $\{\eta_k\}_{k\in\mathbb{N}}$ converges gS-uniformly to η iff $\lim_{k\to\infty} d_{\mathrm{gS}}(\eta,\eta_k) = \lim_{k\to\infty} d^{\infty}_{\mathrm{gS}}(\eta,\eta_k) = 0$;
   (b) $\{\eta_k\}_{k\in\mathbb{N}}$ converges wgS-uniformly to η iff $\lim_{k\to\infty} d_{\mathrm{wgS}}(\eta,\eta_k) = 0$;
   (c) if all but finitely-many of the paths η_k, for $k \in \mathbb{N}$, as well as the path η, have finite length, then the following conditions on $\{\eta_k\}_{k\in\mathbb{N}}$ are equivalent:
      - *it converges to η in the finite-length paths topology $\mathcal{T}_{\mathrm{fgS}}$;*
      - *it converges gS-uniformly to η ; and*
      - *it converges wgS-uniformly to η .*

   Hence when the path set Z is discerning, the topology $\mathcal{T}_{\mathrm{gS}}$ is metricized by d_{gS}, and the topology $\mathcal{T}_{\mathrm{wgS}}$ is metricized by d_{wgS} .

For $T = \mathbb{R}^+$ and $T = \mathbb{H}$, it is easy to find examples of sequences of paths $\{\eta_k\}_{k\in\mathbb{N}}$ that converge wgS-uniformly to an infinite-length path η, but do not converge in the stronger metrics d_{gS} and d^{∞}_{gS}. So the metrics and topologies are quite distinct, with $\mathcal{T}_{\mathrm{wgS}} \subsetneq \mathcal{T}_{\mathrm{gS}}$.

Example 1 revisited. Taking $\varepsilon_1 \leq 0.65$, we can compute rough numerical bounds of $d_{\mathrm{gS}}(\gamma,\gamma') < 0.84$ and $d_{\mathrm{wgS}}(\gamma,\gamma') < 0.61$ for gS-distances between the compact path γ and the spatially-unbounded path γ' with finite-escape time. Compare these with bounds of $d_{\mathrm{wgS}}(\eta,\eta'') \leq d_{\mathrm{gS}}(\gamma,\gamma'') \leq d_{\mathrm{fgS}}(\gamma,\gamma'') \leq 0.95$ for the sampling+quantization, with all three distances about the same. As depicted in Fig. 1, this makes sense: the path γ' is *closer* to γ than the coarsely sampled+quantized path γ''.

6 Relationship with Graphical Set-Convergence of Paths

Goebel and Teel in [3] develop a notion of convergence for sequences of hybrid paths (compact or limit) for the case of Euclidean space $X \subseteq \mathbb{R}^n$ and $T = \mathbb{H} \subset \mathbb{R}^2$ by employing the machinery of *set-convergence* for sequences of subsets Euclidean space, applied to paths $\eta \in \mathsf{CP}^\star(T,X)$ considered via their graphs as subsets of $T \times X \subset \mathbb{R}^{n+2}$; the text [16] is a standard reference on set-convergence. For any sequence $\{A_k\}_{k\in\mathbb{N}}$ of non-empty subsets of a metric space, in general, $\liminf_{k\to\infty} A_k \subseteq \limsup_{k\to\infty} A_k$, and the sequence $\{A_k\}_{k\in\mathbb{N}}$ *set-converges* to a set A if $\limsup_{k\to\infty} A_k = A = \liminf_{k\to\infty} A_k$, in which case A must be closed in the metric, and we write $A = \mathrm{setlim}_{k\to\infty} A_k$.

Proposition 6. [Equivalence of concepts of convergence]
Let S be a finite-dimensional time structure with future T, let (X,d_x) be a metric space, and let $Z \subseteq \mathsf{CP}^\star_{\mathrm{cl}}(T,X)$ be any set of paths with norm-closed time domains. Then for all paths $\eta \in Z$ and for all sequences of paths $\{\eta_k\}_{k\in\mathbb{N}}$ within Z, the following are equivalent:

(1) *the sequence* $\{\eta_k\}_{k\in\mathbb{N}}$ *converges wgS-uniformly to* η;
(2) $\lim_{k\to\infty} d_{\mathrm{wgS}}(\eta,\eta_k) = 0$;
(3) $\eta = \operatorname{setlim}_{k\to\infty} \eta_k$ *as graphs in the product topology on* $T\times X$; *and*
(4) $\forall$ *open sets* O *in* $T\times X$, *if* $\eta\cap O\neq\emptyset$ *then* $(\exists\, m_1\in\mathbb{N})(\forall k\geq m_1)\quad \eta_k\cap O\neq\emptyset$, *and*
$\forall$ *compact sets* K *in* $T\times X$, *if* $\eta\cap K=\emptyset$ *then* $(\exists\, m_2\in\mathbb{N})(\forall k\geq m_2)\quad \eta_k\cap K=\emptyset$.

7 Application: Completeness and Semi-continuity of Hybrid Flows

A key result of [3] (subsequently used in [4,6] and elsewhere) is their Theorem 4.4 on a type of sequential compactness; it identifies conditions on the components of a hybrid system $H = (X, F, G, C, D)$ such that for $P := \mathrm{ran}(\Phi_{\mathrm{H}})$, the path set $Z := P \cup \mathsf{M}(P)$ is such that for every locally eventually bounded sequence $\{\eta_k\}_{k\in\mathbb{N}}$ in Z, there exists a path $\eta\in Z$ and a sub-sequence $\{\eta_{k_m}\}_{m\in\mathbb{N}}$ with $\eta = \operatorname{setlim}_{m\to\infty}\eta_{k_m}$. A sequence $\{\eta_k\}_{k\in\mathbb{N}}$ is *locally eventually bounded* iff for all length-bounds $b\in\mathbb{R}^{>0}$, there exists $m_b\in\mathbb{N}$ and a compact set $K_b\subseteq X$ such that for all $k\geq m_b$ and all $(i,t)\in\mathrm{dom}(\eta_k)$, if $\|(i,t)\|_{\mathbb{H}}\leq b$ then $\eta_k(i,t)\in K_b$. In the result below, we take the same conditions as identified in [3,4,6], and derive stronger conclusions cast in terms of the metrics d_{fgS}, d_{gS} and d_{wgS} on the spaces spaces Z_{fin} and Z.

For metric spaces X and Y, a set-valued map $R\colon X\rightsquigarrow Y$ is *locally-bounded* iff for every compact set $K\subseteq X$, the set-image $R(K)$ is bounded in Y. If $Y\subseteq\mathbb{R}^n$, then $R\colon X\rightsquigarrow Y$ is locally-bounded and outer semi-continuous iff R is upper semi-continuous and has compact values $R(x)\subseteq Y$. For $x\in\mathbb{R}^n$ and a set $C\subset\mathbb{R}^n$, the *tangent cone to* C *at* x is the set $\mathrm{TC}_C(x)$ of all vectors $v\in\mathbb{R}^n$ for which there exists a sequence $\{\alpha_k\}_{k\in\mathbb{N}}$ of positive reals converging monotonically to 0, together with a sequence $\{v_k\}_{k\in\mathbb{N}}$ in $\mathbb{R}^n$ converging to v, such that $v+\alpha_k v_k\in C$ for all $k\in\mathbb{N}$; see [7,16].

Proposition 7. [Cauchy-completeness, and semi-continuity of hybrid trajectories]
Let $H = (X, F, G, C, D)$ *be a hybrid system as described in Section 4, with general flow map* $\Phi_{H}\colon X\rightsquigarrow P_{\mathrm{hyb}}(X)$ *giving the compact-domain trajectories of* H *from any initial state* $x\in\mathrm{dom}(\Phi_{H}) = (C\cup D)\subseteq X$. *From* [3,4,6], *assume:*
(A0) $X\subseteq\mathbb{R}^n$ *is an open set;*
(A1) C *and* D *are relatively closed sets in* X;
(A2) $F:\mathbb{R}^n\rightsquigarrow\mathbb{R}^n$ *is outer semi-continuous and locally-bounded, and* $F(x)$ *is convex and compact in* $\mathbb{R}^n$ *for each* $x\in C$;
(A3) $G:\mathbb{R}^n\rightsquigarrow\mathbb{R}^n$ *is outer semi-continuous;*
(VC) *for all* $x\in C\setminus D$, *there exists an* $\varepsilon>0$ *such that* $\mathrm{TC}_C(x')\cap F(x')\neq\emptyset$ *for every* ε*-close state* $x'\in B_\varepsilon(x)\cap C$; *and*
(VD) $G(x)\subseteq(C\cup D)$ *for all* $x\in D$.
Then let $P := \mathrm{ran}(\Phi_{H}) = \{\gamma\in\Phi_{H}(x)\mid x\in C\cup D\}$, *let* $Z := P\cup\mathsf{M}(P)$, *let* $Z_{\mathrm{inf}} := \mathsf{M}^{\infty}(P)$, *let* $Z_{\mathrm{fe}} := \mathsf{M}(P)\setminus\mathsf{M}^{\infty}(P)$, *and let* $Z_{\mathrm{fin}} := P\cup Z_{\mathrm{fe}}$, *so* $Z = P\cup Z_{\mathrm{fe}}\cup Z_{\mathrm{inf}} = Z_{\mathrm{fin}}\cup Z_{\mathrm{inf}}$, *with the unions disjoint. Further partition* Z_{inf} *as* $Z_{\mathrm{inf}} = Z_0\cup Z_1\cup Z_\infty$, *where* $\eta\in Z_0$ *iff* η *is Zeno;* $\eta\in Z_1$ *iff* η *has finitely-many discrete transitions and* $\mathsf{len}(\eta)=\infty$; *and* $\eta\in Z_\infty$ *iff* η *has infinitely-many discrete transitions and* $\mathsf{len}(\eta)=\infty$. *Then:*

1. P *is prefix-closed and maximally-extendible, and for all* $\eta\in\mathsf{M}(P)$, *either* η *has infinite length or* η *is spatially-unbounded in* X.

2. *Both the sets P and Z_{fe}, as well as the space Z_{fin}, are both open and closed, and Cauchy-complete, in the metric d_{fgS} on Z_{fin}.*
3. *Each of the five path sets P, Z_{fe}, Z_0, Z_1 and Z_∞, as well as the whole space Z, are both open and closed, and Cauchy-complete, in the metrics d_{gS} and d_{gS}^∞ on Z.*
4. *Each of the sets Z_{inf}, $P \cup Z_{\text{inf}}$, and $\mathsf{M}(P)$, as well as the whole space Z, are closed and Cauchy-complete in the metric d_{wgS} on Z, while P and Z_{fe} are both open.*
5. *Additionally assume* (A4) : *the map $G : X \rightsquigarrow X$ is locally-bounded. Then:*
   (a) *The flow map $\Phi_H : X \rightsquigarrow P_{\text{hyb}}(X)$ is (globally) outer semi-continuous w.r.t. the metric d_X on X and the metric d_{fgS} on $P_{\text{hyb}}(X)$, and the map $\mathsf{M}\Phi_H : X \rightsquigarrow Z$ is (globally) outer semi-continuous w.r.t. each of d_{wgS}, d_{gS} and d_{gS}^∞ on Z.*
   (b) *For each $x \in \text{dom}(\Phi_H) = C \cup D$, the set $\Phi_H(x)$ of compact paths is closed and Cauchy-complete in the metric d_{fgS} on P.*
   (c) *For each $x \in C \cup D$, the set $\mathsf{M}\Phi_H(x)$ of maximal paths is closed and Cauchy-complete w.r.t. each of the metrics d_{gS}, d_{gS}^∞ and d_{wgS} on Z.*
   (d) *For each $x \in C \cup D$, if $\mathsf{M}\Phi_H(x) \subset Z_{\text{inf}}$, then for every $(\varepsilon, \nu) \in A_2$, there exists a real $\delta \in (0, \varepsilon]$ such that $\mathsf{M}\Phi_H(B_\delta(x)) \subseteq U_{\varepsilon,\nu}(\mathsf{M}\Phi_H(x))$, hence $\mathsf{M}\Phi_H : X \rightsquigarrow Z$ is locally upper semi-continuous at x w.r.t. the metric d_{wgS} on Z.*
   (e) *If $K \subseteq (C \cup D)$ is compact and $\mathsf{M}\Phi_H(K) \subseteq Z_{\text{inf}}$, then for every $(\varepsilon, \nu) \in A_2$, there exists a real $\delta \in (0, \varepsilon]$ such that $\mathsf{M}\Phi_H(B_\delta(K)) \subseteq U_{\varepsilon,\nu}(\mathsf{M}\Phi_H(K))$.*

Theorem 4.4 of [3] can be used in proving part of *Part 4*, while *Part 5(e)* is an equivalent reformulation of Corollary 4.8 from that paper. From the viewpoint of stability, the upper semi-continuity of the map $\mathsf{M}\Phi_H : X \rightsquigarrow Z$ is highly desirable. The slightly stronger assumptions on the components of H used in [7] are sufficient to ensure that $\mathsf{M}\Phi_H$ is globally upper semi-continuous w.r.t. each of the metrics d_{wgS}, d_{gS} and d_{gS}^∞.

8 Conclusion

This paper develops several generalized Skorokhod pseudo-metrics for hybrid path spaces, cast in a quite general setting, where paths are continuous functions from a normed and partially-ordered time structure into a metric space, with the domains of paths linearly-ordered. The topologies generalize the original Skorokhod metric by allowing set-valued order-preserving retiming maps that are readily invertible and composable, are in practice quite easy to work with, and they include single-valued order-preserving maps as special cases. We determine necessary and sufficient conditions under which these topologies are Hausdorff and the distances are metrics. One of these metrics on arbitrary-length paths, that of weak gS-uniform convergence, is shown to be equivalent to the implicit topology of graphical convergence of hybrid paths, currently used extensively by Teel and co-workers. We apply the framework to investigate topological properties of hybrid general flows in the metrics d_{fgS}, d_{wgS} and d_{gS}.

The original motivation for this work was to develop topological and metric foundations as a prequel to giving a *robust semantics* for the temporal logic **GFL*** [12,13], which generalizes computational tree logic **CTL*** to semantics over general flow systems, uniformly for arbitrary time structures – discrete, continuous or hybrid. The key idea is that if a system satisfies a performance specification robustly, with the specification given by a logic formula, then a path η satisfies the specification only when all the

paths in some ε-tube around η also satisfy the specification. With those foundations in place, that research project is under way.

References

1. Goebel, R., Hespanha, J., Teel, A.R., Cai, C., Sanfelice, R.: Hybrid systems: Generalized solutions and robust stability. In: IFAC Symp. Nonlinear Control Systems, pp. 1–12 (2004)
2. Collins, P.J.: A trajectory-space approach to hybrid systems. In: Proc. of 16th Int. Symp. on Math. Theory of Networks and Systems, MTNS 2004 (2004)
3. Goebel, R., Teel, A.R.: Solutions to hybrid inclusions via set and graphical convergence with stability theory applications. Automatica 42, 596–613 (2006)
4. Cai, C., Teel, A., Goebel, R.: Smooth Lyapunov functions for hybrid systems. Part I: existence is equivalent to robustness. IEEE Trans. Aut. Control 52, 1264–1277 (2007)
5. Collins, P.J.: Generalized hybrid trajectory spaces. In: Proc. of 17th Int. Symp. on Math. Theory of Networks and Systems (MTNS 2006), pp. 2101–2109 (2006)
6. Sanfelice, R., Goebel, R., Teel, A.R.: Invariance principles for hybrid systems with connections to detectability and asymptotic stability. IEEE Trans. Aut. Control 52, 2282–2297 (2007)
7. Aubin, J.P., Lygeros, J., Quincampoix, M., Sastry, S., Seube, N.: Impulse differential inclusions: A viability approach to hybrid systems. IEEE Trans. Aut. Control 47, 2–20 (2002)
8. Lygeros, J., Johansson, K.H., Simic, S.N., Zhang, J., Sastry, S.S.: Dynamical properties of hybrid automata. IEEE Trans. Automatic Control 48, 2–16 (2003)
9. Alur, R., Courcoubetis, C., Halbwachs, N., Henzinger, T.A., Ho, P.-H., Nicollin, X., Olivero, A., Sifakis, J., Yovine, S.: The algorithmic analysis of hybrid systems. Theoretical Computer Science 138, 3–34 (1995)
10. Alur, R., Henzinger, T.A., Ho, P.-H.: Automatic symbolic verification of embedded systems. IEEE Trans. on Software Engineering 22, 181–201 (1996)
11. Davoren, J.M., Nerode, A.: Logics for hybrid systems. Proc. of the IEEE 88, 985–1010 (2000)
12. Davoren, J.M., Coulthard, V., Markey, N., Moor, T.: Non-deterministic temporal logics for general flow systems. In: Alur, R., Pappas, G.J. (eds.) HSCC 2004. LNCS, vol. 2993, pp. 280–295. Springer, Heidelberg (2004)
13. Davoren, J.M., Tabuada, P.: On simulations and bisimulations of general flow systems. In: Bemporad, A., Bicchi, A., Buttazzo, G. (eds.) HSCC 2007. LNCS, vol. 4416, pp. 145–158. Springer, Heidelberg (2007)
14. der Schaft, A.V., Schumacher, J.: An Introduction to Hybrid Dynamical Systems (2000)
15. Julius, A.: On Interconnection and Equivalences of Continuous and Discrete Systems: A Behavioural Perspective. The University of Twente, PhD thesis (2005)
16. Rockafellar, R., Wets, R.J.: Variational Analysis. Springer, Berlin (1998)
17. Broucke, M.: Regularity of solutions and homotopic equivalence for hybrid systems. In: 37th IEEE Conference on Decision and Control (CDC 1998), pp. 4283–4288 (1998)
18. Broucke, M.E., Arapostathis, A.: Continuous selections of trajectories of hybrid systems. Systems and Control Letters 47, 149–157 (2002)
19. Kossentini, C., Caspi, P.: Mixed delay and threshold voters in critical real-time systems. In: Lakhnech, Y., Yovine, S. (eds.) FORMATS 2004 and FTRTFT 2004. LNCS, vol. 3253, pp. 21–35. Springer, Heidelberg (2004)
20. Pollard, D.: Convergence of Stochastic Processes. Springer, New York (1984)
21. Davoren, J., Epstein, I.: Topologies and convergence in general hybrid path spaces. In: Proc. of 18th Int. Symp. on Math. Theory of Networks and Systems (MTNS 2008) (2008)
22. Goodearl, K.: Partially Ordered Abelian Groups With Interpolation. Mathematical Surveys and Monographs. American Mathematical Society, Providence (1986)
23. Stone, M.H.: Pseudo-norms and partial orderings in abelian groups. Annals of Mathematics (series 2) 48, 851–856 (1947)

Stability Analysis of Networked Control Systems Using a Switched Linear Systems Approach

M.C.F. Donkers[1], L. Hetel[2], W.P.M.H. Heemels[1],
N. van de Wouw[1], and M. Steinbuch[1]

[1]Eindhoven University of Technology, The Netherlands
[2]Ecole Centrale de Lille, France

Abstract. In this paper, we study the stability of Networked Control Systems (NCSs) that are subject to time-varying transmission intervals and communication constraints in the sense that, per transmission, only one node can access the network and send its information. The order in which nodes send their information is dictated by a network protocol, such as the well-known Round Robin (RR) or Try-Once-Discard (TOD) protocol. Focussing on linear plants and linear continuous-time or discrete-time controllers, we model the NCS with time-varying transmission intervals as a discrete-time switched linear uncertain system. We obtain bounds for the allowable range of transmission intervals in terms of both minimal and maximal allowable transmission intervals. Hereto, a new convex overapproximation of the uncertain switched system is proposed, using a polytopic system with norm-bounded uncertainty, and new stability results for this class of hybrid systems are developed. On the benchmark example of a batch reactor, we explicitly exploit the linearity of the system, leading to a significant reduction in conservatism with respect to the existing approaches.

1 Introduction

In many control applications nowadays, controllers are implemented on a system having spatially distributed sensors and actuators that are closed over a shared real-time network. These Networked Control Systems (NCSs) offer several advantages such as less wiring and cost, increased system's flexibility and ease of installation and maintenance. To harvest the advantages that NCSs can offer, control algorithms are needed that can deal with communication imperfections and constraints. This latter aspect is recognised widely as is evidenced by the broad attention received by NCSs recently, see, e.g., the overview papers [1,2,3,4].

One source of communication imperfections is the fact that sensors/controllers/actuators do not operate synchronously anymore causing variations in sampling/transmission intervals. Also the presence of the network results in delays between the transmittal and the arrival of the data packets. The finite word length of the packets causes quantisation errors in the transmitted interval. Moreover, communication constraints are induced by restrictions of the network in

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 150–164, 2009.

the sense that not all sensor and control values can be transmitted at the same time. Typically, at each transmission time only a selected set of sensors and actuators (called a node) has access to the shared network to communicate its data. The effects of quantisation and communication delays in NCSs are studied in, e.g., [5,6] and [7,8,9], respectively. In this paper, we will focus on the stability of NCSs with time-varying transmission intervals and the presence of communication constraints in the sense that, per transmission, only one node can access the network.

The communication constraints in NCSs give rise to the problem of how to schedule which nodes are given access to the network and when. The algorithms that dictate the scheduling of tasks are often referred to as protocols. Some well-known and often used protocols are, the Round Robin (RR) protocol and the Try-Once-Discard (TOD) protocol [10,11,12,13,14]. The stability assessment of NCSs with communication constraints and time-varying transmission intervals has already been considered in [10,14,15,16,17]. These papers provide criteria for computing the so-called Maximal Allowable Transmission Interval (MATI). Stability is guaranteed as long as the transmission interval is smaller than the MATI. These results apply for general nonlinear plants and controllers and a wide class of protocols (including the RR and TOD protocols) and are based on a continuous-time modelling paradigm related to hybrid inclusions [18]. However, these results do not include the possibility that the controller is formulated in a discrete-time form, which is of interest in many practical situations due to digital implementations. Only recently, the case of discrete-time controllers has been considered in [19], however, assuming a fixed transmission interval. Another difference is that in [10,14,15,16,17] always a zero lower-bound on the transmission intervals (i.e., $h_k \in (0, \text{MATI}]$) is considered, while we also allow for non-zero lower bounds, which is often more realistic in many situations. Although the work in [10,14,15,16,17] presents a research line that is very general and can accommodate for many nonlinear NCSs, their results might become conservative when more structure is present in the NCS such as, e.g., linearity of the controller and plant.

In this paper, we will focus on linear plants and linear controllers and study the stability of the corresponding NCS in the presence of communication constraints and time-varying transmission intervals, possibly having a *non-zero* lower bound. Moreover, we allow that the controller can be either continuous-time or *discrete-time*, which requires a different approach than in [10,14,15,16,17]. To be more precise, for the RR protocol, the TOD protocol and the newly introduced class of quadratic protocols we will provide techniques for assessing stability of the NCS with time-varying transmission intervals $h_k \in [\underline{h}, \bar{h}]$ using Linear Matrix Inequalities (LMIs). In contrast with [10,14,15,16,17], we will apply a *discrete-time* modelling framework that leads to a switched linear uncertain system. Hybrid stability methods will be used to determine the stability of this NCS model based on a polytopic overapproximation. To obtain this overapproximation, we will present a novel technique that combines ideas from gridding as in [20] and norm-bounding as in [21]. We will show the effectiveness of the

presented approach on the benchmark example of the batch reactor as used also in [10,14,15,16,17]. Moreover, we will show that the linearity of plant and controller can indeed be exploited and leads to a significant reduction of conservatism with respect to the existing approaches.

The following notational conventions will be used: $\mathrm{diag}(A_1, \ldots, A_n)$ denotes a block-diagonal matrix with the entries $A_1, \ldots, A_n$ on the diagonal, $\|x\| := \sqrt{x^\top x}$ the Euclidean norm of a vector $x \in \mathbb{R}^n$, and $\|A\| := \sqrt{\lambda_{\max}(A^\top A)}$ the spectral norm, which is the square-root of the maximum eigenvalue of the matrix $A^\top A$.

2 The Networked Control System and Problem Formulation

In this section, we introduce the Networked Control System (NCS) under study, a discrete-time model describing it and give the problem formulation.

2.1 Description of the NCS

Both the plant and the controller are linear time-invariant systems, where the plant is given in continuous-time by

$$\begin{cases} \dot{x}(t) &= Ax(t) + B\hat{u}(t), \qquad \hat{u}(t) = \hat{u}(t_k)\ \forall t \in [t_k, t_{k+1}) \\ y(t) &= Cx(t) \end{cases} \tag{1}$$

and the controller is given in discrete-time, i.e.,

$$\begin{cases} \xi_{k+1} &= A_c\xi_k + B_c\hat{y}_k \\ u_k &= C_c\xi_k + D_c\hat{y}_{k-1}. \end{cases} \tag{2}$$

In these descriptions, $x \in \mathbb{R}^{n_x}$ and $\xi \in \mathbb{R}^{n_\xi}$ denote the states of the plant and controller, respectively, $y \in \mathbb{R}^{n_y}$ denotes the measured plant output, $u \in \mathbb{R}^{n_u}$ the controller output. The description given by (1) and (2) can cover the situation of a single plant having multiple inputs and outputs, as well as separate plants with separate controllers that share a common network. In the latter case, both (1) and (2) typically have a diagonal structure. Furthermore, t_k, $k \in \mathbb{N}$, denote the transmission times at which the controller is updated. Since the plant and controller are communicating through a network, the actual input of the plant $\hat{u} \in \mathbb{R}^{n_u}$ is not equal to u and the actual input of the controller $\hat{y} \in \mathbb{R}^{n_y}$ is not equal to y. Instead, $\hat{u}$ and $\hat{y}$ are 'networked versions' of u and y, respectively.

To introduce these networked versions $\hat{u}$ and $\hat{y}$ properly, we have to explain the functioning of the network. The plant is equipped with n_y sensors and with n_u actuators. These sensors and actuators are grouped into $N \leq n_y + n_u$ nodes, where we assume that actuators and sensors are not in the same nodes. At each transmission time t_k, $k \in \mathbb{N}$, one node obtains access to the network and its corresponding values in u or y are transmitted. In this work, as in [10,14,16,19], we assume that the data is not delayed and packet loss does not occur. Only the

transmitted values will be updated in $\hat{u}$ and $\hat{y}$, while the other values in $\hat{u}$ and $\hat{y}$ remain the same. Such constrained data exchange can be expressed as

$$\begin{cases} \hat{y}_k &= \Gamma^y_{\sigma_k} y_k + (I - \Gamma^y_{\sigma_k})\hat{y}_{k-1} \\ \hat{u}_k &= \Gamma^u_{\sigma_k} u_k + (I - \Gamma^u_{\sigma_k})\hat{u}_{k-1}, \end{cases} \tag{3}$$

where $\Gamma_{\sigma_k} = \operatorname{diag}\left(\Gamma^y_{\sigma_k}, \Gamma^u_{\sigma_k}\right)$ is a diagonal matrix taken from the set $\mathcal{G} = \{\Gamma_1, \ldots, \Gamma_N\}$, with

$$\Gamma_i = \operatorname{diag}\left(\gamma_{i,1} I_1, \ldots, \gamma_{i,N} I_N\right). \tag{4}$$

In (4), I_j denotes the identity matrix with dimensions corresponding to the number of sensors or actuators in node j. The elements $\gamma_{i,j}$, with $j \in \{1, \ldots, N\}$, of the each matrix Γ_i is given by $\gamma_{i,j} = 1$, when $j = i$, and $\gamma_{i,j} = 0$, when $j \neq i$. Note that $\Gamma_{\sigma_k} \in \mathcal{G}$ also formalises the assumption that actuators and sensors cannot be in the same node, since for each i only one $\gamma_{i,j}$ can be equal to one.

The value of σ_k lies in $\{1, \ldots, N\}$ and its value indicates which node is given access to the network at transmission time t_k. Indeed, (3) reflects that the values in $\hat{u}$ and $\hat{y}$ corresponding to node σ_k are updated with the corresponding transmitted values, while the others stay the same. A protocol determines the values of $(\sigma_0, \sigma_1, \ldots)$, which are made explicit later. Note that because of the functioning of the network, the direct feed-through of the controller is based on y_{k-1}, instead of y_k, as in [19].

The transmission times t_k, $k \in \mathbb{N}$, are not necessarily distributed equidistantly in time. Hence, the transmission intervals $h_k = t_{k+1} - t_k$ are time-varying. We assume that these variations are bounded and lie in the set $[\underline{h}, \overline{h}]$. Hence, $h_k \in [\underline{h}, \overline{h}]$ for all $k \in \mathbb{N}$. Note that in [10,14,16], only $\underline{h} = 0$ was allowed, while here $\underline{h} > 0$ is considered. This latter situation is more natural when using a discrete-time controller, since such a controller is implicitly designed for some nominal transmission interval larger than zero.

2.2 Discrete-Time NCS and Problem Formulation

To arrive at a discrete-time model for the NCS, we have to obtain a discrete-time equivalent of (1). Since the inputs of the controller are constant between subsequent transmissions due to the zero-order hold, we can exactly discretise the plant (1) at the transmission times t_k resulting in

$$\begin{cases} x_{k+1} &= e^{Ah_k} x_k + \int_0^{h_k} e^{As} ds B \hat{u}_k \\ y_k &= C x_k, \end{cases} \tag{5}$$

where $x_k := x(t_k)$ and $u_k := u(t_k)$, $k \in \mathbb{N}$. If we define the network-induced error $e_k = [(e^y_k)^\top (e^u_k)^\top]^\top$, by

$$\begin{cases} e^y_k &:= \hat{y}_{k-1} - y_k \\ e^u_k &:= \hat{u}_{k-1} - u_k, \end{cases} \tag{6}$$

we can obtain the complete NCS model by combining (2), (3), (5), and (6). This results in

$$\bar{x}_{k+1} := \begin{bmatrix} x_{k+1} \\ \xi_{k+1} \\ e^y_{k+1} \\ e^u_{k+1} \end{bmatrix} = \tilde{A}_{\sigma_k,h_k} \begin{bmatrix} x_k \\ \xi_k \\ e^y_k \\ e^u_k \end{bmatrix}, \tag{7}$$

where $\tilde{A}_{\sigma_k,h_k} \in \mathbb{R}^{n\times n}$, with $n = n_x + n_\xi + n_y + n_u$, is given by

$$\tilde{A}_{\sigma_k,h_k} = \begin{bmatrix} e^{Ah_k} + E_{h_k}BD_cC & E_{h_k}BC_c & E_{h_k}BD_c & E_{h_k}B(I-\Gamma^u_{\sigma_k}) \\ B_cC & A_c & B_c(I-\Gamma^y_{\sigma_k}) & 0 \\ C(I-e^{Ah_k}-E_{h_k}BD_cC) & -CE_{h_k}BC_c & I-\Gamma^y_{\sigma_k}-CE_{h_k}BD_c & -CE_{h_k}B(I-\Gamma^u_{\sigma_k}) \\ -C_cB_cC & C_c(I-A_c) & D_c\Gamma^y_{\sigma_k}-C_cB_c(I-\Gamma^y_{\sigma_k}) & I-\Gamma^u_{\sigma_k} \end{bmatrix} \tag{8}$$

and $E_{h_k} = \int_0^{h_k} e^{As}ds$.

In this paper, we focus on two commonly used protocols, see [10,14,15,16,17], namely the Try-Once-Discard (TOD) and the Round-Robin (RR) protocol. In the TOD protocol, the node that has the largest network-induced error, i.e., the difference between the most recently received value and the current value of the node, is granted access to the network. To make this more precise, assume that e_k is partitioned as $e_k = [(e^1_k)^\top, \ldots, (e^N_k)^\top]^\top$, according to the nodes. Hence, e^i_k is the networked induced error for the signals corresponding to node i. For the TOD protocol, the switching function is now given by

$$\sigma_k = \arg\max \left\{ \|e^1_k\|, \ldots, \|e^N_k\| \right\}. \tag{9}$$

In the case that two nodes have the same values, one of them is chosen arbitrarily. For the RR protocol, each node is granted access periodically and the switching function is given by

$$\sigma_k = \begin{cases} 1, & \text{if } \ k = 1 + jN, \quad \text{for some } j \in \mathbb{N} \\ 2, & \text{if } \ k = 2 + jN, \quad \text{for some } j \in \mathbb{N} \\ \vdots & \\ N, & \text{if } \ k = N, \quad \text{for some } j \in \mathbb{N}. \end{cases} \tag{10}$$

The above modelling approach now provides a description of the NCS system in the form of an *uncertain switched linear system* given by (7) and one of the protocols (9) and (10). The system switches between N linear uncertain systems and the switching is due to the fact that only one node accesses the network at each transmission time. The uncertainty is caused by the fact that the transmission interval $h_k \in [\underline{h}, \overline{h}]$ is time-varying. Let us now formally define stability for the NCS.

Definition 1 (Uniform Global Exponential Stability). *System (7) with (9) or (10), is said to be* uniformly globally exponentially stable *(UGES) if there exist $c > 0$ and $0 \leq \lambda < 1$, such that for any initial condition $\bar{x}_0 \in \mathbb{R}^n$, and any sequence of transmission intervals $(h_0, h_1, \ldots)$, with $h_k \in [\underline{h}, \overline{h}]$, for all $k \in \mathbb{N}$, it holds that*

$$\|\bar{x}_k\| \leq c\|\bar{x}_0\|\lambda^k. \tag{11}$$

The problem studied in this paper is to determine the UGES of the NCS model (7) with (9) or (10) given the bounds $h_k \in [\underline{h}, \overline{h}]$, or to find these bounds.

Remark 1. In Definition 1, we defined UGES of the uncertain discrete-time NCS model (7), whereas the states of the plant (1) actually evolve in continuous-time. In [22], it is shown that the intersample behaviour is bounded as a function of the states on the transmission times, and consequently, stability of the discrete-time NCS model also implies stability of the continuous-time NCS. ■

Remark 2. Although, we mainly focus on the case of a discrete-time controller (2), we can also incorporate continuous-time controllers in our framework. Indeed, in case of the continuous-time controller

$$\begin{cases} \dot{\xi} &= \tilde{A}_c \xi + \tilde{B}_c \hat{y} \\ u &= C_c \xi + D_c \hat{y} \end{cases} \tag{12}$$

the A_c and B_c-matrices in (8) for the NCS model (7) have to be modified to

$$A_c = e^{\tilde{A}_c h_k} \quad \text{and} \quad B_c = \int_0^{h_k} e^{\tilde{A}_c s} ds \tilde{B}_c, \tag{13}$$

which then also become uncertain and time-varying. ■

2.3 Overapproximation of the NCS Model by a Polytopic System

The form (7) is not really convenient to obtain efficient techniques for stability analysis due to the nonlinear appearance of the uncertain parameter h_k in (8). Therefore, we will provide a procedure that overapproximates system (7) with a polytopic system with a norm-bounded additive uncertainty of the form

$$\bar{x}_{k+1} = \sum_{l=1}^{M} \left(\alpha_{k,l} \bar{A}_{\sigma_k,l} + \alpha_{k,l} \bar{B}_l \Delta_k \bar{C}_{\sigma_k} \right) \bar{x}_k, \tag{14}$$

where $\bar{B}_l \in \mathbb{R}^{n \times m}$, $\bar{C}_{\sigma_k} \in \mathbb{R}^{m \times n}$, and $\alpha_k = [\alpha_{k,1} \ \dots \ \alpha_{k,M}]^\top \in \mathcal{A}$ denotes an unknown time-varying vector with

$$\mathcal{A} = \left\{ \alpha \in \mathbb{R}^M \mid \sum_{l=1}^{M} \alpha_l = 1, \alpha_l \geq 0 \right\}. \tag{15}$$

Moreover $\Delta_k \in \mathbf{\Delta}$, where $\mathbf{\Delta}$ is a set of matrices in $\mathbb{R}^{m \times m}$, describing the additive uncertainty, which possibly has some structure, as we will see below. Equation (14) should be an overapproximation of (7) in the sense that

$$\left\{ \tilde{A}_{\sigma_k,h_k} \mid h_k \in [\underline{h}, \overline{h}] \right\} \subseteq \left\{ \sum_{l=1}^{M} \alpha_{k,l} \left(\bar{A}_{\sigma_k,l} + \bar{B}_l \Delta_k \bar{C}_{\sigma_k} \right) \mid \alpha_k \in \mathcal{A}, \Delta_k \in \mathbf{\Delta} \right\}. \tag{16}$$

In this paper, we use the idea of [20] to obtain $\bar{A}_{\sigma_k,l}$ by gridding (8) at a collection of selected transmission intervals. However, we choose to allow for convex combinations of the vertices corresponding to the grid points, whereas in [20], the system switches between these vertices. For that reason, we can grid at *a priori* chosen points $\tilde{h}_1, \ldots, \tilde{h}_M \in [\underline{h}, \overline{h}]$, and construct a norm-bounded additive uncertainty $\Delta \in \mathbf{\Delta}$ to capture the remaining approximation error, as done in, e.g., [21]. Hence, $\bar{A}_{\sigma_k,l} := \tilde{A}_{\sigma_k,\tilde{h}_l}$ in (14), with $l \in \{1, \ldots, N\}$. In contrast with [20], this procedure prevents the problem of an iterative procedure in which the number of grid points can become large, resulting in intractability. Furthermore, we obtain smaller bounds on the additive uncertainty than in [21]. This explains that the newly proposed method performs better with respect to both complexity and approximation accuracy.

By specifying the grid points, and thereby determining $\bar{A}_{\sigma_k,l}$, it only remains to show how to specify $\bar{B}_l \Delta_k C_{\sigma_k}$ in (14) and $\mathbf{\Delta}$ as this should be used to satisfy (16). This additive uncertainty is used to capture the approximation error between the original system (7) and the polytopic system

$$\bar{x}_{k+1} = \sum_{l=1}^{M} \alpha_{k,l} \bar{A}_{\sigma_k,l} \bar{x}_k, \tag{17}$$

which consists of the convex combination of the gridded matrices. In order for (16) to hold, for each h and each σ, these should exist some $\alpha \in \mathcal{A}$ and $\Delta \in \mathbf{\Delta}$, such that

$$\sum_{l=1}^{M} \alpha_l \bar{B}_l \Delta \bar{C}_\sigma = \tilde{A}_{\sigma,h} - \sum_{l=1}^{M} \alpha_l \bar{A}_{\sigma,l}. \tag{18}$$

Hence, we should determine the worst-case distance between the real system (7) and the polytopic system (17), leading to an upper bound of the approximation error, see Fig. 1. To obtain a tight bound, we construct different uncertainty bounds between each two grid points. Indeed, for each two grid points $\tilde{h}_l, \tilde{h}_{l+1}$, we compare for $h \in [\tilde{h}_l, \tilde{h}_{l+1}]$, $\tilde{A}_{\sigma_k,h}$ with $\{\tilde{\alpha}\bar{A}_{\sigma_k,l} + (1-\tilde{\alpha})\bar{A}_{\sigma_k,l+1} \mid \alpha \in [0,1]\}$ and compute the worst-case bound between them for all $h \in [\tilde{h}_l, \tilde{h}_{l+1}]$. Finally, we will scale all these bound to get a common additive uncertainty set $\mathbf{\Delta}$.

This procedure is formalised in the theorem below. For ease of exposition, we will focus on the case where A is diagonalisable with real eigenvalues only. The procedure above also applies for general A, using the real Jordan form, although, in these cases, the structure of $\mathbf{\Delta}$ is different than indicated below in (23).

Theorem 1. *Let the NCS model (7) be given with $h \in [\underline{h}, \overline{h}]$ and $A := T\Lambda T^{-1}$ for some invertible matrix $T \in \mathbb{R}^{n_x \times n_x}$ and $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_{n_x})$ with $\lambda_i \in \mathbb{R}$, $i \in \{1, \ldots, n_x\}$. Furthermore, consider the system (14) in which $\bar{A}_{\sigma,l} := \tilde{A}_{\sigma,\tilde{h}_l}$, $l \in \{1, \ldots, M\}$, is obtained by evaluating (8) at M distinct transmission intervals $\{\tilde{h}_1, \ldots, \tilde{h}_M\}$, with $\underline{h} =: \tilde{h}_0 \leq \tilde{h}_1 < \ldots < \tilde{h}_M \leq \tilde{h}_{M+1} := \overline{h}$. Moreover,*

$$\bar{C}_\sigma := \begin{bmatrix} T^{-1} & 0 & 0 & 0 \\ T^{-1}BD_cC & T^{-1}BC_c & T^{-1}BD_c & T^{-1}B(I - \Gamma_\sigma^u) \end{bmatrix} \tag{19}$$

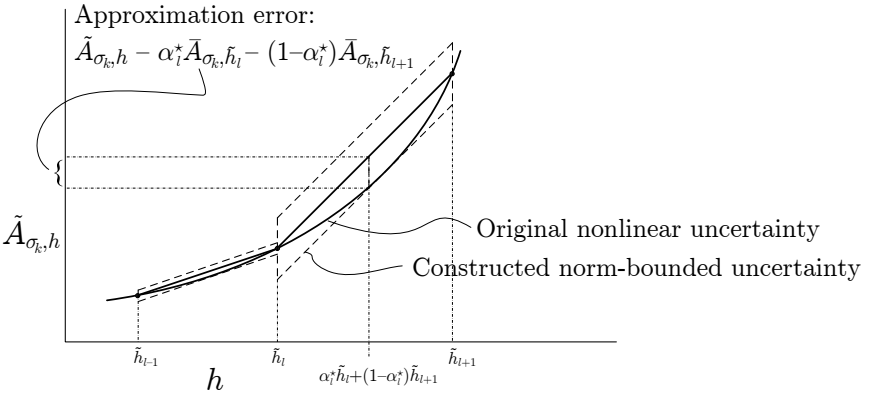


Fig. 1. The procedure of obtaining the overapproximation

and

$$\bar{B}_l := \begin{bmatrix} T & T \\ 0 & 0 \\ -CT & -CT \\ 0 & 0 \end{bmatrix} \cdot \operatorname{diag}\left(\max\{\delta^\star_{1,l}, \delta^\star_{1,l+1}\}, \ldots, \max\{\delta^\star_{2n_x,l}, \delta^\star_{2n_x,l+1}\}\right) \tag{20}$$

in which

$$\delta^\star_{i,l} = \begin{cases} \sup\limits_{h\in[\tilde{h}_{l-1},\tilde{h}_l]} |e^{\lambda_i h} - \alpha_h^\star e^{\lambda_i \tilde{h}_{l-1}} + (\alpha_h^\star - 1)e^{\lambda_i \tilde{h}_l}|, & \text{if } \ 1 \le i \le n_x \\ \sup\limits_{h\in[\tilde{h}_{l-1},\tilde{h}_l]} |\int_{\tilde{h}_l}^{h} e^{\lambda_{i-n_x} s} ds + \alpha_h^\star \int_{\tilde{h}_{l-1}}^{\tilde{h}_l} e^{\lambda_{i-n_x} s} ds|, & \text{if } \ n_x + 1 \le i \le 2n_x, \end{cases} \tag{21}$$

for each $l \in \{1, \ldots, M+1\}$ *and* $\alpha_h^\star$ *is given for* $h \in [\tilde{h}_{l-1}, \tilde{h}_l]$ *by*

$$\alpha_h^\star = \arg \inf_{\tilde{\alpha}\in[0,1]} \| \begin{bmatrix} e^{\Lambda h} - \tilde{\alpha} e^{\Lambda \tilde{h}_{l-1}} + (\tilde{\alpha} - 1)e^{\Lambda \tilde{h}_l} & 0 \\ 0 & \int_{\tilde{h}_l}^{h} e^{\Lambda s} ds + \tilde{\alpha} \int_{\tilde{h}_{l-1}}^{\tilde{h}_l} e^{\Lambda s} ds \end{bmatrix} \|. \tag{22}$$

The additive uncertainty set is given by

$$\boldsymbol{\Delta} := \left\{ diag\,(\delta_1, \ldots, \delta_{2n_x}) \in \mathbb{R}^{2n_x \times 2n_x} \ \mid \ \delta_i \in [-1, 1] \right\}. \tag{23}$$

Then, (7) *holds meaning that* (14) *is an overapproximation of* (7).

Proof. The proof is omitted for the sake of brevity, but can be found in the technical report [23]. □

The stability of (7) with (9) or (10), where $h_k \in [\underline{h}, \overline{h}]$, can now be guaranteed by proving stability of (14) with $\alpha_k \in \mathcal{A}$, $\Delta_k \in \boldsymbol{\Delta}$, $k \in \mathbb{N}$, as (14) is an overapproximation of (7).

Remark 3. In case of a continuous-time controller as in Remark 2, a similar procedure applies. ■

3 Stability of Switched Systems with Parametric Uncertainty

In the previous section, we discussed the NCS model and introduced an effective way to overapproximate it by a switched polytopic system with a norm-bounded uncertainty. Given this uncertain switched system, we can analyse whether a switching sequence, as induced by a protocol, renders the switched system UGES.

We will start with so-called quadratic protocols that include the well-known TOD protocol as a particular case. The analysis is based on extensions of ideas in [24], in which only switched linear systems without any form of uncertainty is considered. Hence, extensions are needed to include switched polytopic systems with norm-bounded uncertainties as in (14). After the stability analysis for quadratic and the TOD protocols, we show how we can analyse stability for the RR protocol.

For proving stability of system (14), we will employ the so-called full block S-procedure [25], which is presented in the following lemma.

Lemma 1 (Full block S-procedure). *Let $\bar{P}$ be given and let*

$$\bar{\mathbf{\Delta}} := \left\{ \Delta \mid \begin{bmatrix} \Delta \\ I \end{bmatrix}^\top \begin{bmatrix} Q & S \\ S^\top & R \end{bmatrix} \begin{bmatrix} \Delta \\ I \end{bmatrix} \succeq 0 \right\} \tag{24}$$

for some matrices $Q = Q^\top$, S, and $R = R^\top \succ 0$ of appropriate dimensions. Then, the following statements are equivalent:

1. $$\begin{bmatrix} I & 0 \\ \bar{A} & \bar{B} \end{bmatrix}^\top \bar{P} \begin{bmatrix} I & 0 \\ \bar{A} & \bar{B} \end{bmatrix} + \begin{bmatrix} 0 & I \\ \bar{C} & 0 \end{bmatrix}^\top \begin{bmatrix} Q & S \\ S^\top & R \end{bmatrix} \begin{bmatrix} 0 & I \\ \bar{C} & 0 \end{bmatrix} \prec 0. \tag{25}$$
2. *For all $\bar{\Delta} \in \bar{\mathbf{\Delta}}$, it holds that*
$$\begin{bmatrix} I \\ \bar{A} + \bar{B}\bar{\Delta}\bar{C} \end{bmatrix}^\top \bar{P} \begin{bmatrix} I \\ \bar{A} + \bar{B}\bar{\Delta}\bar{C} \end{bmatrix} \prec 0. \tag{26}$$

By choosing a suitable $\bar{P}$, (26) can lead to a sufficient condition for stability of (14), as we will show later. To use this result we aim at constructing the matrices Q, S, and R such that the actual additive uncertainty set given by $\mathbf{\Delta}$ as in (23) is equal to $\bar{\mathbf{\Delta}}$ as in (24).

Lemma 2. *Consider $\mathbf{\Delta}$ as in (23). If*

$$\begin{bmatrix} Q & S \\ S^\top & R \end{bmatrix} = \begin{bmatrix} -R & 0 \\ 0 & R \end{bmatrix} \quad \text{with} \quad R \in \mathcal{R} = \{ \operatorname{diag}(r_1, \ldots, r_m) \mid r_i > 0 \}, \tag{27}$$

then $\bar{\mathbf{\Delta}}$ as in (24) is equal to $\mathbf{\Delta}$ i.e., $\mathbf{\Delta} = \bar{\mathbf{\Delta}}$.

Proof. It follows by direct calculation, exploiting the diagonal structure of (23). □

3.1 Quadratic Protocols

In this section, we assume that the switching function is given by

$$\sigma_k = \arg \min_{i=1,\dots,N} \bar{x}_k^\top P_i \bar{x}_k, \tag{28}$$

where P_i with $i \in \{1,\dots,N\}$ are certain given positive definite matrices. We call protocols of the form (28) *quadratic* protocols. We will show later that the TOD protocol is actually a special case of this type of protocols. To analyse stability of (14) having switching law (28), we introduce the non-quadratic Lyapunov function

$$V(\bar{x}_k) = \min_{i=1,\dots,N} \bar{x}_k^\top P_i \bar{x}_k = \min_{\nu \in \mathcal{N}} \bar{x}_k^\top \sum_{i=1}^{N} \nu_i P_i \bar{x}_k, \tag{29}$$

where

$$\mathcal{N} := \left\{ \nu \in \mathbb{R}^N \;\middle|\; \sum_{i=1}^{N} \nu_i = 1, \nu_i \geq 0 \right\}. \tag{30}$$

Furthermore, we introduce the class of so-called Metzler matrices given by

$$\mathcal{M} := \left\{ \Pi \in \mathbb{R}^{N \times N} \;\middle|\; \sum_{j=1}^{N} \pi_{ji} = 1, \pi_{ji} \geq 0 \right\}. \tag{31}$$

The main result of this section is presented in the following theorem.

Theorem 2. *Assume that there exist a matrix $\Pi \in \mathcal{M}$, a set of positive definite matrices $\{P_1,\dots,P_N\}$, and a set of positive definite diagonal matrices $\{R_{1,1},\dots,R_{N,1},\dots,R_{1,M},\dots,R_{N,M}\}$, with $R_{i,l} \in \mathcal{R}$, with $\mathcal{R}$ the set of diagonal matrices as in (27), satisfying*

$$\begin{bmatrix} \bar{A}_{i,l}^\top \sum_{j=1}^{N} \pi_{ji} P_j \bar{A}_{i,l} - P_i + \bar{C}_i^\top R_{i,l} \bar{C}_i & \bar{A}_{i,l}^\top \sum_{j=1}^{N} \pi_{ji} P_j \bar{B}_l \\ \bar{B}_l^\top \sum_{j=1}^{N} \pi_{ji} P_j \bar{A}_{i,l} & \bar{B}_l^\top \sum_{j=1}^{N} \pi_{ji} P_j \bar{B}_l - R_{i,l} \end{bmatrix} \prec 0, \tag{32}$$

for all $i \in \{1,\dots,N\}$ and $l \in \{1,\dots,M\}$. Then, the switching law (28) renders the system (14) UGES. Consequently, the NCS (7) is also UGES if the switching law (28) is employed as the protocol.

Proof. The proof is based on showing that $V(\bar{x}_k)$ as in (29) is a Lyapunov function for the switched uncertain system (14) with switching law (28). Note that $V(\bar{x}_k) = \bar{x}_k^\top P_i \bar{x}_k$, with $\sigma_k = i$, due to (28). Now, we obtain using (29) and (14) that

$$V(\bar{x}_{k+1}) = \min_{\nu \in \mathcal{N}} \bar{x}_{k+1}^\top \sum_{j=1}^{N} \nu_j P_j \bar{x}_{k+1} \leq \bar{x}_{k+1}^\top \sum_{j=1}^{N} \pi_{ji} P_j \bar{x}_{k+1} =$$

$$\sum_{l_1=1}^{M} \alpha_{k,l_1} \bar{x}_k^\top \left(\bar{A}_{i,l_1} + \bar{B}_{l_1} \Delta_k \bar{C}_i\right)^\top \sum_{j=1}^{N} \pi_{ji} P_j \sum_{l_2=1}^{M} \alpha_{k,l_2} \left(\bar{A}_{i,l_2} + \bar{B}_{l_2} \Delta_k \bar{C}_i\right) \bar{x}_k. \tag{33}$$

UGES is now implied by requiring that the Lyapunov function is strictly decreasing in the sense that (due to (33))

$$\sum_{l_1=1}^{M} \alpha_{k,l_1} \left(\bar{A}_{i,l_1} + \bar{B}_{l_1} \Delta \bar{C}_i\right)^\top \sum_{j=1}^{N} \pi_{ji} P_j \sum_{l_2=1}^{M} \alpha_{k,l_2} \left(\bar{A}_{i,l_2} + \bar{B}_{l_2} \Delta \bar{C}_i\right) - P_i \prec 0. \quad (34)$$

for all $i \in \{1, \ldots, N\}$. By taking a Schur complement, and realising that $\sum_{j=1}^{N} \pi_{ji} P_j \succ 0$, we obtain that (34) is equivalent to

$$\sum_{l=1}^{M} \alpha_l \underbrace{\begin{bmatrix} P_i & \left(\bar{A}_{i,l} + \bar{B}_l \Delta \bar{C}_i\right)^\top \sum_{j=1}^{N} \pi_{ji} P_j \\ \sum_{j=1}^{N} \pi_{ji} P_j \left(\bar{A}_{i,l} + \bar{B}_l \Delta \bar{C}_i\right) & \sum_{j=1}^{N} \pi_{ji} P_j \end{bmatrix}}_{G_{i,l}} \succ 0 \quad (35)$$

for all $i \in \{1, \ldots, N\}$. A sufficient condition for the satisfaction of (35) is that $G_{i,l} \succ 0$ for all $i \in \{1, \ldots, N\}$ and $l \in \{1, \ldots, M\}$. Using again a Schur complement, we can rewrite the condition $G_{i,l} \succ 0$ as follows:

$$P_i - \left(\bar{A}_{i,l} + \bar{B}_l \Delta \bar{C}_i\right)^\top \sum_{j=1}^{N} \pi_{ji} P_j \left(\bar{A}_{i,l} + \bar{B}_l \Delta \bar{C}_i\right) \succ 0 \quad (36)$$

or equivalently,

$$\begin{bmatrix} I \\ \bar{A}_{i,l} + \bar{B}_l \Delta \bar{C}_i \end{bmatrix}^\top \begin{bmatrix} -P_i & 0 \\ 0 & \sum_{j=1}^{N} \pi_{ji} P_j \end{bmatrix} \begin{bmatrix} I \\ \bar{A}_{i,l} + \bar{B}_l \Delta \bar{C}_i \end{bmatrix} \prec 0, \quad (37)$$

for all $i \in \{1, \ldots, N\}$ and $l \in \{1, \ldots, M\}$. As (37) has the form of (26) of Lemma 1, it can, therefore, be rewritten in a form equivalent to (25) in which we use (27). This yields (32) for all $i \in \{1, \ldots, N\}$ and all $l \in \{1, \ldots, M\}$. Hence, we can conclude that $V(\bar{x}_k)$ is strictly decreasing in spite of the presence of the uncertainty if the inequalities (32) are feasible. Standard Lyapunov-based stability arguments now prove that (14) with (28) is UGES. □

Remark 4. The results of Theorem 2 can be exploited in two ways: (i) *For the design of a stabilising protocol.* Then the conditions in (32) are not LMIs, but Bilinear Matrix Inequalities (BMIs) due to the presence of the product of π_{ji} and P_j. Although literature on solving BMIs is available, see, e.g., [26,27,28], solving BMIs is considered to be of a high numerical complexity. If the number of nodes is relatively small, one way to proceed is gridding the possible solutions in $\Pi \in \mathcal{M}$, and subsequently solving the resulting LMIs. (ii) *Stability analysis for a given protocol.* In the situation that the set of matrices $\{P_1, \ldots, P_N\}$ is completely dictated by a particular quadratic protocol, the conditions (32) are LMIs. ■

3.2 The TOD Protocol

In this section, we will show that the TOD protocol is a special case of the class of quadratic protocols and thus that the Lyapunov-Metzler inequalities can be

employed to determine the allowable range of transmission intervals of the NCS using the TOD protocol as well. Since the switching sequence is given by (28), we can arrive at the TOD protocol by adopting the following structure in the P_i matrices:

$$P_i = \bar{P} + \begin{bmatrix} 0 & 0 \\ 0 & \tilde{P}_i \end{bmatrix}. \tag{38}$$

Each $\tilde{P}_i \in \mathbb{R}^{(n_y+n_u)\times(n_y+n_u)}$ is partitioned according to the partitioning of the nodes in the sense that

$$\tilde{P}_i \in \{\mathrm{diag}\,(-I_1, 0_2, \ldots, 0_N), \ldots, \mathrm{diag}\,(0_1, \ldots, 0_{N-1}, -I_N)\}, \tag{39}$$

where I_i, $i = 1, \ldots, N$, are identity matrices and 0_i, $i = 1, \ldots, N$, are null matrices, both having dimensions $\mathbb{R}^{n_i \times n_i}$ with n_i corresponding to the number of actuators or sensors in node i. Indeed, this structure implies that (28) becomes

$$\sigma_k = \arg\min \left\{-\|e_k^1\|^2, \ldots, -\|e_k^N\|^2\right\} = \arg\max \left\{\|e_k^1\|, \ldots, \|e_k^N\|\right\} \tag{40}$$

which is exactly the TOD protocol as described by (9). This proves that the TOD protocol can be regarded as a special case of the class of quadratic protocols. Therefore, stability of the NCS with the TOD protocol can be analysed using Theorem 2.

3.3 The RR Protocol

We will analyse an other well-known communication protocol, namely the RR protocol. Therefore, we need to analyse stability of the system (14) with a switching sequence induced by (10). This system is essentially a periodic uncertain system with period N. For this system, we introduce a set of positive definite matrices $\{P_1, \ldots, P_N\}$ and a mode-dependent Lyapunov function given by $V_{\sigma_k}(\bar{x}_k) = \bar{x}_k^\top P_{\sigma_k} \bar{x}_k$. We can now present the main result of this section.

Theorem 3. *Assume that there exist a set of positive definite matrices* $\{P_1, \ldots, P_N\}$ *and a set of positive definite diagonal matrices* $\{R_{1,1}, \ldots, R_{N,1}, \ldots R_{1,M}, \ldots, R_{N,M}\}$, *with* $R_{i,l} \in \mathcal{R}$ *with* $\mathcal{R}$ *as in* (27), *satisfying*

$$\begin{bmatrix} \bar{A}_{i,l}^\top P_{i+1} \bar{A}_{i,l} - P_i + \bar{C}_i^\top R_{i,l} \bar{C}_i & \bar{A}_{i,l}^\top P_{i+1} \bar{B}_l \\ \bar{B}_l^\top P_{i+1} \bar{A}_{i,l} & \bar{B}_l^\top P_{i+1} \bar{B}_l - R_{i,l} \end{bmatrix} \prec 0, \tag{41}$$

where $P_{N+1} := P_1$, *for all* $i \in \{1, \ldots, N\}$ *and* $l \in \{1, \ldots, M\}$. *Then, the system* (14) *with* (10) *is UGES and consequently, the NCS* (7) *with* (10) *is UGES.*

Proof. The proof follows the same lines as the proof of Theorem 2. □

4 Illustrative Example

In this section, we illustrate the usefulness of the presented theory using a well-known benchmark example in the NCS literature [10,14,19], consisting of a model

Table 1. Allowable Range of Transmission Intervals

Method	**Range**
Simulation based, obtained in [10]	$h_k \in (\varepsilon, 0.06]$
Theoretical, obtained in [10]	$h_k \in (\varepsilon, 10^{-5}]$
Theoretical, obtained in [14]	$h_k \in (\varepsilon, 0.01]$
Theoretical, obtained in [16]	$h_k \in (\varepsilon, 0.0108]$
Newly obtained theoretical bound	$h_k \in [0.001, 0.032]$

of a batch reactor. First, we will analyse the continuous-time controller as also used in [10,14]. This will show that our results provide less conservative bounds on the uncertain transmission intervals than earlier results in the literature. Secondly, we show that our framework can also deal with discrete-time controllers. For both examples, we consider the TOD protocol.

The details of the linearised model of the batch reactor model used in this example and the continuous-time controller can be found in [10,14,19]. As in these references, we assume here that the controller is directly connected to the actuator and that only the two outputs are transmitted via the network. Hence, we have $N = 2$ nodes. Therefore, we have $\mathcal{G} = \{\text{diag}(1,0),\ \text{diag}(0,1)\}$, as defined in Section 2.1.

4.1 Continuous-Time Controller

In order to assess the bounds on the allowable transmission intervals, we first obtain the uncertain polytopic system (14) that overapproximates the NCS model (7). In this example we choose to grid at $\tilde{h}_l \in \{0.001,\ 0.004,\ 0.015,\ 0.032\}$ and determine an upper bound on the approximation error as in Theorem 1. Now we check the matrix inequalities (32) in Theorem 2, using the structure of the P_i-matrices as in (38).

Using this procedure we obtain a feasible solution to (32) on the basis of which we conclude that the TOD protocol stabilises the NCS for any transmission interval between $h \in [10^{-3},\ 0.032]$. In Table 1, we compare our results with the existing results in [10,14,16]. The results in [10,14,16] can guarantee UGES for the given ranges of Table 1, where $\varepsilon > 0$ can be arbitrary small. We can conclude that taking $\underline{h} = 10^{-3}$ as a lower bound on the transmission intervals leads to a guaranteed MATI $\overline{h} = 0.032$, which is much larger than the recently obtained results. The real MATI was estimated to be 0.06 in [10], hence, we are getting closer to this estimate.

4.2 Discrete-Time Controller

Next, we compute $[\underline{h}, \overline{h}]$ for the NCS given a discrete-time controller. The discrete-time controller is obtained by discretising the continuous-time controller (12) with the matrices given in [10,14,19] by using a zero order hold, assuming a fixed sample

time of 0.003. Following the procedure presented in this paper, we conclude that this controller stabilises the NCS using the TOD protocol if $h_k \in [0.001, 0.032]$. Hence, the bound $\overline{h} = 0.032$ of the continuous-time controller can also be guaranteed by a discrete-time equivalent of the controller. Of course, a discrete-time controller has the advantage over the continuous-time controller that it is much easier to implement.

5 Conclusions

In this paper, we studied the stability of Networked Control Systems (NCSs) that are subject to communication constraints and time-varying transmission intervals. These communications constraints impose that per transmission, only one node can access the network and send its information. We analysed the stability of the NCS when the communication sequence is determined by the Round Robin (RR), the Try-Once-Discard (TOD) or a quadratic protocol. This analysis was based on a discrete-time switched uncertain linear system to describe the NCS. A new and efficient convex overapproximation was proposed that allowed us to analyse stability using a finite number of matrix inequalities. On a benchmark example, we illustrated the effectiveness of the theory. In particular, we showed that if the minimum allowable transmission interval is not infinitesimally small, stability can be guaranteed for a much larger maximum allowable transmission interval, when compared to the existing results in the literature. Interestingly, our results can be applied to both continuous-time and discrete-time controllers.

References

1. Hespanha, J., Naghshtabrizi, P., Xu, Y.: A survey of recent results in networked control systems. Proc. of the IEEE, 138–162 (2007)
2. Zhang, W., Branicky, M., Phillips, S.: Stability of networked control systems. IEEE Control Systems Magazine 21(1), 84–99 (2001)
3. Tipsuwan, Y., Chow, M.-Y.: Control methodologies in networked control systems. Control Engineering Practice 11, 1099–1111 (2003)
4. Yang, T.C.: Networked control system: a brief survey. IEE Proc. Control Theory & Applications 153(4), 403–412 (2006)
5. Nešić, D., Liberzon, D.: A unified approach to controller design for systems with quantization and time scheduling. In: Proc. of the 46th IEEE Conf. on Decision and Control, pp. 3939–3944 (2007)
6. Liberzon, D.: Quantization, time delays, and nonlinear stabilization. IEEE Trans. on Autom. Control 51(7), 1190–1195 (2006)
7. Cloosterman, M.B.G., van de Wouw, N., Heemels, W.P.M.H., Nijmeijer, H.: Stability of networked control systems with large delays. In: Proc. of the 46th IEEE Conf. on Decision and Control, pp. 5017–5022 (2007)
8. Hetel, L., Daafouz, J., Iung, C.: Stabilization of arbitrary switched linear systems with unknown time-varying delays. IEEE Trans. on Autom. Control 51(10), 1668–1674 (2006)

9. Naghshtabrizi, P., Hespanha, J.P.: Stability of networked control systems with variable sampling and delay. In: 44th Allerton Conf. on Communications, Control, and Computing (2006)
10. Walsh, G.C., Ye, H., Bushnell, L.G.: Stability analysis of networked control systems. IEEE Trans. on Control Systems Technology 10(3), 438–446 (2002)
11. Brockett, R.: Stabilization of motor networks. In: Proc. of the 34th IEEE Conf. on Decision and Control, vol. 2, pp. 1484–1488 (1995)
12. Hristu, D., Morgansen, K.: Limited communication control. Systems & Control Letters 37(4), 193–205 (1999)
13. Rehbinder, H., Sanfridson, M.: Scheduling of a limited communication channel for optimal control. Automatica 40(3), 491–500 (2004)
14. Nešić, D., Teel, A.: Input-output stability properties of networked control systems. IEEE Trans. on Autom. Control 49(10), 1650–1667 (2004)
15. Walsh, G., Belidman, O., Bushnell, L.: Asymptotic behavior of nonlinear networked control systems. IEEE Trans. on Autom. Control 46, 1093–1097 (2001)
16. Carnevale, D., Teel, A., Nešić, D.: A Lyapunov proof of improved maximum allowable transfer interval for networked control systems. IEEE Trans. on Autom. Control 52, 892–897 (2007)
17. Tabbara, M., Nešić, D., Teel, A.: Stability of wireless and wireline networked control systems. IEEE Trans. on Autom. Control 52(9), 1615–1630 (2007)
18. Goebel, R., Teel, A.: Solution to hybrid inclusions via set and graphical convergence with stability theory applications. Automatica 42, 573–587 (2006)
19. Dačić, D.B., Nešić, D.: Quadratic stabilization of linear networked control systems via simultaneous protocol and controller design. Automatica 43(7), 1145–1155 (2007)
20. Fujioka, H.: Stability analysis for a class of networked/embedded control systems: A discrete-time approach. In: Proc of the American Control Conf., pp. 4997–5002 (2008)
21. Hetel, L., Daafouz, J., Iung, C.: LMI control design for a class of exponential uncertain systems with application to network controlled switched systems. In: Proc. of the American Control Conf., pp. 1401–1406 (2007)
22. Cloosterman, M.B.G., van de Wouw, N., Heemels, W.P.M.H., Nijmeijer, H.: Robust stability of networked control systems with time-varying network-induced delays. In: Proc. of the 45th IEEE Conf. on Decision and Control, pp. 4980–4985 (2006)
23. Donkers, M., Hetel, L., Heemels, W., van de Wouw, N., Steinbuch, M.: Stability analysis of networked control systems using a switched linear systems approach. Technical report, Eindhoven University of Technology, DCT 2009-003 (2009)
24. Geromel, J.C., Colaneri, P.: Stability and stabilization of discrete time switched systems. Int. Journal of Control 79(7), 719–728 (2006)
25. Scherer, C.W.: 10. In: Robust mixed control and LPV control with full block scalings, Niculescu. Volume Advances in Design and Control, Niculescu, pp. 187–207. Springer, Heidelberg (1999)
26. Goh, K.-C., Safonov, M.G., Papavassilopoulos, G.P.: A global optimization approach for the BMI problem. In: Proc. of the 33rd IEEE Conf. on Decision and Control, vol. 3, pp. 2009–2014 (1994)
27. Hassibi, A., How, J., Boyd, S.: A path-following method for solving BMI problems in control. In: Proc. of the American Control Conf., vol. 2, pp. 1385–1389 (1999)
28. Iwasaki, T., Skelton, R.E.: The XY-centring algorithm for the dual LMI problem: a new approach to fixed-order control design. Int. Journal of Control 62(6), 1257–1272 (1995)

Parameter Synthesis for Hybrid Systems with an Application to Simulink Models

Alexandre Donzé[1], Bruce Krogh[2], and Akshay Rajhans[2]

[1] Dept. of Computer Science
[2] Dept. of Electrical and Computer Engineering,
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213, USA
adonze@cs.cmu.edu, {krogh,arahjans}@ece.cmu.edu

Abstract. This paper addresses a parameter synthesis problem for nonlinear hybrid systems. Considering a set of uncertain parameters and a safety property, we give an algorithm that returns a partition of the set of parameters into subsets classified as safe, unsafe, or uncertain, depending on whether respectively all, none, or some of their behaviors satisfy the safety property. We make use of sensitivity analysis to compute approximations of reachable sets and an error control mechanism to determine the size of the partition elements in order to obtain the desired precision. We apply the technique to Simulink models by combining generated code with a numerical solver that can compute sensitivities to parameter variations. We present experimental results on a non-trivial Simulink model of a quadrotor helicopter.

1 Introduction

A standard problem in model-based analysis and design is to find the ranges of parameters (including initial states) for which the system behavior will be acceptable [HWT96, FJK08]. We call this the *parameter synthesis problem*. One approach to this problem is to run simulations of the system for a set of parameter values that covers the range of values of interest. This approach is attractive because of its generality: one can simulate almost any system. It can take a very large number of simulations to cover the parameter space at a sufficient level of granularity, however.

Reachability analysis offers an alternative to simulation [ADF+06]. By computing reachable sets rather than simulating single trajectories, it may be possible to explore the design space more efficiently. Although this is the case for low-dimensional systems, the ability to perform reachability computations for higher-dimensional systems remains an elusive goal, even for so-called linear hybrid automata [HHWT97, Fre05]. The parameter synthesis problem for nonlinear and higher-dimensional systems remains intractable using reachability tools.

This paper proposes an approach to the parameter synthesis problem that offers the strength of reachability analysis while using only numerical simulations. This approach is in the spirit of other work on methods for obtaining

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 165–179, 2009.

reachable set information from single simulation runs [GP06, DM07, LKCK08]. Most of this work has focused on using simulations to propagate representations of reachable sets that are guaranteed to be conservative [GP06, LKCK08]. We use a different approach that leverages the simplicity of sensitivity analysis to generate approximations to reachable sets very efficiently [DM07]. Speed is achieved with a slight sacrifice in accuracy–the reachable set approximations are not guaranteed to be conservative. We are able to estimate the error in the approximations, however, providing a mechanism for gaining some assurance that the final estimation of the set of good parameters is reasonable.

The paper is organized as follows. The following section introduces notation and the basic algorithm for simulating hybrid dynamic systems. Section 3 recalls the method for generating sensitivity matrices with only a slight increase in computation during the simulation run. Section 4 presents the formulation and solution of the parameter synthesis problem using sensitivity-based reachability. We describe an implementation of the approach in Section 5 that solves the parameter synthesis problem for hybrid systems modeled in MATLAB Simulink and illustrate its application to the design of a supervisory safety control algorithm for a quadrotor helicopter. The concluding section summarizes the contributions of the paper and identifies directions for future research.

2 Hybrid Model and Simulation

The set $\mathbb{R}^n$ is equipped with the infinity norm, noted $\|\mathbf{x}\| = \max_i |x_i|$. It is extended to $n \times n$ matrices as usual. We define the diameter of a compact set $\mathcal{P}$ to be $\|\mathcal{P}\| = \sup_{(\mathbf{p},\mathbf{p}')\in\mathcal{P}^2} \|\mathbf{p}-\mathbf{p}'\|$. The distance from $\mathbf{x}$ to a set $\mathcal{R}$ is $d(\mathbf{x},\mathcal{R}) = \inf_{\mathbf{y}\in\mathcal{R}} \|\mathbf{x}-\mathbf{y}\|$. The Hausdorff distance between two sets $\mathcal{R}_1$ and $\mathcal{R}_2$ is

$$d_H(\mathcal{R}_1,\mathcal{R}_2) = \max(\sup_{\mathbf{x}_1\in\mathcal{R}_1} d(\mathbf{x}_1,\mathcal{R}_2), \sup_{\mathbf{x}_2\in\mathcal{R}_2} d(\mathbf{x}_2,\mathcal{R}_1)).$$

Given a matrix S and a set $\mathcal{P}$, $S\mathcal{P}$ represents the set $\{S\mathbf{p}, \mathbf{p}\in\mathcal{P}\}$. Given two sets $\mathcal{R}_1$ and $\mathcal{R}_2$, $\mathcal{R}_1 \oplus \mathcal{R}_2$ is the Minkowski sum of $\mathcal{R}_1$ and $\mathcal{R}_2$, i.e., $\mathcal{R}_1 \oplus \mathcal{R}_2 = \{\mathbf{x}_1+\mathbf{x}_2, \mathbf{x}_1\in\mathcal{R}_1, \mathbf{x}_2\in\mathcal{R}_2\}$.

2.1 Dynamics

We consider a dynamical system $\mathscr{S} = (\mathcal{Q}, f, e, g)$ with evolutions described by

$$\begin{cases} \dot{\mathbf{x}} = f(q,\mathbf{x},\mathbf{p}),\ \mathbf{x}(0)=\mathbf{x}_0 \\ q^+ = e(q^-,\lambda),\ q(0)=q_0 \\ \lambda = \text{sign}(g(\mathbf{x})) \end{cases} \tag{1}$$

where

– $\mathbf{x}\in\mathbb{R}^n$ is the *continuous state*, $\mathbf{p}$ is the *parameter vector* lying in a compact set $\mathcal{P}\subset\mathbb{R}^{n_p}$, $q\in\mathcal{Q}$ is the *discrete state*,

- λ is a vector in $\{-1, 0, +1\}^{n_g}$,
- g is the *guard function* mapping $\mathbb{R}^n$ to $\mathbb{R}^{n_g}$,
- sign is the usual sign function extended to vectors, i.e., if $\lambda = \text{sign}(g(\mathbf{x}))$, then $\lambda_i = 1$ if $g_i(\mathbf{x}) > 0$, $\lambda_i = -1$ if $g_i(\mathbf{x}) < 0$ and $\lambda_i = 0$ if $g_i(\mathbf{x}) = 0$.
- e is the *event function* which updates the discrete state when a component of the guard function g changes its sign. At each time t, q^+ (respectively q^-) represents the value of q immediatly after t (respectively immediatly before t). It is assumed that $q^+ = e(q^-, \lambda) = q^-$ if $\lambda_i \neq 0$ for all i. In words, q^+ may differ from q^- only when one component of the guard function g is zero.

Let $\mathcal{T} = \mathbb{R}^+$ be the *time set*. Given $\mathbf{p} \in \mathcal{P}$, a trajectory $\xi_\mathbf{p}$ is a function from $\mathcal{T}$ to $\mathbb{R}^n$ which satisfies (1), i.e., for all t in $\mathcal{T}$, we have

$$\dot{\xi}_\mathbf{p}(t) = f(q(t), \xi_\mathbf{p}(t), \mathbf{p}),\ q(t^+) = e(q(t^-), \lambda(t)) \text{ and } \lambda(t) = \text{sign}(g(\xi_\mathbf{p}(t)))$$

For convenience, the initial state $\mathbf{x}_0$ is included in the parameter vector $\mathbf{p}$. The dimension n_g of $\mathcal{P}$ is thus greater than n and we have $\xi_\mathbf{p}(0) = \mathbf{x}_0 = (x_{0_1}, x_{0_2}, \ldots, x_{0_n})$, where for all $i \leq n$, $x_{0_i} = p_i$.

In this work, we assume that a trajectory can always be computed by appending solutions of (1) on successive time intervals of the form $[t_k, t_{k+1}]$. This is possible if for all i, there is a neighborhood of $(t_k, \xi_\mathbf{p}(t_k))$ where $(t, \mathbf{x}) \mapsto f(q, \mathbf{x}, \mathbf{p})$ is continuously differentiable (C^1). In this case, we know by the Cauchy-Lipshitz theorem that there exists $h_k > 0$ such that a solution of the $\dot{\mathbf{x}} = f(q, \mathbf{x}, \mathbf{p})$ can be uniquely continued on the interval $(t_k, t_k + h_k]$. Thus t_{k+1} can be defined as $t_{k+1} = t_k + h_k$ and the process can be repeated indefinitely to form a unique trajectory on the whole time set $\mathcal{T}$ given a parameter vector $\mathbf{p}$.

2.2 Event Detection

In the model (1) above, the event function triggered by the guard function makes it possible to introduce discontinuities in the evolution. The function f can be discontinuous in a state $\mathbf{x}$ where some component of g is zero. Assume that $g_i(\xi_\mathbf{p}(\tau)) = 0$ for some i and $\tau > t_k$. It can thus be that

$$f(q(\tau^-), t^-, \xi_\mathbf{p}(\tau^-), \mathbf{p}) \neq f(q(\tau^+), t^+, \xi_\mathbf{p}(\tau^+), \mathbf{p})$$

This means that at time τ, the system switches from one continuous dynamics to another continuous dynamics, which is called a *switching event*. In such a situation, the Cauchy-Lipshtz theorem does not apply in $(\tau, \xi_\mathbf{p}(\tau))$ and standard numerical schemes may have problem to provide an accurate result. A solution is to integrate the dynamics until the time event τ, set $t_{k+1} = \tau$ and then continue from t_{k+1} with the new dynamics. Thus τ needs to be detected as precisely as possible, which can be done through *discontinuity locking* and *zero-crossing* detection [EKP01]. The idea is to fix (or *lock*) the value of q to $q(t_k)$ in order to prevent the occurrence of a switching event, and to integrate the equation $\dot{\mathbf{x}} = f_k(\mathbf{x}, \mathbf{p})$, where $f_k(\mathbf{x}, \mathbf{p}) = f(q(t_k), \mathbf{x}, \mathbf{p})$, on an interval $[t_k, t_k + h_k[$. Then check whether there is a time $t \in]t_k, t_k + h_k[$ such that the sign of g changed,

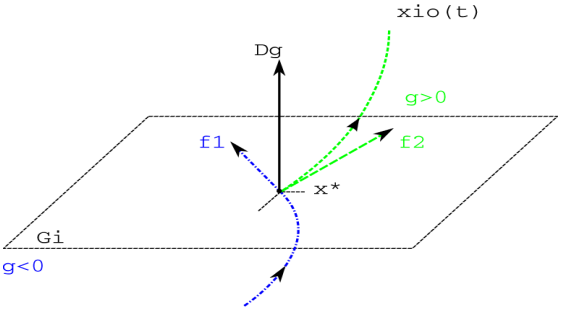
i.e., $\text{sign}(g(\mathbf{x}(t_k))) \neq \text{sign}(g(\mathbf{x}(t)))$, in which case, by continuity, it is guaranteed that g has at least one zero on the interval $]t_k, t[$. A bisection procedure can be applied to determine the first time τ when a component of g is zero.

This is summarized in the following algorithm to compute $\xi_{\mathbf{p}}(t_{k+1})$ knowing $\xi_{\mathbf{p}}(t_k)$.

1: Compute $\xi_{\mathbf{p}}$ solution of $\dot{\mathbf{x}} = f_k(\mathbf{x}, \mathbf{p})$ on $[t_k, t_k + h_k]$
2: **if** $\forall t \in [t_k, t_k + h_k]$, $\text{sign}(g(\xi_{\mathbf{p}}(t)) = \text{sign}(g(\xi_{\mathbf{p}}(t_k))$ **then**
3: Return $t_{k+1} = t_k + h_k$ and $\xi_{\mathbf{p}}(t_{k+1})$
4: **else**
5: Find the minimum time $\tau > t_k$ such that $g_i(\tau) = 0$ for some i
6: Return $t_{k+1} = \tau$ and $\xi_{\mathbf{p}}(t_{k+1}) = \xi_{\mathbf{p}}(\tau)$
7: **end if**

For the above algorithm to be correct, we make the following assumption: at the time τ of an event,

$$\langle \nabla g_i(\mathbf{x}(\tau)),\ f(q^-, \mathbf{x}(\tau^-), \mathbf{p}) \rangle \ \langle \nabla g_i(\mathbf{x}(\tau)),\ f(q^+, \mathbf{x}(\tau^+), \mathbf{p}) \rangle > 0. \tag{2}$$



This assumption, illustrated in the figure above, guarantees that when a transition occurs, the dynamics of the systems leans strictly toward the guard before the switch and strictly away from it after the transition. Thus we do not allow *sliding*, i.e., when a trajectory remains on the transition surface, nor *grazing*, i.e., when a trajectory hits the surface tangentially. This assumption holds for many real physical systems. Condition (2) can often be checked for all possible states in the model.

3 Sensitivity Analysis

The concept of *sensitivity to parameters* is a classical topic in the theory of dynamical systems. It is concerned with the question of the influence of a parameter change $\delta\mathbf{p}$ on a trajectory $\xi_{\mathbf{p}}$. A first order approximation of this influence can be obtained by a Taylor expansion of $\xi_{\mathbf{p}}(t)$ with respect to $\mathbf{p}$. For $\delta\mathbf{p} \in \mathbb{R}^{n_p}$, we have:

$$\xi_{\mathbf{p}+\delta\mathbf{p}}(t) = \xi_{\mathbf{p}}(t) + \frac{\partial \xi_{\mathbf{p}}}{\partial \mathbf{p}}(t)\ \delta\mathbf{p} + \varphi(t, \delta\mathbf{p}) \text{ where } \varphi(t, \delta\mathbf{p}) = \mathcal{O}\big(\|\delta\mathbf{p}\|^2\big) \tag{3}$$

The second term in the right hand side of (3) is the derivative of the trajectory with respect to the parameter $\mathbf{p}$. Since $\mathbf{p}$ is a vector, this derivative is a matrix called the *sensitivity matrix*, denoted as $S_{\mathbf{p}}(t) = \frac{\partial \xi_{\mathbf{p}}}{\partial \mathbf{p}}(t)$. By applying the chain rule to the time derivative of $\frac{\partial \xi_{\mathbf{p}}}{\partial \mathbf{p}}(t)$ we get

$$\dot{S}_{\mathbf{p}}(t) = \frac{\partial f}{\partial \mathbf{x}}(q, \xi_{\mathbf{p}}(t), \mathbf{p})\ S_{\mathbf{p}}(t) + \frac{\partial f}{\partial \mathbf{p}}(q, \xi_{\mathbf{p}}(t), \mathbf{p}) \tag{4}$$

Here $\frac{\partial f}{\partial \mathbf{x}}(q, \xi_{\mathbf{p}}(t), \mathbf{p})$ is the Jacobian matrix of f at the trajectory point at time t. This equation is thus an affine, time-varying ODE that, in the absence of discontinuity, can be solved in parallel with the ODE defining the dynamics (1).

When a trajectory switches from a mode q_1 to a mode q_2 due to the crossing of a surface given by $g_i(\mathbf{x}) = 0$, the dynamics of the system changes from $\dot{\mathbf{x}} = f_1(\mathbf{x}, \mathbf{p}) \triangleq f(q_1, \mathbf{x}, \mathbf{p})$ to $\dot{\mathbf{x}} = f_2(\mathbf{x}, \mathbf{p}) \triangleq f(q_2, \mathbf{x}, \mathbf{p})$. It follows that the evolution of the sensitivity matrix also changes from $\dot{S}_{\mathbf{p}} = \frac{\partial f_1}{\partial \mathbf{x}} S_{\mathbf{p}} + \frac{\partial f_1}{\partial \mathbf{p}}$ to $\dot{S}_{\mathbf{p}} = \frac{\partial f_2}{\partial \mathbf{x}} S_{\mathbf{p}} + \frac{\partial f_2}{\partial \mathbf{p}}$. Even though we do not consider resets in our models, i.e., the continuous state remains unaffected by the switching ($\mathbf{x}(\tau^-) = \mathbf{x}(\tau^+)$), the sensitivity matrix $S_{\mathbf{p}}$ can be discontinuous in τ. It can be shown that the jump condition, i.e. the difference between τ^- and τ^+ is given by [HP00]

$$S_{\mathbf{p}}(\tau^+) - S_{\mathbf{p}}(\tau^-) = \frac{d\tau}{d\mathbf{p}}\big(f_2(\tau, \mathbf{x}^*, \mathbf{p}) - f_1(\tau, \mathbf{x}^*, \mathbf{p})\big), \tag{5}$$

$$\text{where } \frac{d\tau}{d\mathbf{p}} = \frac{\langle \nabla_{\mathbf{x}} g_i(\mathbf{x}^*), S_{\mathbf{p}}(\tau) \rangle}{\langle \nabla_{\mathbf{x}} g_i(\mathbf{x}^*), f_1(\tau, \mathbf{x}^*, \mathbf{p}) \rangle}. \tag{6}$$

In [HP00], conditions for the computation of sensitivity matrices are given for hybrid models more general than ours. They include evolutions given by differential algebraic equations, state resets, etc. Since our technique relies on the ability to compute numerical simulations and sensitivity matrices, it means that it can be straigthfowardly extended to handle these systems.

4 Parameter Synthesis Algorithm

In this section, we consider an hybrid system $\mathscr{S} = (\mathcal{Q}, f, e, g)$, a compact set of parameters $\mathcal{P}$ and a set of so-called "bad" states, $\mathcal{B} \in \mathbb{R}^n$. Our goal is to partition $\mathcal{P}$ into *safe*, *unsafe* and *uncertain* subsets, defined as follows.

Definition 1 (Parameter Synthesis Problem)

- *A* parameter synthesis problem *is a 4-uple* $(\mathscr{S}, \mathcal{P}, \mathcal{B}, T)$ *where* $\mathscr{S}$ *is an hybrid system,* $\mathcal{P}$ *a compact set,* $\mathcal{B}$ *a set and* T *a non-negative real number;*
- *A* solution *of the parameter synthesis problem* $(\mathscr{S}, \mathcal{P}, \mathcal{B}, T)$ *is a partition of* $\mathcal{P}$ *into three sets* $(\mathcal{P}_{saf}, \mathcal{P}_{unc}, \mathcal{P}_{bad})$ *such that: for all* $\mathbf{p} \in \mathcal{P}_{bad}$, $\xi_{\mathbf{p}}(t) \in \mathcal{B}$ *for some* $0 \leq t \leq T$*; for all* $\mathbf{p} \in \mathcal{P}_{saf}$, $\xi_{\mathbf{p}}(t) \notin \mathcal{B}$ *for all* $0 \leq t \leq T$*; and* $\mathcal{P}_{unc} = \mathcal{P} - \mathcal{P}_{saf} \cup \mathcal{P}_{bad}$.

Solutions to the parameter synthesis problem can be defined in terms of *reachable sets*, which we define next.

Definition 2 (Reachable Set). *The reachable set induced by a set of parameters $\mathcal{P}$ at time t is $\mathcal{R}_t(\mathcal{P}) = \bigcup_{\mathbf{p}\in\mathcal{P}} \xi_{\mathbf{p}}(t)$*

It is clear that $(\mathcal{P}_{\text{saf}}, \mathcal{P}_{\text{unc}}, \mathcal{P}_{\text{bad}})$ is a solution of $(\mathscr{S}, \mathcal{P}, \mathcal{B}, T)$ if and only if $\mathcal{R}_t(\mathcal{P}_{\text{saf}}) \cap \mathcal{B} = \emptyset\ \forall 0 \leq t \leq T$ and $\mathcal{P}_{\text{bad}} = \cup_l \mathcal{P}_l$ where for each l, $\mathcal{R}_t(\mathcal{P}_l) \subset \mathcal{B}$ for some $0 \leq t \leq T$. To characterize the precision of a solution, we use the following definition:

Definition 3 (δp-precise solution). *A solution $(\mathcal{P}_{saf}, \mathcal{P}_{unc}, \mathcal{P}_{bad})$ is said to be $\delta\mathbf{p}$-precise either if $\mathcal{P}_{unc}$ is empty or if it can be decomposed into a finite number of sets $\mathcal{P}_{unc} = \mathcal{S}_1 \cup \mathcal{S}_2 \cup \ldots \cup \mathcal{S}_l$ such that for all j,*

- *The diameter of $\mathcal{S}_j$ is smaller than $\delta\mathbf{p}$, i.e., $\|\mathcal{S}_j\| \leq \delta\mathbf{p}$,*
- *The reachable set induced by $\mathcal{S}_j$ intersects with $\mathcal{B}$ for some $0 \leq t \leq T$, i.e., $\mathcal{R}_t(\mathcal{S}_j) \cap \mathcal{B} \neq \emptyset$,*
- *The reachable set induced by $\mathcal{S}_j$ is not a subset of $\mathcal{B}$ for any $0 \leq t \leq T$, i.e., $\mathcal{R}_t(\mathcal{S}_j) \not\subseteq \mathcal{B}$.*

Intuitively a $\delta\mathbf{p}$-precise solution covers the boundary between safe and unsafe parameters with a finite number of sets whose sizes are at most $\delta\mathbf{p}$. Also, by this definition, a solution for which the uncertain set is empty is $\delta\mathbf{p}$-precise for any $\delta\mathbf{p}$. In the remainder of this section, we present an algorithm that aims at computing a $\delta\mathbf{p}$-precise solution. The method is based on an iterative partitioning of the parameter space, the computation of reachable set estimates and their intersections with the bad set.

4.1 Reachable Set Estimation Using Sensitivity

For some subset $\mathcal{S}$ of $\mathcal{P}$, set $\mathcal{R}_t(\mathcal{S})$ can be approximated by using sensitivity analysis. Let $\mathbf{p}$ and $\mathbf{p}'$ be two parameter vectors in $\mathcal{S}$ and assume that we computed the trajectory $\xi_{\mathbf{p}}$ and the sensitivity matrix $S_{\mathbf{p}}$ at time t. Then we can use $\xi_{\mathbf{p}}(t)$ and $S_{\mathbf{p}}(t)$ to estimate $\xi_{\mathbf{p}'}(t)$. We denote this estimate by $\hat{\xi}^{\mathbf{p}}_{\mathbf{p}'}(t)$. The idea is to drop higher order terms in the Taylor expansion (3), which gives

$$\hat{\xi}^{\mathbf{p}}_{\mathbf{p}'}(t) = \xi_{\mathbf{p}}(t) + S_{\mathbf{p}}(t)(\mathbf{p}' - \mathbf{p}). \tag{7}$$

If we extend this estimate to all parameters $\mathbf{p}'$ in $\mathcal{S}$, we get the following estimate for the reachable set $\mathcal{R}_t(\mathcal{S})$:

$$\hat{\mathcal{R}}^{\mathbf{p}}_t(\mathcal{S}) = \bigcup_{\mathbf{p}'\in\mathcal{S}} \hat{\xi}_{\mathbf{p}'}(t) = \{\xi_{\mathbf{p}} - S_{\mathbf{p}}(t)\mathbf{p}\} \oplus S_{\mathbf{p}}(t)\mathcal{S} \tag{8}$$

Note that this is an affine transformation of the initial set $\mathcal{S}$ (see Fig. 1). As a particular situation, if the dynamics of the system is affine, the estimate is exact as there are no higher order terms in the Taylor expansion.

When the dynamics is nonlinear, $\hat{\mathcal{R}}^{\mathbf{p}}_t(\mathcal{S})$ is different from $\mathcal{R}_t(\mathcal{S})$. For instance, we have the following lemma.

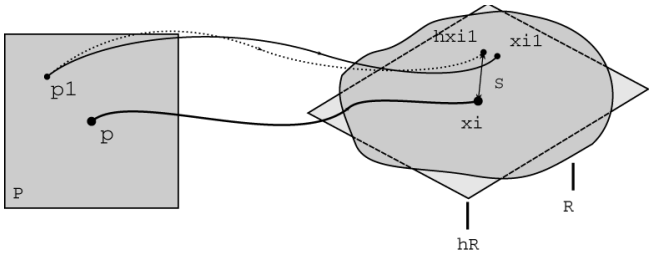


Fig. 1. Approximation of the reachable set using one trajectory and the corresponding sensitivity matrix

Lemma 1. *There exists a real number $K > 0$ such that*

$$d_H(\hat{\mathcal{R}}_t^{\mathbf{p}}(\mathcal{S}), \mathcal{R}_t(\mathcal{S})) \;\leq\; K\ \|\mathcal{S}\|^2.$$

Proof. Let $\mathbf{y}$ be in $\hat{\mathcal{R}}_t^{p}(\mathcal{S})$, $\mathbf{p_y} \in \mathcal{S}$ be such that $\mathbf{y} = \hat{\xi}^{p}_{\mathbf{p}_y}(t)$ and $\mathbf{x} = \xi_{\mathbf{p}_y}(t) \in \mathcal{R}_t(\mathcal{S})$. From (3) we have $\mathbf{x} - \mathbf{y} = \xi_{\mathbf{p}_y}(t) - \hat{\xi}^{\mathbf{p}}_{\mathbf{p}_y}(t) = \varphi(t, \mathbf{p}_y - \mathbf{p})$ where φ is a function such that $\varphi(t, \mathbf{p}_y - \mathbf{p}) = \mathcal{O}\big(\|\mathbf{p} - \mathbf{p}_y\|^2\big)$, meaning that we can find $K > 0$ such that $\|\mathbf{y} - \mathbf{x}\| = \|\xi_{\mathbf{p}_y}(t) - \hat{\xi}^{\mathbf{p}}_{\mathbf{p}_y}(t)\| \leq K\|\mathbf{p}_y - \mathbf{p}\|^2 \leq K\|\mathcal{S}\|^2$. Since this is true for any $\mathbf{y}$ in $\hat{\mathcal{R}}_t^{\mathbf{p}}(\mathcal{S})$, $\sup_{\mathbf{y}\in\hat{\mathcal{R}}_t^{\mathbf{p}}(\mathcal{S})} d(y, \mathcal{R}_t(\mathcal{S})) \leq K\|\mathcal{S}\|^2$. Similarly we can prove that $\sup_{\mathbf{x}\in\mathcal{R}_t(\mathcal{S})} d(x, \hat{\mathcal{R}}_t^{p}(\mathcal{S})) \leq K\|\mathcal{S}\|^2$ which implies the result. □

Thus, the error depends on the diameter of $\mathcal{S}$. In order to improve the estimation, we can partition $\mathcal{S}$ into smaller subsets $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_l$ and introduce new parameters, $\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_l$ to compute more precise local estimates. Then we need to be able to estimate the benefit of such a refinement. To do so, we can compare for each $\mathcal{S}_j$ the estimate $\hat{\mathcal{R}}_t^{\mathbf{p}}(\mathcal{S}_j)$ that we get using the "global" center $\mathbf{p}$ with the estimate $\hat{\mathcal{R}}_t^{\mathbf{p}_j}(\mathcal{S}_j)$ that we get when using the "local" center $\mathbf{p}_j$. We have the following result:

Proposition 1. *Let $\mathcal{S}_j$ be a subset of a parameter set $\mathcal{S}$. Let $\mathbf{p} \in \mathcal{S}$ and $\mathbf{p}_j \in \mathcal{S}_j$. Then*

$$d_H(\hat{\mathcal{R}}_t^{p}(\mathcal{S}_j), \hat{\mathcal{R}}_t^{p_j}(\mathcal{S}_j)) \leq Err(\mathcal{S}, \mathcal{S}_j) \tag{9}$$

where $Err(\mathcal{S}, \mathcal{S}_j) = \|\xi_{\mathbf{p}_j}(t) - \hat{\xi}^{\mathbf{p}}_{\mathbf{p}_j}(t)\| + \|S_{\mathbf{p}_j}(t) - S_{\mathbf{p}}(t)\|\|\mathcal{S}_j\|$.

Proof. Let $\mathbf{y}$ be in $\hat{\mathcal{R}}_t^{\mathbf{p}}(\mathcal{S}_j)$, $\mathbf{p}_y$ in $\mathcal{S}_j$ such that $\mathbf{y} = \hat{\xi}^{\mathbf{p}}_{\mathbf{p}_y}(t)$ and $\mathbf{x} = \hat{\xi}^{\mathbf{p}_j}_{\mathbf{p}_y}(t)$. We need to compare

$$\mathbf{y} = \xi_{\mathbf{p}}(t) + S_{\mathbf{p}}(t)(\mathbf{p}_y - \mathbf{p}) \text{ with } \mathbf{x} = \xi_{\mathbf{p}_j}(t) + S_{\mathbf{p}_j}(t)(\mathbf{p}_y - \mathbf{p}_j). \tag{10}$$

We introduce the quantity $\hat{\xi}^{\mathbf{p}}_{\mathbf{p}_j}(t) = \xi_{\mathbf{p}_j}(t) + S_{\mathbf{p}_j}(t)(\mathbf{p}'_j - \mathbf{p}_j)$ and with some algebraic manipulations of (10), we get

$$\hat{\xi}^{\mathbf{p}}_{\mathbf{p}'_j}(t) - \hat{\xi}^{\mathbf{p}_j}_{\mathbf{p}'_j}(t) = \xi_{\mathbf{p}_j}(t) - \hat{\xi}^{\mathbf{p}}_{\mathbf{p}_j}(t) + (S_{\mathbf{p}_j}(t) - S_{\mathbf{p}}(t))(\mathbf{p}'_j - \mathbf{p}_j)$$

which implies that $\|\mathbf{y} - \mathbf{x}\| \leq Err(\mathcal{S}, \mathcal{S}_j)$. The end of the proof is then similar to that of Lemma 1. □

As illustrated in Fig. 2, the difference between the global and the local estimate can thus be decomposed into the error in the estimate $\hat{\xi}^{\mathbf{P}}_{\mathbf{p}_j}(t)$ of the state reached at time t using $\mathbf{p}_j$ and another term involving the difference between the local and the global sensitivity matrices and the distance from local center. The quantity $Err(\mathcal{S}, \mathcal{S}_j)$ can be easily computed knowing the trajectory states $\xi_{\mathbf{p}}(t)$ and $\xi_{\mathbf{p}_j}(t)$ and their corresponding sensitivity matrices and from the diameter of $\mathcal{S}_j$.[1] Err has the following interesting properties:

- If the dynamics is affine, then $Err(\mathcal{S}, \mathcal{S}_j) = 0$. Indeed, in this case, $\hat{\xi}^{\mathbf{P}}_{\mathbf{p}_j} = \xi_{\mathbf{p}_j}$, so the first term vanishes and $\mathcal{S}_{\mathbf{p}} = \mathcal{S}_{\mathbf{p}_j}$ so the second term vanishes as well;
- If limit $\|\mathcal{S}\|$ is 0 then limit $Err(\mathcal{S}, \mathcal{S}_j)$ is also 0. Indeed, as $\|\mathcal{S}\|$ decreases, so does $\|\mathbf{p} - \mathbf{p}_j\|$ and thus $\|\xi_{\mathbf{p}_j}(t) - \hat{\xi}^{\mathbf{P}}_{\mathbf{p}_j}(t)\|$ and $\|\mathcal{S}_j\|$ since $\mathcal{S}_j$ is a subset of $\mathcal{S}$. Moreover, $Err(\mathcal{S}, \mathcal{S}_j) = \mathcal{O}\big(\|\mathcal{S}\|^2\big)$.

Thus we can compute a reachable set $\mathcal{R}_t(\mathcal{S})$ at a given time instant t and estimate the approximation error. To get an estimate $\mathcal{R}_{[0,T]}(\mathcal{S})$ on the interval $[0, T]$, one can do the computation for $t_0 = 0$, t_1, ... and $t_N = T$ and use some form of interpolation between t_k and t_{k+1}. This introduces additional error which depends on $|t_{k+1} - t_k|$ and the order of the interpolation method used.

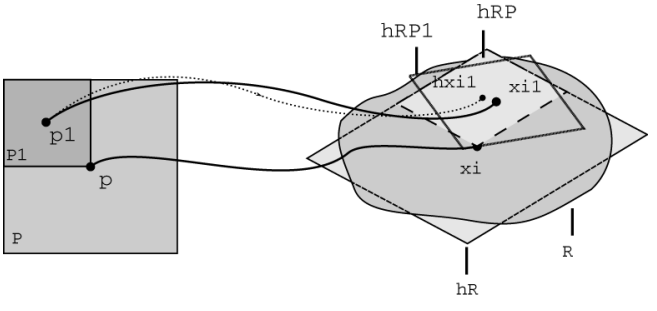


Fig. 2. "global" and "local" estimate of the reachable set $\mathcal{R}_t(\mathcal{S}_j)$

4.2 Algorithm

The key ideas of the algorithm presented below are the following:

- use the estimate $\hat{\mathcal{R}}^{\mathbf{P}}_t$ and its intersection with $\mathcal{B}$ to classify sets as safe, uncertain or unsafe;
- use Err to testify whether $\hat{\mathcal{R}}^{\mathbf{P}}_t$ is a reliable estimate: if it is more than a given tolerance $Tol > 0$ for a set $\mathcal{S}$, we classify $\mathcal{S}$ as uncertain;
- iteratively apply a refining operator on uncertain subsets to produce a finer partitioning from which we deduce more safe or unsafe subsets;
- stop when there are no uncertain subsets left or when all uncertain subsets are smaller in diameter than δp.

[1] Note that the value of Err actually depends not only on $\mathcal{S}$ and $\mathcal{S}_j$ but also on the choice of $\mathbf{p}$ and $\mathbf{p}_j$. We leave it implicit to simplify the notation.

To guarantee that the algorithm always terminates in a finite number of steps, we partition uncertain sets into a set of subsets that are at least γ times smaller. We define these γ-Refining partitions as follows.

Definition 1 (γ-Refining partition). *A γ-refining partition, where $0 < \gamma < 1$, of a set $\mathcal{S}$ is a finite set of sets $\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_l\}$ such that*

$$\mathcal{S} = \bigcup_{j=1}^{l} \mathcal{S}_j \quad \text{and} \quad \max_{j\in\{1,\ldots,l\}} \|\mathcal{S}_j\| \leq \gamma \|\mathcal{S}\|$$

We assume the existence of a function Refine_γ that maps a set to one of his γ-refining partitions for some $0 < \gamma < 1$ and give the complete algorithm below.

Algorithm 1. Parameter Synthesis Algorithm

```
procedure ParamSynthesis(P, B, T, δp, Tol)
    P_saf = P_bad = ∅, P_unc = {P}
    repeat
        Pick and remove S from P_unc
        for each S_j ∈ Refine_γ(S) do
            if Err(S, S_j) ≤ Tol then                    ▷ Reach set estimate is reliable
                if R̂^q_[0,T](S_j) ∩ B = ∅ then           ▷ Reach set away from B
                    P_saf = P_saf ∪ S_j
                else if R̂^q_[0,T](S_j) ⊂ B then          ▷ Reach set inside B
                    P_bad = P_bad ∪ S_j
                else
                    P_unc = P_unc ∪ {S_j}                ▷ Some intersection with the bad set
                end if
            else
                P_unc = P_unc ∪ {S_j}                    ▷ Reach set estimate not enough precise
            end if
        end for
    until P_unc ≠ ∅ and max_{P_j ∈ P_unc} ||P_j|| ≤ δp
    return P_saf, P_unc, P_bad
end procedure
```

5 Implementation and Experimentations

We implemented Algorithm 1 within the toolbox `Breach` described in [Don07]. Parameter sets are specified as symmetrical rectangular sets $\mathcal{S}(\mathbf{p}, \epsilon)$ where $\mathbf{p}$ and ϵ are in $\mathbb{R}^{n_p}$ and such that $\mathcal{S}(\mathbf{p}, \epsilon) = \{\mathbf{p}' : \mathbf{p} - \epsilon \leq \mathbf{p}' \leq \mathbf{p} + \epsilon\}$. The procedure uses a simple refinement operator $\text{Refine}_{\frac{1}{2}}$ such that

$$\text{Refine}_{\frac{1}{2}}(\mathcal{S}(\mathbf{p}, \epsilon)) = \{\mathcal{S}(\mathbf{p}^1, \epsilon^1), \mathcal{S}(\mathbf{p}^2, \epsilon^2), \ldots, \mathcal{S}(\mathbf{p}^l, \epsilon^l)\}$$

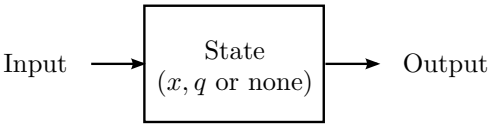
with $\epsilon^k = \epsilon/2$ and $\mathbf{p}^k = \mathbf{p} + (\nu_1^k \frac{\epsilon_1}{2}, \nu_2^k \frac{\epsilon_2}{2}, \ldots, \nu_n^k \frac{\epsilon_n}{2})$ where $\nu_i^k \in \{-1, +1\}$ so that when k ranges over $\{1, \ldots, l\}$ all possible sign combinations are met[2]. The

[2] The use of alternative refinement operators is a direction for further investigation.

toolbox interfaces MATLAB with CVODES [SH05], a numerical solver designed to solve efficiently and accurately ODEs and sensitivity equations of the form Eq. (4).

5.1 Sensitivity Analysis for Simulink Models

A Simulink block diagram model is a graphical representation of a mathematical model of an hybrid dynamic system [Mat]. It is composed of interconnected *time-based* blocks of the form shown below



where the state contained in the block (if present) can be either discrete or discontinuous. At each time step, the Simulink engine:

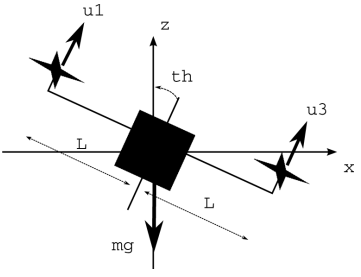
1. Computes each block output;
2. Updates the discrete states;
3. Computes the time derivatives $f(\mathbf{x})$ of continuous states;
4. Updates the continuous states by integrating $\dot{\mathbf{x}} = f(\mathbf{x})$ for one step;
5. Optionally checks for zero-crossing;
6. Updates the time for the next time step.

Thus the simulation scheme of Simulink is similar to the simulation algorithm presented in Section 2. To apply our algorithm to a Simulink model we need to extract from it the function f defining the continuous dynamics, the event function e and the guard function g, and to make them available for `Breach` to compute trajectories and sensitivity matrices. This is done using code generation provided by the Real-Time Workshop Toolbox [Mat]. The generated code implements routines for each of the above steps. For instance f can be obtained from step 3, e can be obtained from the discrete states update in step 2 and g is obtained from step 5. The overall procedure is shown in Fig. 3. A script generates C routines compatible with CVODES from the code generated by the Real-Time Workshop. Then `f()` calls the routines `MdlOutputs()` and `MdlDerivative()` for integration, `e()` calls `MdlUpdate()` to update discrete states and `g()` is used for zero-crossing detection (which is a CVODES built-in feature). Eq. (6) and (7) are used to update the sensitivity matrices at switching times.

5.2 Starmac Model with Navigation Supervisor

We consider a simplified model of a quad-rotor helicopter [HHWT07] where only the altitude z and the axis x are considered. The equations of motions for the quadrotor illustrated below are given by:

$$\ddot{x} = -\frac{b}{m}\dot{x} + \frac{1}{m}(u_1 + u_2 + u_3 + u_4)sin(\theta)$$
$$\ddot{z} = -\frac{b}{m}\dot{z} + \frac{1}{m}(u_1 + u_2 + u_3 + u_4)cos(\theta) - g$$
$$\ddot{\theta} = \frac{L}{I_y}(u_1 - u_3) - \frac{c}{L}\dot{\theta}$$



where $m = 0.5184$, $c = 0.15$, $L = 0.236$, $I_y = 0.04774$. The state vector is then $\mathbf{x} = (\dot{x}, x, \dot{z}, z, \dot{\theta}, \theta)$.

Given a goal state $\mathbf{x}^*$, a standard linear quadratic regulator (LQR) of the form $\mathbf{u} = K\ (\mathbf{x} - \mathbf{x}^*) + \frac{mg}{4}\ \mathbf{1}$ (where $\mathbf{1}$ is the vector $(1, 1, 1, 1)$) was designed to drive the system to $\mathbf{x}^*$ from any state $\mathbf{x}_0$. While doing so, the Starmac needs to avoid collisions with obstacles and maintain a pre-specified minimum safe flying altitude above an unknown terrain. This is monitored using two on-board proximity sensors, one in the horizontal (x) and the other in vertical (z) directions. Using the sensors and the value of the current state, a supervisor implements the following navigation strategy: in absence of proximity warnings, use the LQR control and move towards the target ('GoToTarget' mode); if either of the proximity warnings is active, switch to a constant control $\mathbf{u} = (\bar{u}, \bar{u}, \bar{u}, \bar{u})$, for some $\bar{u} > 0$, in order to go up until being safe ('GoUp' Mode) then resume to GoToTarget mode (see Figure 4).

While crashing of the Starmac into an obstacle is certainly undesirable, it may be desirable for it to be able to hover close to an obstacle. Hence the horizontal proximity warning was made velocity-dependent in the GoToTarget state, i.e., the more the velocity the farther away the system needs to be from the obstacle. The critical distance is set to be the product $\dot{x}\ t_{safe}$, for some $t_{safe} > 0$. In GoUp mode, the supervisor checks for a fixed horizontal distance h_{safe} from the obstacle. In both GoToTarget and GoUp modes, the vertical proximity from ground is a fixed desired vertical distance v_{safe}. This switching of control strategies leads

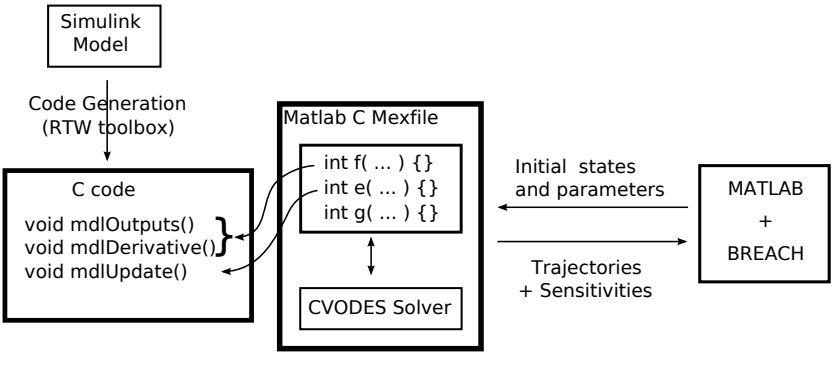


Fig. 3. Implementing sensitivity analysis for Simulink models in `Breach`

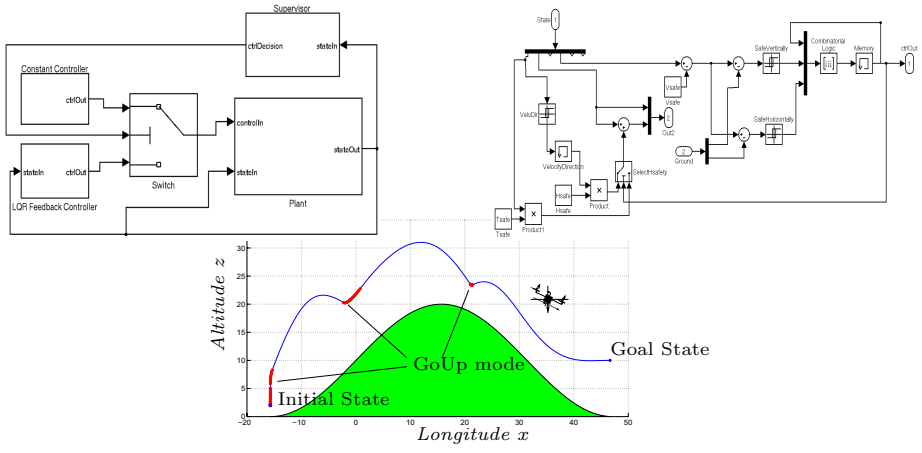


Fig. 4. Simulink diagram of the model (top left) and the supervisor (top right) and a sample (safe) trajectory

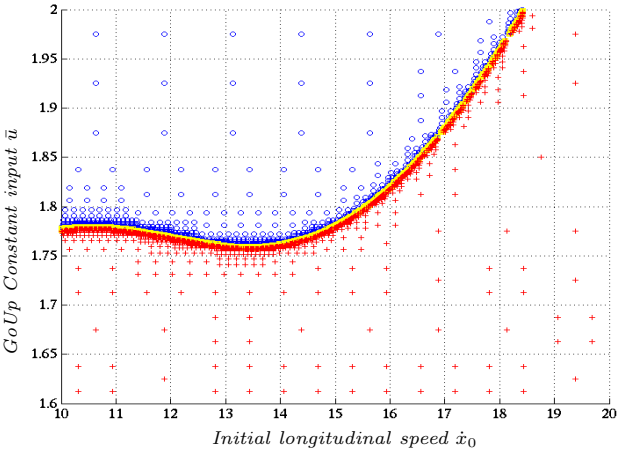
to hybrid dynamics with two discrete states, namely 'GoToTarget' and 'GoUp'. The proximity warning conditions in the two directions serve as the guards for discrete jumps between the two states.

5.3 Experimental Results

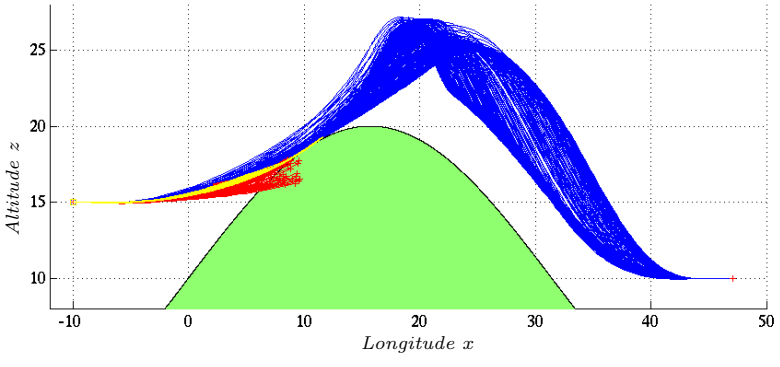
The Starmac dynamics, the LQR control and the supervisor were modeled in Simulink (Fig. 4). Proximity detection was modeled using relay blocks. The difference between desired distance from the obstacles and the current distance from obstacles can be fed as the input to these relays. These input signals to the relays are in turn extracted from the generated C code and fed to our sensitivity analysis machinery as the zero-crossing detection function $g()$.

We applied our parameter synthesis to the Starmac and the supervisor for different sets of parameters and a given terrain. The parameters that can vary in this model include the initial state variables (x_0, z_0, θ_0, $\dot{x}_0$, $\dot{z}_0$ and $\dot{\theta}_0$), the supervisor parameters (h_{safe}, v_{safe}, t_{safe}, $\bar{u}$), the Starmac characteristics (m, I_y, b), etc. We present the results we obtained for a situation where an initial position was set on one side of a hill (described by a simple sinusoid) and a goal state on the other side. The varying parameters where chosen to be the initial horizontal speed $\dot{x}_0$ and the constant control input $\bar{u}$ in GoUp mode, so that if we omit other parameters with a fixed value, $\mathcal{P} = \{(\dot{x}_0, \bar{u}) : 10 \leq \dot{x}_0 \leq 20,\ 1.6 \leq \bar{u} \leq 2\}$. The ground was set to be the bad set, given by $\mathcal{B} = \{z \leq \text{Terrain}(x)\}$ where Terrain is a sinusoidal function. The results are presented in Figure 5.

The algorithm performed 3642 simulations for a computational time of 55 seconds on a laptop with a Dual Core 1.8GHz processor. Most simulations stem from the neighborhood of a curve delimiting values of $(\dot{x}_0, \bar{u})$ for which trajectories cannot avoid the ground from values for which the avoidance maneuver



(a)



(b)

Fig. 5. Results varying the initial horizontal speed $\dot{x}_0$ and the GoUp constant input $\bar{u}$. (a) Parameters used for the simulation. Crosses represent values for which trajectories hit the ground while circles represent values for which the goal state is safely reached. (b) Resulting trajectories in the (x, z) plane.

works and the Starmac safely reaches the goal state. The algorithm refined the parameter set until a precision of $\delta\dot{x}_0 = 0.001$ and $\delta\bar{u} = 0.0004$. Note that performing simulations from parameters on a complete grid of this resolution would have required 262,144 simulations, more than 70 times the number of simulations executed by the sensitivity-based algorithm.

6 Conclusion

This paper presents a parameter synthesis algorithm for nonlinear hybrid systems based on numerical simulation and sensitivity analysis. The algorithm is

scalable in terms of number of state variables and is implemented in a MATLAB toolbox, `Breach`, that can handle Simulink models directly. The proposed approach is illustrated for a six-dimensional nonlinear Simulink model of the STARMAC quadrotor helicopter with a non-trivial hybrid supervisor.

The primary limitation of the algorithm is that the complexity of the refinement procedure is exponential in the number parameters. We are currently investigating methods for scaling the approach to large numbers of parameters. We are also extending the prototype implementation, which currently handles zero crossing detection for relay blocks, to a larger set of Simulink and Stateflow blocks. Directions for future research include the ability to handle models with uncertain inputs, i.e., dynamics of the form $\dot{\mathbf{x}} = f(q, \mathbf{x}(t), u(t), \mathbf{p})$, and extensions to stochastic systems.

References

[ADF+06] Asarin, E., Dang, T., Frehse, G., Girard, A., Le Guernic, C., Maler, O.: Recent progress in continuous and hybrid reachability analysis. In: Proc. IEEE International Symposium on Computer-Aided Control Systems Design. IEEE Computer Society Press, Los Alamitos (2006)

[DM07] Donzé, A., Maler, O.: Systematic simulation using sensitivity analysis. In: Bemporad, A., Bicchi, A., Buttazzo, G. (eds.) HSCC 2007. LNCS, vol. 4416, pp. 174–189. Springer, Heidelberg (2007)

[Don07] Donzé, A.: Trajectory-Based Verification and Controller Synthesis for Continuous and Hybrid Systems. PhD thesis, University Joseph Fourier (June 2007)

[EKP01] Esposito, J.M., Kumar, V., Pappas, G.J.: Accurate event detection for simulating hybrid systems. In: Di Benedetto, M.D., Sangiovanni-Vincentelli, A.L. (eds.) HSCC 2001. LNCS, vol. 2034, pp. 204–217. Springer, Heidelberg (2001)

[FJK08] Frehse, G., Jha, S.K., Krogh, B.H.: A counterexample-guided approach to parameter synthesis for linear hybrid automata. In: Egerstedt, M., Mishra, B. (eds.) HSCC 2008. LNCS, vol. 4981, pp. 187–200. Springer, Heidelberg (2008)

[Fre05] Frehse, G.: Phaver: Algorithmic verification of hybrid systems past hytech, pp. 258–273. Springer, Heidelberg (2005)

[GP06] Girard, A., Pappas, G.J.: Verification using simulation. In: Hespanha, J.P., Tiwari, A. (eds.) HSCC 2006. LNCS, vol. 3927, pp. 272–286. Springer, Heidelberg (2006)

[HHWT97] Henzinger, T.A., Ho, P.-H., Wong-Toi, H.: Hytech: A model checker for hybrid systems. Software Tools for Technology Transfer 1, 460–463 (1997)

[HHWT07] Hoffmann, G., Huang, H., Waslander, S., Tomlin, C.J.: Quadrotor helicopter flight dynamics and control: Theory and experiment. In: Proceedings of the AIAA Conference on Guidance, Navigation and Control, Hilton Head, South Carolina (August 2007)

[HP00] Hiskens, I.A., Pai, M.A.: Trajectory sensitivity analysis of hybrid systems. IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications 47(2), 204–220 (2000)

[HWT96] Henzinger, T.A., Wong-Toi, H.: Using hytech to synthesize control parameters for a steam boiler. In: Abrial, J.-R., Börger, E., Langmaack, H. (eds.) Dagstuhl Seminar 1995. LNCS, vol. 1165, pp. 265–282. Springer, Heidelberg (1996)

[LKCK08] Lerda, F., Kapinski, J., Clarke, E.M., Krogh, B.H.: Verification of supervisory control software using state proximity and merging. In: Egerstedt, M., Mishra, B. (eds.) HSCC 2008. LNCS, vol. 4981, pp. 344–357. Springer, Heidelberg (2008)

[Mat] The Mathworks. Simulink User Guide

[SH05] Serban, R., Hindmarsh, A.C.: Cvodes: the sensitivity-enabled ode solver in sundials. In: Proceedings of IDETC/CIE 2005, Long Beach, CA (September 2005)

Convergence of Distributed WSN Algorithms: The Wake-Up Scattering Problem

Daniele Fontanelli, Luigi Palopoli, and Roberto Passerone*

Dipartimento di Ingegneria e Scienza dell'Informazione
University of Trento - Trento, Italy

Abstract. In this paper, we analyze the problem of finding a periodic schedule for the wake-up times of a set of nodes in a Wireless Sensor Network that optimizes the coverage of the region the nodes are deployed on. An exact solution of the problem entails the solution of an Integer Linear Program and is hardly viable on low power nodes. Giusti et. al. [6] have recently proposed an efficient decentralized approach that produces a generally good suboptimal solution. In this paper, we study the convergence of this algorithm by casting the problem into one of asymptotic stability for a particular class of linear switching systems. For general topologies of the WSN, we offer local stability results. In some specific special cases, we are also able to prove global stability properties.

1 Introduction

In the past few years, Wireless Sensor Networks (WSN) have emerged as one of the most interesting innovations introduced by the ICT industry. Their potential fields of application cover a wide spectrum, including security, disaster management, agricultural monitoring and building automation.

The most relevant feature of a WSN is that it is a dynamic distributed system, in which complex tasks are performed through the coordinated action of a large number of small devices (nodes). The integrity of the network, however, can be affected if nodes become suddenly unoperational, especially when the system is deployed in a remote environment. Therefore, a prominent issue is the ability of the WSN to robustly fulfill its goals, countering possible changes in the environment and/or in the network. The same level of importance is commonly attached to the system lifetime. Since replacing batteries may be too expensive and since even modern scavenging mechanisms cannot drain large quantities of energy from the environment, a WSN is required to minimize energy dissipation. Therefore, the main stay of the research on WSN are distributed algorithms for data processing and resource management that attain an optimal trade-off between functionality, robustness and lifetime.

A popular way for pursuing this result is the application of "duty-cycling". The idea is to keep a node inactive for a long period of time when its operation is

* This work was supported by the EC under contract IST-2008-224428 "CHAT - Control of Heterogeneous Automation Systems".

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 180–193, 2009.

not needed, and then to awake it for a short interval to perform its duties (e.g., to sense the surrounding environment). The performance of the WSN heavily depends on the application and on how this duty-cycling is scheduled. The determination of an optimal duty-cycle schedule has been the subject of intense research. In this paper we focus on the problem of determining a schedule that maximizes the average area "sensed" by the network, given a desired value for the lifetime. While an optimal solution can be found using a centralized formulation [17,4,9,1], we are interested in this paper in analyzing strategies where the schedule is computed online by the nodes themselves.

A very simple and effective heuristic to obtain a suboptimal solution is the wake-up scattering algorithm presented in [6], which we describe in Section 2. The idea is to "scatter" the execution of neighboring nodes, under the assumption that two nodes communicating with each other also share large portions of the covered area, and should therefore operate at distinct times. Experimental evidence suggests that the solution thus found is frequently very close to the optimal one (its distance ranging from 15% to 5%). The schedule is computed from a random solution by iteratively adjusting the wake-up times using information from the neighbors. In this paper, we offer a theoretical study of the algorithm by formally proving its convergence.

Our first contribution is to model the evolution of the system by a state-space description, in which state variables represent the distance between the wake-up times of an appropriate subset of the nodes. The model is generally a switching linear system, in which the dynamic matrix can change depending on the ordering of the distances between the node wake-up times. The problem of convergence of the wake-up scattering algorithm can be cast into a stability problem for this system. For each and every of the linear dynamics composing the switching system, we prove the existence of a subspace composed of equilibrium points for the system. We also show that this equilibrium set is asymptotically stable under the hypothesis that it does not coincide with a switching surface.

This local stability can be strengthened if additional hypotheses are made on the topology of the network. In the particular case in which each node can communicate with the ones whose wake-up times are the closest to its own (the nearest neighbors), we show a particular coordinate transformation, whereby the dynamics of the autonomous linear system are governed by a doubly stochastic dynamic matrix. The stability of this type of systems is well studied [14], and it recently found an interesting application in the consensus problem (see [15,12,13,5] and references therein). In particular, under the restrictive assumptions stated above (visibility of the nearest neighbors), the wake-up scattering problem can be viewed as a deployment task, solved with respect to time, over a cyclic set of possible configurations [10,8,11]. However, as shown below, there are reasonable situations under which switches in the linear dynamics can happen and the classical analysis on consensus problems cannot be applied to the convergence of the wake-up scattering. Intuitively, the reason of this divergence is the fact that while agents moving on a line are "physically" prevented from overtaking each other, this limitations does not apply to the wake-up times of the nodes. Indeed,

as shown below, nodes can change their relative time positions if they do not see each other. As a final result for the paper, we prove *global* convergence for some particular topologies for which the visibility of the nearest neighbor does not hold. In our view, this is the first step toward more general global stability results for the wake-up scattering problem.

The paper is organized as follows. In Section 2, we provide some background information about the wake-up scattering algorithm. In Section 3, we construct a state-space model for the evolution of the system. In Section 4, we describe the stability results, which constitute the trunk of the paper. In Section 5, we show some simple numerical examples that clarify the results of the paper and the potential of the algorithm. Finally, in Section 6, we state our conclusions and outline future developments.

2 Background

In this paper we analyze the problem of reducing the power consumption (and therefore extend the lifetime) of a sensor network while providing continuous node coverage over a monitored area. To save power, we switch nodes off for a period of time if another node covering the same area is guaranteed to be active. This technique results in a (typically periodic) schedule of the wake-up intervals of the nodes.

An optimal schedule may be computed either centrally, before deployment, or online by the network itself, in a distributed fashion. Online techniques are preferable in those cases in which the network topology may change, or is not known a priori, and access to a central server is expensive or not available. These techniques typically use information from neighboring nodes to iteratively refine the local schedule [6,19,18,7,3,2]. Of particular interest, in this case, is determining whether the distributed algorithm converges to a solution, how far the solution is from optimal, and how long the transient of the computation lasts.

Here, we consider the scheme proposed by Giusti et al. [6], and focus on the problem of convergence. The considered algorithm computes a periodic schedule over an epoch E, where each node wakes up for only a defined interval of time W to save power. The procedure optimizes the coverage by *scattering* the wake-up times of neighboring nodes (nodes that can communicate directly over the radio channel), i.e., nodes are scheduled so that they wake up as far in time as possible from neighboring nodes. The rationale behind this approach is the assumption that neighboring nodes are more likely to cover the same area. This is true when the sensing range and the radio range are comparable in length. Scattering, in this case, results in a schedule where the wake-up intervals of nodes covering the same area do not overlap, thus achieving a better coverage. While this assumption is clearly an approximation, the technique is extremely simple and relies solely on connectivity, instead of requiring exact position information.

More in detail, the wake-up scattering algorithm proposed in [6] starts from a random schedule and then proceeds in rounds. At every round, nodes broadcast their current wake-up time to all their neighbors. With this information, a node

may construct a local copy of the current schedule, limited to information related to its neighboring nodes. By inspecting this schedule, nodes update their wake-up time to fall exactly in the middle between the closest neighboring nodes that wake up immediately before and immediately after their current position in the schedule. This way, a node tries to maximize its distance in time from the closest (in time) neighboring node.

To formalize this procedure, consider N nodes $n_1, \ldots, n_N$ and let E be the duration of the epoch. We denote by $w_i \in [0, E]$ the wake-up time of node n_i. Let also $\mathcal{V}_i$ be the set of nodes visible from node n_i ($i \notin \mathcal{V}_i$). The wake-up time of node n_i at step k is then updated as follows:

$$w_i^{k+1} = (1-\alpha)\, w_i^k + \frac{\alpha}{2}\left(\min_{j\in\mathcal{V}_i}\{w_j^k : w_j^k \geq w_i^k\} + \max_{j\in\mathcal{V}_i}\{w_j^k : w_j^k \leq w_i^k\}\right) \mod E, \tag{1}$$

where $\alpha \in [0, 1]$ controls the speed at which the position of the node in the scheduled is updated during an iteration. The formula is ill-defined if the set $\{w_j^k : w_j^k \geq w_i^k\}$ is empty (because n_i is the last node to wake up in the schedule among its neighbors). In that case, according to the proposed algorithm, the empty set is replaced with the set $\{w_j^k : w_j^k + E \geq w_i^k\}$, i.e., we wrap around the schedule to consider the next nodes to wake up, which will be in the following epoch. A similar wrap around is required when $\{w_j^k : w_j^k \leq w_i^k\}$ is empty. Taking this and the remainder operation into account makes the analysis of the model difficult. In the next section we describe how to simplify the formulation by switching our attention from the wake-up time to the distance in the schedule between the nodes.

3 System Model

To study the convergence of the algorithm in Equation (1), it is convenient to reason about the distance between the nodes (their relative position), rather than about their absolute position in time (with the additional advantage of abstracting away the exact position, which is irrelevant). The distance between two nodes is always positive and between 0 and E. For each pair of nodes (n_i, n_j) we define two distances: one, denoted $\overrightarrow{d}_{i,j}$ that goes forward in time, the other, denoted $\overleftarrow{d}_{i,j}$ that goes backward. Since distances are always positive, we have

$$\overrightarrow{d}_{i,j} = \begin{cases} w_j - w_i & \text{if } w_i \leq w_j, \\ w_j - w_i + E & \text{otherwise.} \end{cases}$$

$$\overleftarrow{d}_{i,j} = \begin{cases} w_i - w_j & \text{if } w_j \leq w_i, \\ w_i - w_j + E & \text{otherwise.} \end{cases}$$

From the definition above it follows that

$$\overleftarrow{d}_{i,j} = E - \overrightarrow{d}_{i,j}, \tag{2}$$

and hence

$$\max_{j\in\mathcal{V}_i}(\overrightarrow{d}_{i,j}) = \max_{j\in\mathcal{V}_i}(E - \overleftarrow{d}_{i,j}) = E - \min_{j\in\mathcal{V}_i}(\overleftarrow{d}_{i,j}).$$

We are interested in computing the new distance between every pair of nodes after an update. To do so, we first compute the amount Δ by which each node moves after the update. This is given by

$$\Delta_i^k = w_i^k - w_i^{k-1} = \frac{\alpha}{2}\left(\min_{l\in\mathcal{V}_i}(\overrightarrow{d}_{i,l}^{\,k}) - \min_{l\in\mathcal{V}_i}(\overleftarrow{d}_{i,l}^{\,k})\right).$$

The distance between two nodes at iteration $k+1$ can be computed as the distance at iteration k corrected by the displacement. Hence,

$$\overrightarrow{d}_{i,j}^{\,k+1} = \overrightarrow{d}_{i,j}^{\,k} - \Delta_i^k + \Delta_j^k, \tag{3}$$

$$\overleftarrow{d}_{i,j}^{\,k+1} = \overleftarrow{d}_{i,j}^{\,k} + \Delta_i^k - \Delta_j^k. \tag{4}$$

The distance between two nodes remains bounded by 0 and E during the iterations, i.e., the distances always belong to the set $\mathcal{S}_E = \{x \in \mathbb{R} | 0 \le x \le E\}$.

Theorem 1. *Let n_i and n_j be nodes that see each other (i.e., $n_j \in \mathcal{V}_i$ and $n_i \in \mathcal{V}_j$). Let $\overrightarrow{d}_{i,j}^{\,0}, \overleftarrow{d}_{i,j}^{\,0} \in \mathcal{S}_E$. Then, for all $k > 0$, $\overrightarrow{d}_{i,j}^{\,k}, \overleftarrow{d}_{i,j}^{\,k} \in \mathcal{S}_E$.*

Proof. By adding (3) and (4) we have $\overrightarrow{d}_{i,j}^{\,k+1} + \overleftarrow{d}_{i,j}^{\,k+1} = \overrightarrow{d}_{i,j}^{\,k} + \overleftarrow{d}_{i,j}^{\,k}$. From (2) we have $\overrightarrow{d}_{i,j}^{\,0} + \overleftarrow{d}_{i,j}^{\,0} = E$, therefore, by induction, $\overrightarrow{d}_{i,j}^{\,k} + \overleftarrow{d}_{i,j}^{\,k} = E$. We will now bound the displacement of the nodes at each iteration.

$$\Delta_i^k = \frac{\alpha}{2}\left(\min_{l\in\mathcal{V}_i}(\overrightarrow{d}_{i,l}^{\,k}) - \min_{l\in\mathcal{V}_i}(\overleftarrow{d}_{i,l}^{\,k})\right) \le \frac{\alpha}{2}\overrightarrow{d}_{i,j}^{\,k}$$

Also, since $\overrightarrow{d}_{i,j} = \overleftarrow{d}_{j,i}$ (proof left to the reader),

$$\Delta_j^k = \frac{\alpha}{2}\left(\min_{l\in\mathcal{V}_j}(\overrightarrow{d}_{j,l}^{\,k}) - \min_{l\in\mathcal{V}_j}(\overleftarrow{d}_{j,l}^{\,k})\right) = \frac{\alpha}{2}\left(\min_{l\in\mathcal{V}_j}(\overleftarrow{d}_{l,j}^{\,k}) - \min_{l\in\mathcal{V}_j}(\overrightarrow{d}_{l,j}^{\,k})\right) \ge -\frac{\alpha}{2}\overrightarrow{d}_{i,j}^{\,k}$$

Therefore $\Delta_i^k - \Delta_j^k \le \frac{\alpha}{2}\overrightarrow{d}_{i,j}^{\,k} + \frac{\alpha}{2}\overrightarrow{d}_{i,j}^{\,k} = \alpha\overrightarrow{d}_{i,j}^{\,k} \le \overrightarrow{d}_{i,j}^{\,k}$. Hence, from (3),

$$\overrightarrow{d}_{i,j}^{\,k+1} \ge 0. \tag{5}$$

Similarly, one shows that $\overleftarrow{d}_{i,j}^{\,k+1} \ge 0$. Therefore, since their sum is E and they are positive, we obtain the result.

The theorem above shows that nodes that see each other do not overtake each other after an update, since their distance remains positive and bounded by E.

To prove the stability of the update rule, i.e., that the wake-up intervals converges to a periodic schedule preserving the WSN power while ensuring the area coverage, a state space description is needed. At this point, a straightforward choice for the state vector is to contain all the possible $N(N-1)$ wake–up distances among all the N nodes of the WSN. However, since in the update

equations (3) and (4) only the distances between nodes that see each other are involved, a more interesting choice is to select the state variables among such node distances.

Therefore, without loss of generality, consider $\mathcal{V}_i \neq \emptyset$, $\forall i = 1, \ldots, N$, i.e. each node sees at least another node[1]. Consider a state vector $\mathbf{x}$ whose entries are the distances $\overrightarrow{d}_{i,l}$, $\forall l \in \mathcal{V}_i$ and for $i = 1, \ldots, N$. Similarly, let $\mathbf{y}$ be the vector of distances $\overleftarrow{d}_{i,l}$, $\forall l \in \mathcal{V}_i$ and for $i = 1, \ldots, N$. Trivially, the number of elements N_x in the state vector $\mathbf{x}$ depends on the visibility graph.

Introducing the notation $\bar{\Delta}^k_{\overrightarrow{d}} = \frac{\alpha}{2}\left(\min_{l\in\mathcal{V}_j}(\overrightarrow{d}^k_{j,l}) - \min_{l\in\mathcal{V}_i}(\overrightarrow{d}^k_{i,l})\right)$, $\bar{\Delta}^k_{\overleftarrow{d}} = \frac{\alpha}{2}\left(\min_{l\in\mathcal{V}_j}(\overleftarrow{d}^k_{j,l}) - \min_{l\in\mathcal{V}_i}(\overleftarrow{d}^k_{i,l})\right)$, we can rewrite (3) and (4) as:

$$\overrightarrow{d}^{k+1}_{i,j} = \overrightarrow{d}^k_{i,j} + \bar{\Delta}^k_{\overrightarrow{d}} - \bar{\Delta}^k_{\overleftarrow{d}}, \tag{6}$$

$$\overleftarrow{d}^{k+1}_{i,j} = \overleftarrow{d}^k_{i,j} - \bar{\Delta}^k_{\overrightarrow{d}} + \bar{\Delta}^k_{\overleftarrow{d}}. \tag{7}$$

With the proposed choice of the state variables, the update displacement $\bar{\Delta}^k_{\overrightarrow{d}}$ only depends on the distances in $\mathbf{x}$, and $\bar{\Delta}^k_{\overleftarrow{d}}$ only on the distances in $\mathbf{y}$. Rewriting (6) and (7) in matrix notation, yields

$$\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix}^{k+1} = \begin{bmatrix}\mathbf{x}^k + \Gamma'_x\mathbf{x}^k + \Gamma''_x\mathbf{y}^k\\ \mathbf{y}^k - \Gamma'_x\mathbf{x}^k - \Gamma''_x\mathbf{y}^k\end{bmatrix}, \tag{8}$$

where matrices Γ'_x, Γ''_x collects the $\pm\alpha/2$ factors.

Observing that the invariance property (2) can be written as $\mathbf{y} = E\mathbf{1} - \mathbf{x}$, where $\mathbf{1} \in \mathbb{R}^{N_x}$ is the column vector with all entries equal to 1, the discrete time evolution of system (8) is simplified as:

$$\mathbf{x}^{k+1} = (I_{N_x} + \Gamma'_x - \Gamma''_x)\mathbf{x}^k + E\Gamma''_x\mathbf{1} = A\mathbf{x}^k + bE. \tag{9}$$

Therefore the stability of the system is related to the eigenvalues of A and to the time response to the constant input E.

Since Γ'_x in (8) is related only to $\bar{\Delta}^k_{\overrightarrow{d}}$, it contains two entries in each row that are equal to $\alpha/2$ and $-\alpha/2$ respectively. Similar considerations apply to Γ''_x, as summarized in the following theorem.

Theorem 2. *The rows of the system matrices Γ'_x and Γ''_x have exactly two entries that are not equal to zero. Furthermore, the sum of the elements of each rows is zero.*

A consequence of Theorem 2 is that the discrete time system (9) is autonomous

$$\mathbf{x}^{k+1} = (I_{N_x} + \Gamma'_x - \Gamma''_x)\mathbf{x}^k = A\mathbf{x}^k, \tag{10}$$

and it has, at least, one eigenvalue equals to one. Hence, $\mathbf{x}$ contains the number of elements that are sufficient for the whole system dynamic description. Nevertheless, it may contain redundant variables, as the following example highlights.

[1] Blind nodes have no dynamics and do not participate to the dynamic of the other nodes.

Example 1. Let us consider a network with $N = 3$, where n_1 sees only n_2 and n_3 sees only n_2. Without loss of generality, assume $w_1 < w_2 < w_3$. The proposed choice of state variables yields $\mathbf{x} = [\overrightarrow{d}_{1,2}, \overrightarrow{d}_{2,3}, \overrightarrow{d}_{3,2}, \overrightarrow{d}_{2,1}]^T$. Notice that $\overrightarrow{d}_{2,1} = E - \overrightarrow{d}_{1,2}$. Hence it can be erased without loss of information. Nonetheless, erasing $\overrightarrow{d}_{2,1}$ (and, hence, substituting the term $\overrightarrow{d}_{2,1} = \overleftarrow{d}_{1,2}$ in $\bar{\Delta}^k_{\overleftarrow{d}}$ of equation (6)) yields to an input vector $\hat{b} = [\alpha/2, 0, 0]^T$ that makes the stability analysis more complex in the general case.

As shown in Section 5, if two nodes do not see each other, they can overtake each other. This behavior, together with the fact that the updating equations (6) and (7) are nonlinear, makes the matrices Γ'_x and Γ''_x time variant. Therefore, the overall system dynamics is switching. Defining $\sigma(k)$ as the switching signal, that takes values $1, \ldots, S$, the switching system $\mathbf{x}^{k+1} = A_{\sigma(k)}\mathbf{x}^k$ is thus derived, with system matrices $\{A_1, A_2, \ldots, A_S\}$. The region of the state space in which the system evolves using a dynamic A_i is a convex polyhedron delimited by a set of subspaces of the type $x_i < x_j$, for appropriate choices of i and j.

On the other hand, in view of Theorem 1, a node cannot overtake any other node that it sees. Therefore, if all nodes see their nearest neighbors, the application of Equation (6) and (7) always produces the same dynamic A_1 and the system evolves with a linear and time–invariant dynamics. In the general case, the number S of linear dynamics is upper bounded by the number of pairs of nodes that do not see each other.

In the rest of the section, we will first study the stability properties of each linear dynamic system A_i. Then we will extend our analysis to the global stability properties for some specific topologies. For the sake of brevity, we will not focus on the case $N > 2$, since for $N = 2$ the study of the behavior of the system is straightforward.

4 Stability Analysis

4.1 Local Analysis

As discussed above, the evolution of the system is generally described by a linear switching system. Our first task is to study the evolution of the system in each of its linear dynamics.

Consider the set $\mathcal{S}_E^{N_x} = \{\mathbf{x} \in \mathbb{R}^{N_x} | 0 \leq x_i \leq E\}$. The stability of each of the linear dynamics of the system is showed in the subsequent Lemma.

Lemma 1. *Given the system* $\mathbf{x}^{k+1} = A\mathbf{x}^k$ *and* $\mathbf{x}^0 \in \mathcal{S}_E^{N_x}$ *the following statements hold true:*

- $\mathbf{x}^k \in \mathcal{S}_E^{N_x}$ $\forall k > 0$*;*
- *the system is stable;*
- *the equilibrium points* $\bar{\mathbf{x}}$ *belong to a linear subspace defined by the* $m \geq 1$ *eigenvectors* $\mathbf{v}_i$ *associated to the* m *eigenvalues* $\lambda_i = 1$*.*

Proof. By construction, each element of the state vector $\mathbf{x}$ is a distance between two nodes that see each other, therefore, by Theorem 1, the nodes do not overtake each other during an update – i.e. the set $\mathcal{S}_E^{N_x}$ is invariant for A. Since the distances $\mathbf{x}^0$ are positive by hypothesis, the first statement holds.

The diagonal elements of the matrix A are equal to $1 - l\alpha/2$, where l is an integer number in the set $\{0, \ldots, 4\}$. Indeed, each element $\overrightarrow{d}_{i,j} \in \mathbf{x}$ may appear up to four times in the update equations (6) and (7). Applying the Gerschgorin principle to each row and recalling that $0 < \alpha < 1$, it follows that the eigenvalues associated to $l = 4$ and $l = 3$ are inside the unit circle, while for $l = 2$ we have at most an eigenvalue $\lambda = 1$. The case of $l = 1$ is more challenging, since the Gerschgorin principle cannot be applied to demonstrate convergence, although it states that $\mathrm{Re}(\lambda_i) > -1$. However, since each element of the state vector $\mathbf{x}$ is a distance between two nodes that see each other, Theorem 1 holds, so it is not possible to have expansive dynamics, i.e. $\max_i |\lambda_i| = 1$. Finally, since cancellations are not possible, the presence of 1 on the diagonal of A ($l = 0$) means that the associated distance does not contribute to its dynamic and to any other dynamic, i.e. if the distance is $\overrightarrow{d}_{i,j}$, there is a node closer to i than j and vice-versa. Therefore, applying the Gerschgorin principle to the column results, again, in an eigenvalue $\lambda_i = 1$. Notice that if m eigenvalues $\lambda_i = 1$, they must be simple, i.e. associated to distinct eigenvectors $\mathbf{v}_i$, since, again, no expansive dynamics are allowed.

To show the stability of the system to a point $\bar{\mathbf{x}} = \sum_{i=1}^{m} \beta_i \mathbf{v}_i$, it is sufficient to prove that there is not a persistent oscillation in the system. Trivially, oscillating modes exists if there will be one or more simple eigenvalues $\lambda_i = -1$, excluded by the aforementioned Gerschgorin analysis, or in the presence of complex eigenvalues. Since complex eigenvalues do not exist (indeed $\mathbf{x}^k \geq 0 \; \forall k > 0$), the Lemma is proved.

As an immediate consequence of the above, if we perturb the system state from an equilibrium point that is not on a switching surface, i.e., a surface delimited by constraints of the type $\overrightarrow{d}_{i,j} = \overrightarrow{d}_{i,l}$, $l, j \in \mathcal{V}_i$ and $\forall i = 1, \ldots, N$, the system will recover its equilibrium. In plain words, the wake-up scattering algorithm converges if we initialize the vector of wake-up times with an initial value close to a fixed point and far enough from a switching surface. Unfortunately, since the property is local, we are not able to easily quantify the maximum amount of the allowed perturbation.

4.2 Global Analyis

The result described above does not *per se* ensure convergence starting from a general initial condition. However, there are some interesting topologies for which such global "provisions" can indeed be given.

Visibility of the nearest neighbors. In case of visibility of the nearest neighbors, the topology of the system structurally prevents any switch. Therefore the local stability results that we stated above for each linear dynamic have, in this

case, global validity. In other words, the wake-up scattering converges from any positive initial assignment for the wake-up times.

The complete visibility topology, i.e., $\mathcal{V}_i = j, \forall i, j = 1, \ldots, N, \forall j \neq i$, and the cyclic topology have been explicitly considered in the literature on the consensus problem ([8,11]). Indeed, choosing only the entries of $\mathbf{x}$ equal to $\min_{l \in \mathcal{V}_i}(\overrightarrow{d}_{i,l})$ for $i = 1, \ldots, N$, hence $\mathbf{x} \in \mathbb{R}^N$, the discrete time system (10) is again autonomous and its dynamic matrix A turns out to be doubly stochastic and circulant ([12]). Therefore, it is possible to determine the closed form of its eigenvalues and the equilibrium point ([11]), the rate of convergence with respect to the number of nodes ([15]) and the network communication constraints to accomplish the desired task ([5]). We will not consider this case any further (a numeric example is presented below).

Removing one link to the nearest neighbor. For this case, we start from a complete graph and remove one link to the nearest neighbor. In this case, we end up with two nodes, say j and p, that do not see each other and are the nearest ones to each other. This situation is depicted in Figure 1(A). Consider the two update laws with respect to node i (nearest neighbor to j):

$$\overrightarrow{d}^{k+1}_{i,p} = \overrightarrow{d}^{k}_{i,p} + \frac{\alpha}{2}(\overrightarrow{d}^{k}_{p,f} - \overrightarrow{d}^{k}_{i,j}) - \frac{\alpha}{2}(\overrightarrow{d}^{k}_{i,p} - \overrightarrow{d}^{k}_{z,i})$$
$$\overrightarrow{d}^{k+1}_{i,j} = \overrightarrow{d}^{k}_{i,j} + \frac{\alpha}{2}(\overrightarrow{d}^{k}_{j,f} - \overrightarrow{d}^{k}_{i,j}) - \frac{\alpha}{2}(\overrightarrow{d}^{k}_{i,j} - \overrightarrow{d}^{k}_{z,i})$$

The update law of the distance $\overrightarrow{d}^{k}_{j,p} = \overrightarrow{d}^{k}_{i,p} - \overrightarrow{d}^{k}_{i,j}$ is

$$\overrightarrow{d}^{k+1}_{j,p} = \overrightarrow{d}^{k+1}_{i,p} - \overrightarrow{d}^{k+1}_{i,j} = \left(1 - \frac{\alpha}{2}\right)\overrightarrow{d}^{k}_{j,p} + \frac{\alpha}{2}(\overrightarrow{d}^{k}_{p,f} - \overrightarrow{d}^{k}_{j,f}) = (1-\alpha)\,\overrightarrow{d}^{k}_{j,p}.$$

Thereby, even though the nodes do not see each other, they do not overtake each other and will, eventually, occupy the same time position. Hence, the switching never happens and the system will converge as in the complete visibility topology with $N-1$ nodes.

Removing four links to the nearest neighbor. Consider a more involved condition, in which two pairs of nodes do not see their nearest neighbor, as in Figure 1(B). Consider the distances of $\overrightarrow{d}^{k}_{j,p} = \overrightarrow{d}^{k}_{i,p} - \overrightarrow{d}^{k}_{i,j}$ and $\overrightarrow{d}^{k}_{l,f} = \overrightarrow{d}^{k}_{l,w} - \overrightarrow{d}^{k}_{f,w}$. Noticing that $\overrightarrow{d}^{k}_{p,f} - \overrightarrow{d}^{k}_{j,l} = \overrightarrow{d}^{k}_{l,f} - \overrightarrow{d}^{k}_{j,p}$, one gets

$$\overrightarrow{d}^{k+1}_{j,p} = (1-\alpha)\,\overrightarrow{d}^{k}_{j,p} + \frac{\alpha}{2}\overrightarrow{d}^{k}_{l,f}$$
$$\overrightarrow{d}^{k+1}_{l,f} = (1-\alpha)\,\overrightarrow{d}^{k}_{l,f} + \frac{\alpha}{2}\overrightarrow{d}^{k}_{j,p},$$

that is a linear system with a pair of real eigenvalues: $\lambda_1 = 1 - \alpha/2$ and $\lambda_2 = 1 - 3\alpha/2$. We can distinguish two case: 1) $\lambda_1 > 0$, $\lambda_2 > 0$, 2) $\lambda_1 > 0$ or $\lambda_2 < 0$. In the first case (corresponding to $0 < \alpha \leq 2/3$), we can easily see that for each initial condition there is a maximum time beyond which switchings no longer occur. Then, recalling Lemma 1, global stability is ensured. In the second case,

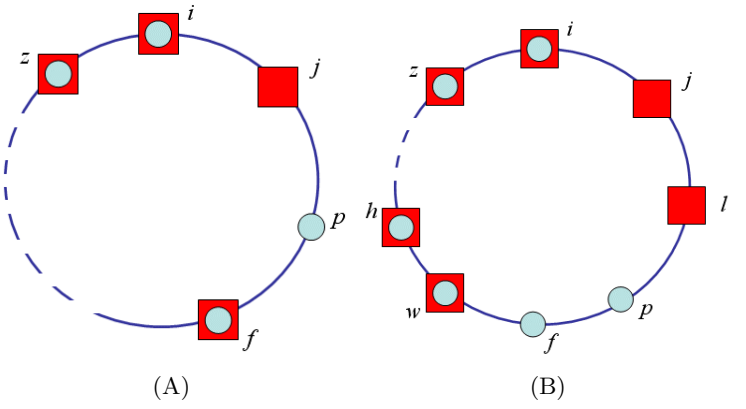


Fig. 1. Visibility and proximity of the partial visibility topology. Nearest neighbors are the closest nodes in clockwise and counter–clockwise direction, i.e. the nearest neighbor to l are p and j. Visibility is depicted with squares and circles respectively. (A) $p \notin \mathcal{V}_j$ and $j \notin \mathcal{V}_p$. (B) $p, f \notin \mathcal{V}_j$, $p, f \notin \mathcal{V}_l$, $j, l \notin \mathcal{V}_p$, $j, l \notin \mathcal{V}_f$.

we cannot rule out an infinite number of switchings determined by the oscillating behavior of the power of the negative real eigenvalue. Even so, we get $\overrightarrow{d}^{\,k}_{j,p} \to 0$ and $\overrightarrow{d}^{\,k}_{l,f} \to 0$. As a result the two pair of nodes will eventually behave as a single node with complete visibility and the network will stabilize on the equilibrium point a network with complete visibility would have with $N - 2$ nodes.

The same approach can also be used if, for example, node f is removed from the network, which implies that the symmetry among the nodes with partial neighbor visibility is no longer valid. From the previous analysis it follows that

$$\overrightarrow{d}^{\,k+1}_{j,p} = (1 - \alpha)\, \overrightarrow{d}^{\,k}_{j,p} + \frac{\alpha}{2} \overrightarrow{d}^{\,k}_{l,w}.$$

Since $\overrightarrow{d}^{\,k}_{l,w} > 0$ for $k > 0$, $\overrightarrow{d}_{j,p} \to \overrightarrow{d}_{l,w}/2 > 0$. The same statement holds also for $\overrightarrow{d}^{\,k}_{p,l}$, hence node p will converge to the midpoint between j and l. The number of switching is then limited in time also in this case.

5 Numerical Examples

In this section, we provide some numerical evidence of the effectiveness of the approach. The section is composed of two parts. In the first part, we show the convergence properties of the algorithm in a simple example. In the second one, we show how the algorithm converges to a schedule achieving a good coverage of the area the WSN is deployed on.

Fig. 2. Visibility Graph for the first scenario considered in Section 5.1

5.1 Convergence Properties

For this set of simulations, we consider a set of 6 nodes in two different scenarios: complete visibility and partial visibility. In the latter case, we assume the visibility graph shown in Figure 2. In both cases we consider a duration for the epoch (i.e., the period used for the schedule) equal to 5 and we set initial wake-up times to the value $w(0) = [1, 1.15, 1.1, 0.9, 3.5, 3.1]$ and the parameter $\alpha = 0.3$.

For partial visibility, the application of the wake-up scattering algorithm yields the evolution of the wake-up times depicted in Figure 3 (A). As it is possible to see, some of the nodes "overtake" each other. However, as we discussed above, if we study the dynamics of the distances between the wake-up times of the nodes that see each other (which can be considered as state variables), we deal with a convergent linear dynamic as shown in Figure 3 (B).

In the case of complete visibility, as discussed above we can make much stronger claims than simple convergence. Indeed, not only are we able to conclude that the wake-up times of the nodes will be evenly spaced out (in the steady state), but we can also compute the rate of convergence. In Figure 4 (A), we report the evolution of the wake-up times. As we discussed in Section 4, to properly describe the dynamics of the system, it is appropriate to rename the nodes so that their initial wake-up times are ordered in increasing order. After this renaming a convenient choice of state variables is:

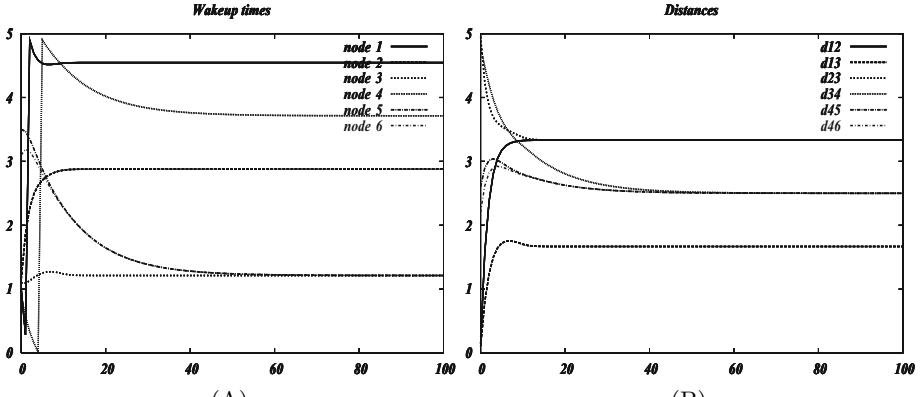


Fig. 3. Convergence in the case of partial visibility. (A) wake-up times, (B) state variables.

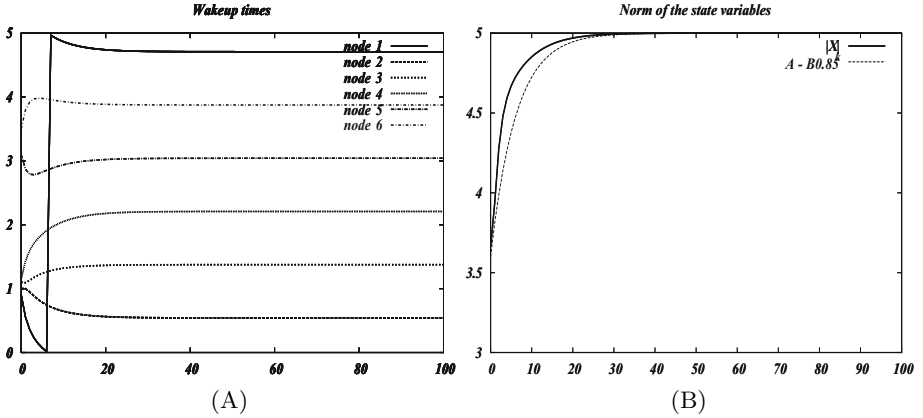
$$X = \begin{bmatrix} \overrightarrow{d}_{1,2} - \overrightarrow{d}_{2,3} \\ \overrightarrow{d}_{2,3} - \overrightarrow{d}_{3,4} \\ \cdots \\ \overrightarrow{d}_{5,6} - \overrightarrow{d}_{6,1} \\ \overrightarrow{d}_{1,2} + \overrightarrow{d}_{2,3} + \overrightarrow{d}_{3,4} + \overrightarrow{d}_{4,5} + \overrightarrow{d}_{5,6} + \overrightarrow{d}_{6,1} \end{bmatrix}$$

As discussed above, this vector provably converges to $[0, 0, \ldots, E]$ with a rate dictated by the second eigenvalue. Its dynamic matrix is a circulant matrix and its eigenvalues can be computed in closed form. The largest eigenvalue is 1 and the second one is 0.85 (see [11]) . Therefore, the convergence decay rate is $\approx 0.85^k$, as shown in Figure 4.(B).

5.2 Coverage Properties

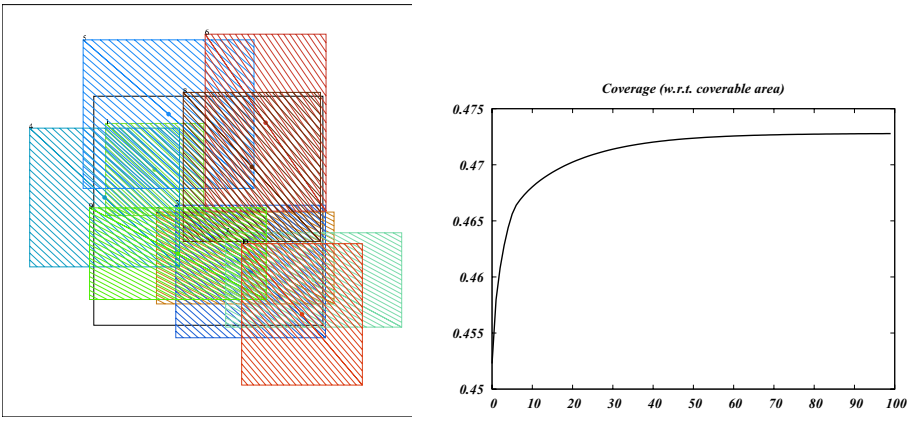
In order to show the performance of the wake-up scattering algorithm for the coverage problem, we consider a very simple deployment consisting of 10 nodes. For the sake of simplicity and without loss of generality, we consider a rectangular sensing range for the nodes. The nodes are randomly distributed over a 500×500 bi-dimensional area. The resulting deployment is shown in Figure 5.(A). We consider a period for the schedule equal to 5 time units and a wake-up interval for the nodes equal to 1. Therefore, each node is awake for 20% of the total time.

Several regions of the considered arena are covered by multiple nodes. Therefore, a good schedule is one where the wake-up times of nodes sharing "large" areas are far apart. Using the algorithm presented in [16], we come up with an optimal schedule, where an average of 52.94% of the "coverable" area (i.e., the area actually within the sensing range of the nodes) is actually covered. The application of the wake-up scattering algorithm over 100 iterations, assuming complete visibility between the nodes, produces the result shown in Figure 5.(B). The attained relative coverage is 47.3%. The deviation from the optimal solution is in this case lower



(A) (B)

Fig. 4. Convergence in the case of total visibility. (A) wake-up times, (B) state variables



(A) (B)

Fig. 5. The coverage scenario. (A) Spatial distribution of the nodes, (B) Evolution of the ratio between covered area and coverable area.

than 10% of the optimal coverage. The result is particularly interesting because the coverage problem is known to be exponential, while the wake-up scattering algorithm operates in polynomial time and is entirely distributed.

6 Conclusions

In this paper, we have presented convergence results of a distributed algorithm used for maximizing the lifetime of a WSN. We have focused our attention on an algorithm recently proposed in the literature, showing how its convergence can be cast into a stability problem for a linear switching system. We have found local stability results in the general case, and global stability results for specific topologies of the WSN.

Several issues have been left open and will offer interesting opportunities for future research. The first obvious point to address is to study global stability for general topologies. Another point is to study conditions under which the wake-up scattering algorithm produces a good coverage, developing improvements for the cases in which the result is not satisfactory. From a practical view–point, the scattering algorithm here presented reduces its performance if node clock synchronization is not guaranteed or in the presence of communication delays. Future analysis will consider algorithm convergence also in the presence of such random nuisances.

References

1. Alfieri, A., Bianco, A., Brandimarte, P., Chiasserini, C.F.: Maximizing system lifetime in wireless sensor networks. European Journal of Operational Research 127(1), 390–402 (2007)

2. Cărbunar, B., Grama, A., Vitek, J., Cărbunar, O.: Redundancy and coverage detection in sensor networks. ACM Transaction on Sensor Networks 2(1), 94–128 (2006)
3. Cao, Q., Abdelzaher, T., He, T., Stankovic, J.: Towards optimal sleep scheduling in sensor networks for rare-event detection. In: Proc. of the 4th Int. Symp. on Information Processing in Sensor Networks (IPSN) (April 2005)
4. Cardei, M., Thai, M., Li, Y., Wu, W.: Energy-efficient target coverage in wireless sensor networks. In: Proc. of INFOCOM (2005)
5. Carli, R., Fagnani, F., Speranzon, A., Zampieri, S.: Communication constraints in the average consensus problem. Automatica 44(3), 671–684 (2008)
6. Giusti, A., Murphy, A.L., Picco, G.P.: Decentralized Scattering of Wake-up Times in Wireless Sensor Networks. In: Langendoen, K.G., Voigt, T. (eds.) EWSN 2007. LNCS, vol. 4373, pp. 245–260. Springer, Heidelberg (2007)
7. Hsin, C., Liu, M.: Network coverage using low duty-cycled sensors: Random & coordinated sleep algorithms. In: Proc. of the 3th Int. Symp. on Information Processing in Sensor Networks (IPSN) (2004)
8. Jadbabaie, A., Lin, J., Morse, A.S.: Coordination of groups of mobile autonomous agents using nearest neighbor rules. IEEE Trans. on Automatic Control 48(6), 988–1001 (2003)
9. Liu, H., Jia, X., Wan, P., Yi, C., Makki, S., Pissinou, N.: Maximizing lifetime of sensor surveillance systems. IEEE/ACM Trans. on Networking 15(2), 334–345 (2007)
10. Marshall, J.A., Broucke, M.E., Francis, B.A.: Formations of vehicles in cyclic pursuit. IEEE Trans. on Automatic Control 49(11), 1963–1974 (2004)
11. Martínez, S., Bullo, F.: Optimal sensor placement and motion coordination for target tracking. Automatica 42(4), 661–668 (2006)
12. Martinez, S., Bullo, F., Cortes, J., Frazzoli, E.: On synchronous robotic networks - Part I: Models, tasks and complexity. IEEE Trans. on Automatic Control 52(12), 2199–2213 (2007)
13. Martinez, S., Bullo, F., Cortes, J., Frazzoli, E.: On synchronous robotic networks - Part II: Time complexity of rendezvous and deployment algorithms. IEEE Trans. on Automatic Control 52(12), 2214–2226 (2007)
14. Meyer, C.D. (ed.): Matrix analysis and applied linear algebra. Society for Industrial and Applied Mathematics, Philadelphia (2000)
15. Olfati-Saber, R., Fax, J.A., Murray, R.M.: Consensus and cooperation in networked multi–agent systems. Proc. of IEEE 95(1), 215–233 (2007)
16. Palopoli, L., Passerone, R., Picco, G.P., Murphy, A.L., Giusti, A.: Maximizing sensing coverage in wireless sensor networks through optimal scattering of wake-up times. Technical Report DIT-07-048, Dipartimento di Informatica e Telecomunicazioni, University of Trento (July 2007)
17. Slijepcevic, S., Potkonjak, M.: Power efficient organization of wireless sensor networks. In: Proc. of the IEEE Int. Conf. on Communications (ICC) (June 2001)
18. Tian, D., Georganas, N.D.: A coverage-preserving node scheduling scheme for large wireless sensor networks. In: First ACM Int. Wkshp. on Wireless Sensor networks and Applications (WSNA) (2002)
19. Ye, F., Zhong, G., Cheng, J., Lu, S., Zhang, L.: PEAS: A robust energy conserving protocol foo long-lived sensor networks. In: 3rd Int. Conf. on Distributed Computing Systems (ICDCS 2003) (May 2003)

Finite Automata as Time-Inv Linear Systems Observability, Reachability and More

Radu Grosu

Department of Computer Science, Stony Brook University
Stony Brook, NY 11794-4400, USA

Abstract. We show that regarding finite automata (FA) as discrete, time-invariant linear systems over semimodules, allows to: (1) express FA minimization and FA determinization as particular observability and reachability transformations of FA, respectively; (2) express FA pumping as a property of the FA's reachability matrix; (3) derive canonical forms for FAs. These results are to our knowledge new, and they may support a fresh look into hybrid automata properties, such as minimality. Moreover, they may allow to derive generalized notions of characteristic polynomials and associated eigenvalues, in the context of FA.

1 Introduction

The technological developments of the past two decades have nurtured a fascinating and very productive convergence of automata- and control-theory. An important outcome of this convergence are hybrid automata (HA), a popular modeling formalism for systems that exhibit both continuous and discrete behavior [3,11]. Intuitively, HA are extended finite automata whose discrete states correspond to the various modes of continuous dynamics a system may exhibit, and whose transitions express the switching logic between these modes.

HA have been used to model and analyze embedded systems, including automated highway systems, air traffic management, automotive controllers, robotics and real-time circuits. They have also been used to model and analyze biological systems, such as immune response, bio-molecular networks, gene-regulatory networks, protein-signaling pathways and metabolic processes.

The analysis of HA typically employs a combination of techniques borrowed from two seemingly disjoint domains: finite automata (FA) theory and linear systems (LS) theory. As a consequence, a typical HA course first introduces one of these domains, next the other, and finally their combination. For example, it is not unusual to first discuss FA minimization and later on LS observability reduction, without any formal link between the two techniques.

In this paper we show that FA and LS can be treated in a unified way, as FA can be conveniently represented as discrete, time-invariant LS (DTLS). Consequently, many techniques carry over from DTLS to FA. One has to be careful however, because the DTLS associated to FA are not defined over vector spaces, but over more general semimodules. In semimodules for example, the row rank of a matrix may differ from its column rank.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 194–208, 2009.

In particular, we show that: *(1) deterministic-FA minimization and nondeterministic-FA determinization [2] are particular cases of observability and reachability transformations [5] of FA, respectively; (2) FA pumping [2] is a property of the reachability matrix [5] associated to an FA; (3) FA admit a canonical FA in observable or reachable form, related through a standard transformation.*

While the connection between LS and FA is not new, especially from a language-theoretic point of view [2,6,10], our observability and reachability results for FA are to our knowledge new. Moreover, our treatment of FA as DTLS has the potential to lead to a knew understanding of HA minimization, and of other properties common to both FA and LS.

The rest of the paper is organized as follows. Section 2 reviews observability and reachability of DTLS. Section 3 reviews regular languages, FA and grammars, and introduces the representation of FA as DTLS. Section 4 presents our new results on the observability of FA. Section 5 shows that these results can be used to obtain by duality similar results for the reachability of FA. In Section 6 we address pumping and minimality of FA. Finally, Section 7 contains our concluding remarks and directions for future work.

2 Observability and Reachability Reduction of DTLS

Consider a *discrete, time-invariant linear system* (DTLS) with no input, only one output, and with no state and measurement noise. Its $[I, A, C]$, state-space description in left-linear form is then given as below [5]:[1]

$$x(0) = I, \quad x^T(t+1) = x^T(t)A, \quad y(t) = x^T(t)C$$

where x is the *state vector* of dimension n, y is the (scalar) *output*, I is the *initial state vector*, A is the *state transition matrix* of dimension $n \times n$, C is the *output matrix* of dimension $n \times 1$, and x^T is the transposition of x.

Observability. A DTLS is called *observable*, if its *initial state* I can be determined from a sequence of observations $y(0), \ldots, y(t-1)$ [5].

Rewriting the state-space equations in terms of $x(0) = I$ and the given output up to time $t-1$ one obtains the following output equation:

$$[y(0)\ y(1)\ \ldots\ y(t-1)] = I^T\,[C\ AC\ \ldots\ A^{t-1}C] = I^T\,O_t$$

Let X be the *state space* and $W = \text{span}[C\ AC \ldots A^kC \ldots]$ be the *A-cyclic subspace (A-CS) of X generated by C*. Since $C \neq 0$, the dimension of W is $1 \leq k \leq n$, and $[C\ AC \ldots A^{k-1}C]$ is a basis for W [7].[2] As a consequence, for each $t \geq k$, there exist scalars $a_0 \ldots a_{k-1}$ such that $A^tC = (C)a_0 + \ldots + (A^{k-1}C)a_{k-1}$.

If $k < n$ then setting $x^T O_t = \sum_{i=0}^{k-1} (A^iC) f_i(x_1, \ldots, x_n)$ to 0 results in k linear equations $f_i(x_1, \ldots, x_n) = 0$ in n unknowns, as A^iC are linearly independent for $i \in [0, k-1]$. Hence, there exist $n-k$ linearly independent vectors x, such that $x^T O_t = 0$, i.e. the dimension of the *null space* $\mathcal{N}(O_t) = \mathcal{N}(O_n)$ is $n-k$ and the *rank* $\rho(O_t) = \rho(O_n) = k$. If $k = n$ then $\mathcal{N}(O_n) = \{0\}$. The set $\mathcal{N}(O_n)$ is called the *unobservable space* of the system because $y(s) = 0$ for all s if $x(0) \in \mathcal{N}(O)$, and the matrix $O = O_n$ is called the *observability matrix*.

[1] The left-linear representation is more convenient in the following sections.

[2] This fact is used by the Cayley-Hamilton theorem.

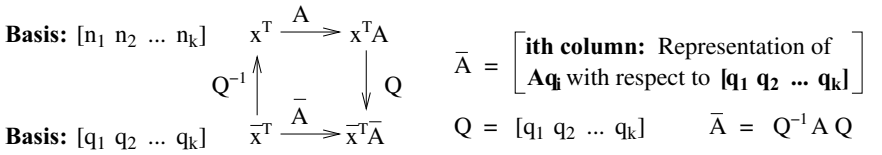


Fig. 1. Similarity transformations

If $\rho(O) = k < n$ then the *system can be reduced* to an observable system of dimension k. The reduction is done as follows. Pick columns $C, AC, \ldots, A^{k-1}C$ in O and add $n-k$ linearly independent columns, to obtain matrix Q. Then apply the *similarity transformation* $\overline{x}^T = x^T Q$, to obtain the following system:

$$\begin{array}{lcllcll} \overline{x}^T(t+1) & = & x^T(t)AQ & = & \overline{x}^T(t)Q^{-1}AQ & = & \overline{x}^T(t)\overline{A} \\ y(t) & = & x^T(t)C & = & \overline{x}^T(t)Q^{-1}C & = & \overline{x}^T(t)\overline{C} \end{array}$$

The transformation is shown in Figure 1, where n_i are the standard basis vectors for n-tuples (n_i is 1 in position i and 0 otherwise), and q_i are the column vectors in Q. Each column i of $\overline{A}$ is the representation of Aq_i in the basis Q, and $\overline{C}$ is the representation of C in Q. Since $q_1, \ldots, q_k$ is a basis for W, all $A_{i,j}$ and C_i for $j \leq k \leq i$ are 0. Hence, the new system has the following form:

$$\begin{array}{rcl} [\overline{x}_o^T(t+1) \quad \overline{x}_{\overline{o}}^T(t+1)] & = & [\overline{x}_o^T(t) \quad \overline{x}_{\overline{o}}^T(t)] \begin{bmatrix} \overline{A}_o & \overline{A}_{12} \\ 0 & \overline{A}_{\overline{o}} \end{bmatrix} \\ y(t) & = & [\overline{x}_o^T(t) \quad \overline{x}_{\overline{o}}^T(t)] \begin{bmatrix} \overline{C}_o \\ 0 \end{bmatrix} \end{array}$$

where $\overline{x}_o$ has dimension k, $\overline{x}_{\overline{o}}$ has dimension $n-k$, and $\overline{A}_o$ has dimension $k \times k$. Instead of working with the unobservable system $[A, C]$ one can therefore work with the *reduced*, observable system $[\overline{A}_o, \overline{C}_o]$ that produces the *same output*.

Reachability. Dually, the system $S = [I, A, C]$ is called *reachable*, if its *final state* C can be uniquely determined from $y(0), \ldots, y(t-1)$. Rewriting the state-space equation in terms of C, one obtains the following equation:

$$[y(0)\ y(1)\ \ldots\ y(t-1)]^T = [I\ (I^T A)^T\ \ldots\ (I^T A^{t-1})^T]^T C = R_t C$$

Since $R_t C = (C^T R_t^T)^T$ and $(I^T A^{t-1})^T = (A^T)^{t-1} I$, the reachability problem of $S = [I, A, T]$ is the observability problem of the *dual* system $S^T = [C^T, A^T, I^T]$. Hence, in order to study the reachability of S, one can study the observability of S^T instead. As for observability, $\rho(R_t) = \rho(R_n)$, where n is the dimension of the state space X. Matrix $R = R_n$ is called the *reachability matrix* of S.

Let $k = \rho(R)$. If $k = n$ then the system is reachable. Otherwise, there is an equivalence transformation $\overline{x}^T = x^T Q$ which transforms S into a reachable system $\overline{S}_r = [\overline{I}_r, \overline{A}_r, \overline{C}_r]$ of dimension k. The reachability transformation of S is the same as the observability transformation of S^T.

3 FA as Left-Linear DTLS

Regular expressions. A *regular expression* (RE) R over a finite set Σ and its associated semantics $L(R)$ are defined inductively as follows [2]: (1) $0 \in \text{RE}$ and

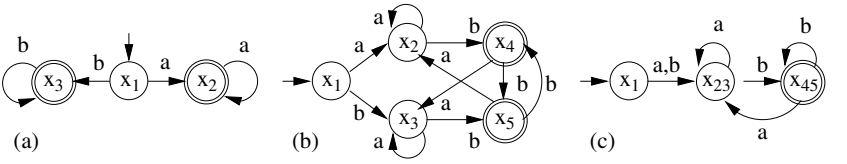


Fig. 2. (a) DFA M_1. (b) DFA M_2. (c) DFA M_3.

$L(0)=\emptyset$; (2) $\epsilon \in \text{RE}$ and $L(\epsilon)=\{\epsilon\}$; (3) If $a \in \Sigma$ then $a \in \text{RE}$ and $L(a)=\{a\}$; (4) If $P, Q \in \text{RE}$ then: $P+Q \in \text{RE}$ and $L(P+Q)=L(P) \cup L(Q)$; $P \cdot Q \in \text{RE}$ and $L(P \cdot Q)=L(P) \times L(Q)$; $P^* \in \text{RE}$ and $L(P^*)= \cup_{n \in \mathbb{N}} L(P)^n$. The denotations of regular expressions are called *regular sets*.[3]

For example, the denotation $L(R_1)$ of the regular expression $R_1 = aa^* + bb^*$, is the set of all strings (or words) consisting of more than one repetition of a or of b, respectively. It is custom to write a^+ for aa^*, so $R_1 = a^+ + b^+$. A *language* L is a subset of Σ^* and consequently any regular set is a (regular) language.

If two regular expressions R_1, R_2 denote the same set one writes $R_1 = R_2$. In general, one can write equations whose indeterminates and coefficients represent regular sets. For example, $X = X\alpha + \beta$. Its *least solution* is $X = \beta\alpha^*$ [2].

The structure $\mathcal{S} = (\Sigma^*, +, \cdot, 0, \epsilon)$ is a *semiring*, as it has the following properties: (1) $\mathcal{A} = (\Sigma^*, +, 0)$ is a commutative monoid; (2) $\mathcal{C} = (\Sigma^*, \cdot, \epsilon)$ is a monoid; (3) Concatenation left (and right) distributes over sum; (4) Left (and right) concatenation with 0 is 0. Matrices $\mathcal{M}_{m \times n}(\mathcal{S})$ over a semiring with the usual matrix sum and multiplication also form a semiring, but note that in a semiring there is *no inverse* operation for addition and multiplication, so the inverse of a square matrix is not defined in a classic sense. If $\mathcal{V} = \mathcal{M}_{m \times 1}(\mathcal{A})$ and *scalar multiplication* is concatenation then $\mathcal{R} = (\mathcal{V}, \mathcal{S}, \cdot)$ is an $\mathcal{S}$-*right semimodule* [10].

Finite automata. A *finite automaton* (FA) $M = (Q, \Sigma, \delta, I, F)$ is a tuple where Q is a finite set of *states*, Σ is a finite set of *input symbols*, $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ is the *transition function* mapping each state and input symbol to a set of states, $I \subseteq Q$ is the set of *initial states* and $F \subseteq Q$ is the set of *final states* [2]. If I and $\delta(q, a)$ are singletons, the FA is called *deterministic* (DFA); otherwise it is called *nondeterministic* (NFA). Three examples of FAs are shown in Figure 2.

Let δ^* extend δ to words. A word $w \in \Sigma^*$ is accepted by FA M if for any $q_0 \in I$, the set $\delta^*(q_0, w) \cap F \neq \emptyset$. The set $L(M)$ of all words accepted by M is called the *language of* M. For example, $L(M_1) = L(a^+ + b^+)$.

Grammars. A *left-linear grammar* (LLG) $G = (N, \Sigma, P, S)$ is a tuple where N is a finite set of *nonterminal symbols*, Σ is a finite set of *terminal symbols* disjoint from N, $P \subseteq N \times (N \cup \Sigma)^*$ is a finite set of *productions*[4] of the form $A \to Bx$ or $A \to x$ with $A, B \in N$ and $x \in \Sigma \cup \{\epsilon\}$, and $S \in N$ is the *start symbol* [2].

A word $a_1 \dots a_n$ is derived from S if there is a sequence of nonterminals $N_1 \dots N_n$ in N such that $S \to N_1\, a_1$ and $N_{i-1} \to N_i\, a_i$ for each $i \in [2, n]$. The set $L(G)$ of all words derived from S is called the *language of* G.

[3] The concatenation operator $\cdot$ is usually omitted when writing a regular expression.
[4] It is custom to write pairs $(x, y) \in P$ as $x \to y$.

Equivalence. FAs, LLGs and REs are equivalent, i.e. $L = L(M)$ for some FA M if and only if $L = L(G)$ for some LLG G and if and only if $L = L(E)$ for some RE E [2]. In particular, given an FA $M = (Q, \Sigma, \delta, I, F)$ one can construct an equivalent LLG $G = (Q \cup \{y\}, \Sigma, P, y)$ where P is defined as follows: (1) $y \to q$ for each $q \in F$, (2) $q \to \epsilon$, for $q \in I$, and (3) $r \to q\,a$ if $r = \delta(q, a)$. Replacing each set of rules $A \to \alpha_1, \ldots, A \to \alpha_n$ with one rule $A \to \alpha_1 + \ldots + \alpha_n$ leads to a more concise representation. For example the LLG G_1 derived from M_1 is:

$$x_1 \to \epsilon, \quad y \to x_2 + x_3, \quad x_2 \to x_1\mathtt{a} + x_2\mathtt{a}, \quad x_3 \to x_1\mathtt{b} + x_3\mathtt{b}$$

Each nonterminal denotes the set of words derivable from that nonterminal. One can regard G_1 as a linear system S over REs. One can also *regard G_1 as a discrete, time-invariant linear system* (DTLS) S_1 defined as below:

$$x^T(t+1) = x^T(t)A, \quad y(t) = x^T(t)C$$

$$I = \begin{bmatrix} \epsilon \\ 0 \\ 0 \end{bmatrix} \quad A = \begin{bmatrix} 0 & \mathtt{a} & \mathtt{b} \\ 0 & \mathtt{a} & 0 \\ 0 & 0 & \mathtt{b} \end{bmatrix} \quad C = \begin{bmatrix} 0 \\ \epsilon \\ \epsilon \end{bmatrix}$$

The initial state of S_1 is the same as the initial state of DFA M_1 and it corresponds to the production $x_1 \to \epsilon$ of LLG G_1. The output matrix C sums up the words in x_2 and x_3. It corresponds to the final states of DFA M_1 and to the production $y \to x_2 + x_3$ in LLG G_1. Matrix A is obtained from DFA M_1 by taking $v \in A_{ij}$ if $\delta(x_i, v) = x_j$ and $A_{ij} = 0$ if $\delta(x_i, v) \neq x_j$ for all $v \in \Sigma$. The set of all outputs of S_1 over time is $\cup_{t \in \mathbb{N}}\{y(t)\} = I^T A^* C = L(M_1)$.

Matrix A^* can be computed in $\mathcal{R}$ as described in [6]. This provides one method for computing $L(M)$. Alternatively, one can use the least solution of an RE equation, and apply *Gaussian elimination*. This method is equivalent to the *rip-out-and-repair* method for converting an FA to an RE [2].

In the following, all four equivalent representations, RE, FA, LLG and DTLS, of a finite automaton, are simply referred to as an FA. The *observability/reachability* problem for an FA is to determine its initial/final state given $y(t)$ for $t \in [0, n-1]$. In vector spaces, these are unique if the rank of O/R is n. In semimodules however, the *row rank* is generally different from the *column rank*.

4 Observability Transformations of FA

Lack of finite basis. Let I be a set of indices and $\mathcal{R}$ be an $\mathcal{S}$-semimodule. A set of vectors $Q = \{q_i \mid i \in I\}$ in $\mathcal{R}$ is called *linearly independent* if no vector $q_i \in Q$ can be expressed as a linear combination $\sum_{j \in (I-i)} q_j\, a_j$ of the other vectors in Q, for arbitrary scalars $a_j \in \mathcal{S}$. Otherwise, Q is called *linearly dependent*. The independent set Q is called a *basis* for $\mathcal{R}$ if it covers $\mathcal{R}$, i.e. $span(Q) = \mathcal{R}$ [4].

$$E = \begin{array}{c} \begin{matrix} E_0 & E_1 & E_2 \end{matrix} \\ \begin{bmatrix} 0 & \mathtt{1a2+1b3} & \mathtt{1a2a2+1b3b3} \\ 2 & \mathtt{2a2} & \mathtt{2a2a2} \\ 3 & \mathtt{3b3} & \mathtt{3b3b3} \end{bmatrix} \end{array} \quad O = \begin{array}{c} \begin{matrix} C & AC & A^2C \end{matrix} \\ \begin{bmatrix} 0 & \mathtt{a+b} & \mathtt{a^2+b^2} \\ \epsilon & \mathtt{a} & \mathtt{a^2} \\ \epsilon & \mathtt{b} & \mathtt{b^2} \end{bmatrix} \end{array} \begin{matrix} x_1 \\ x_2 \\ x_3 \end{matrix}$$

Now consider DFA M_1. Its observability matrix O is given above. Each row i of O consists of the words accepted by M_1 starting in state x_i sorted by their length in increasing order. Each column j of O is the vector $A^{j-1}C$, consisting of the accepted words of length $j-1$ starting in x_i. The corresponding executions E_0, E_1 and E_2 of DFA M_1 are also given above.

The columns of O belong to the *A-cyclic subspace of X generated by C* (A-CS), which has finite dimension in any vector space. In the $\mathcal{S}$-semimodule $\mathcal{R}$, where $\mathcal{S}$ is the semiring of REs, however, A-CS may not have a finite basis.

For example, for DFA M_1 it is not possible to find REs r_{i_j} and vectors $A^{i_j}C$ such that $A^iC = \sum_{j=1}^{k}(A^{i_j}C)r_{i_j}$, for $i_j < i$. Intuitively, abstracting out the states of an FA from its executions, eliminates linear dependencies.

The state information included in E_1 and E_2 allows to capture their linear dependence: *E_2 is obtained from E_1 by substituting the last occurrence of states* 2 *and* 3 *with the loops* 2a2 *and* 3a3*, respectively.* Regarding substitution as a multiplication with a scalar, one can therefore write $E_2 = E_1(\mathtt{2a2} + \mathtt{3b3})$.

Indexed boolean matrices. In the above multiplication we tacitly assumed that, e.g. $(\mathtt{1a2})(\mathtt{3b3}) = 0$, because a b-transition valid in state 3 cannot be taken in states 1 and 2. Treating *independently* the σ-successors/predecessors of an FA $M = (Q, \Sigma, \delta, I, F)$, for each input symbol $\sigma \in \Sigma$, allows to capture this intuition in a "stateless" way. Formally, this is expressed with *indexed boolean matrices* (IBM), defined as follows [12]: (1) $C_i = (i \in F)$; (2) $I_i = (i \in I)$; (2) For each $\sigma \in \Sigma$, $(A_\sigma)_{ij} = (\delta(i,\sigma) = j)$; and (3) $A_{\sigma_1 \ldots \sigma_n} = A_{\sigma_1} \ldots A_{\sigma_n}$. For example, one obtains the following matrices for the DFA M_1:

$$I = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad A_a = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad A_b = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad C = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

Indexing enforces a word by word analysis of acceptance and ensures, for example for M_1, that $A_{ab}C = A_a(A_bC) = 0$. Consequently, for every word $w \in \Sigma^*$ the vector A_wC has row i equal to 1, if and only if, w is accepted starting in x_i.

Ordering all vectors $A_{w_i}C$, for $w_i \in \Sigma^i$, in lexicographic order, results in a *boolean observability matrix* $O = [A_{w_0}C \ldots A_{w_m}C]$. This matrix has n rows and $|\Sigma|^n - 1$ columns. Its column rank *is* the dimension of the A-CS W of the *boolean semimodule* $\mathcal{B}$ because all $O_{ij} \in \mathbb{B}$. Hence it is finite and less than $2^n - 1$.

$$O = \begin{array}{c} \begin{matrix} C & A_aC & A_bC & A_{aa}C & A_{ab}C & A_{ba}C & A_{bb}C \end{matrix} \\ \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array} \begin{matrix} x_1 \\ x_2 \\ x_3 \end{matrix}$$

For example, matrix O for M_1 is shown above. It is easy to see that vectors C, A_aC and A_bC are independent. Moreover, $A_{aa}C = A_aC$, $A_{bb}C = A_bC$. Hence, all vectors A_wC, for $w \in \{a, b\}^*$, are generated by the basis $Q = [C, A_aC, A_bC]$.

The structure of O is intimately related to the states and transitions of the associated FA. Column C is the set of accepting states, and each column A_wC

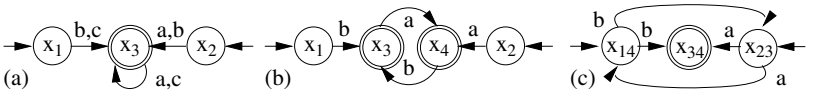


Fig. 3. (a) FA M_4. (b) FA M_5. (c) FA M_6.

is the set of states that can reach C by reading word w. In other words, A_wC is the set of all w-predecessors of C.

In the following we do not distinguish between an FA and its IBM representation. The latter is used to find appropriate bases for similarity transformations and prove important properties about FA. To this end, let us first review and prove three important properties about the ranks of boolean matrices.

Theorem 1. *(Rank independence) If $n \geq 3$ then the* row rank $\rho_r(O)$ *and the* column rank $\rho_c(O)$ *of a boolean matrix O may be different.*

Proof. Consider the observability matrices[5] of FA M_4 and M_5 shown in Figure 3: $\rho_r(O(M_4)) = 3$, $\rho_c(O(M_4)) = 4$, and $\rho_r(O(M_5)) = 4$, $\rho_c(O(M_5)) = 3$.

$$O(M_4) = \begin{array}{c} \begin{matrix} C & A_aC & A_bC & A_cC \end{matrix} \\ \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix} \begin{matrix} x_1 \\ x_2 \\ x_3 \end{matrix} \end{array} \qquad O(M_5) = \begin{array}{c} \begin{matrix} C & A_aC & A_bC \end{matrix} \\ \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \begin{matrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{matrix} \end{array}$$

To ensure that an FA is transformed to an equivalent DFA, it is convenient to introduce two more ranks: $\rho_r^d(O)$ and $\rho_c^d(O)$. They represent the number of *distinct* rows and columns in O, respectively. Hence, these ranks consider only linear dependencies in which the sum is identical to its summands.

Theorem 2. *(Rank bounds) The various row and column ranks are bounded and related to each other by the following inequalities:*

$$1 \leq \rho_r(O) \leq \rho_r^d(O) \leq n,\ 1 \leq \rho_c(O) \leq \rho_c^d(O) \leq 2^n - 1,\ 1 \leq \rho_c(O) \leq C^n_{\lceil n/2 \rceil} + \lfloor n/2 \rfloor$$

Proof. First and second inequalities are obvious. For the third observe that: (1) The set of combinations C_i^n is independent; (2) It covers all C_j^n with $j > i$; (3) Only $i-1$ independent vectors may be added to C_i^n from all C_j^n, with $j < i$.

The A-CS of $\mathcal{B}$ is very similar to the A-CS of a vector space. For example, let A^kC be the set of all vectors A_wC with $|w| = k$. Then the following holds.

Theorem 3. *(Rank computation) If all vectors in A^kC are linearly dependent on a basis Q for $[C\, AC \ldots A^{k-1}C]$, then so are all the ones in A^jC, with $j \geq k$.*

Proof. The proof is identical to the one for vector spaces, except that induction is on the length of words in A_wC, and A^kC are sets of vectors.

[5] We show only the basis columns of the observability matrix.

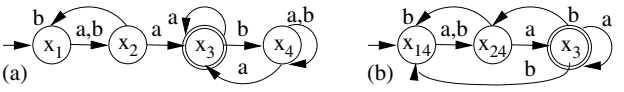


Fig. 4. (a) FA M_7. (b) FA M_8.

Observability transformations. The four ranks discussed above suggest the definition of *four equivalence transformations* $\overline{x}^T = x^T Q$, where Q consists of the independent (or the distinct), rows (or columns) in O, respectively. Each state $q \in Q$ of the resulting FA $\overline{M}$, is therefore a subset of the states of M, and each σ-transition to q in $\overline{M}$ is computed by representing its σ-predecessor $A_\sigma q$ in Q.

Row-basis transformations. These transformations utilize *sets of observably-equivalent states in* M to build the independent states $q \in Q$ of $\overline{M}$. The length of the observations, necessary to characterize the equivalence, is determined by Theorem 3. The equivalence among state-observations itself, depends on whether $\rho_r(O)$ (linear equivalence) or $\rho_r^d(O)$ (identity equivalence) is used.

Using $\rho_r(O)$, one fully exploits linear dependencies to reduce the number of states in $\overline{M}$. For example, suppose that $x_3 = x_1 + x_2$, and that x_1 and x_2 are independent. Then one can replace the states x_1, x_2 and x_3 in M, with states $q_1 = \{x_1, x_3\}$ and $q_2 = \{x_2, x_3\}$ in $\overline{M}$. This generalizes to multiple dependencies, and each new state $q \in Q$ contains *only one* independent state x. Consequently, the language $L(q) = L(x)$. Among the states $q \in Q$, the state C is accepting, and each q that contains an initial state in M is initial in $\overline{M}$.

The transitions among states $q \in Q$ are inferred from the transitions in M. The general rule is that $q_i \xrightarrow{\sigma} q_j$, if all states in q_i are σ-predecessors of the states in q_j. However, as Q is not necessarily a column basis, the σ-predecessor of a state like q_1 above, could be either x_1 or x_3, which are not in Q. Extending x_1 to q_1 does not do any harm, as $L(q_1) = L(x_1)$. Ignoring state x_3 does not do any harm either, as x_3 is covered by x_1 and x_2, possibly on some other path. These completion rules are necessary when computing the "inverse" of Q, i.e. representing AQ in Q to obtain $\overline{A}$.

Theorem 4. *(Row reduction) Given FA M with $\rho_r(O) = k < n$, let R be a* row basis *for O. Define $Q = [q_1, \ldots, q_k]$ as follows: for every $i \in [1, k]$ and $j \in [1, n]$, if row O_j is linearly dependent on R_i then $q_{ij} = 1$; otherwise $q_{ij} = 0$. Then a change of basis $\overline{x}^T = x^T Q$ obeying above completion rules results in FA $\overline{M}$ that: (1) has same output; (2) has states with independent languages.*

Proof. (1) States q satisfy $L(q) = L(x)$, where x is the independent state in q. Transitions $\overline{A}_\sigma = Q^{-1}[A_\sigma q_1 \ldots A_\sigma q_n]$, have $A_\sigma q_i$ as the σ-predecessors of states in q_i. The role of Q^{-1} is to represent $A_\sigma q_i$ in Q. If this fails, it is corrected as discussed above. (2) Dependent rows have been identified with their summands.

For example, consider FA M_7 in Figure 4(a). The observability matrix O of M_7 is given below:[6]

[6] We show only part of the columns in O.

$$O(M_7) = \begin{array}{c} \begin{matrix} C & A_aC & A_bC & A_{aa}C & A_{ab}C & A_{ba}C & A_{bb}C \end{matrix} \\ \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \end{array} \begin{matrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{matrix} \qquad Q(M_7) = \begin{array}{c} \begin{matrix} q_1 & q_2 & q_3 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \end{array}$$

Row $x_4 = x_1 + x_2$. This determines the construction of Q as shown above. Using Q in $\overline{x}^T = x^T Q$, results in FA M_8 shown in Figure 4(b). Note that $A_a q_1 = x_4$ has been removed when representing $A_a q_1$ in Q.

Using $\rho_r(O)$ typically results in an NFA, even when starting with a DFA. This is because vectors in Q may have overlapping rows, due to linear dependencies in O. The use of $\rho_r^d(O)$ ensures a resulting DFA, as columns do not overlap.

Identity equivalence also simplifies the transformation. First, Theorem 3 and the computation of ρ_r^d can be performed on-the-fly as a *partition-refinement*: $[C]$, partitions states, based on observations of length 0; $[C\ AC]$, further distinguishes the states in previous partition, based on observations of length 1; and so on. Second, no completion is ever necessary, as Aq is always representable in Q.

Theorem 5. *(Deterministic row reduction) Given an FA M with $\rho_r^d(O) = k < n$ proceed as in Theorem 4 but using $\rho_r^d(O)$. Then if M is a DFA, then so is $\overline{M}$.*

Proof. (1) Theorem 4 ensures correctness. (2) States in Q are disjoint. Hence, no row in $\overline{A} = Q^{-1}AQ$ has two entries for the same input symbol.

For example, let us apply Theorem 5 to the DFA M_2 in Figure 2(b). The corresponding observability matrix is shown below:

$$O(M_2) = \begin{array}{c} \begin{matrix} C & A_bC & A_{ab}C & A_{bb}C \end{matrix} \\ \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \end{array} \begin{matrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{matrix} \qquad Q(M_2) = \begin{array}{c} \begin{matrix} q_1 & q_2 & q_3 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \end{array}$$

Rows $x_2 = x_3$ and $x_4 = x_5$. This determines the construction of the basis Q as shown above. Using this basis in the equivalence transformation, results in DFA M_3, which is shown graphically in Figure 2(c).

Corollary 1 (Myhill-Nerode theorem). *Theorem 5 is equivalent to the DFA minimization algorithm of the Myhill-Nerode theorem [2].*

Column-basis transformations. These transformations pick Q as a column basis for O. The definition of basis depends on the notion of linear independence used, and this also impacts the column rank computation via Theorem 3.

Using $\rho_r(O)$, one fully exploits linear dependencies, and chooses a minimal column basis Q as the states of $\overline{M}$. The transitions of $\overline{M}$ are then determined by representing all the predecessors AQ of the states $Q = [q_1 \dots q_k]$ of $\overline{M}$ in Q. In contrast to the general row transformation, Aq_i, for $i \in [1, k]$, is representable in Q, as Q is a column basis for O. Hence, no completion is ever necessary. Like in vector spaces, the resulting matrices $\overline{A}$ are in *companion form*.

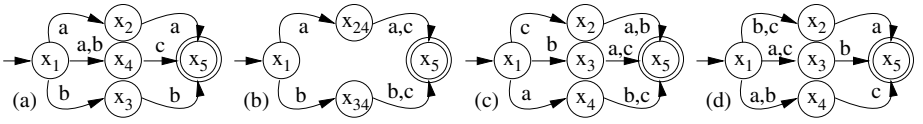


Fig. 5. (a) NFA M_9. (b) DFA M_{10}. (c) DFA M_{11}. (d) NFA M_{12}.

Theorem 6. *(Column reduction) Given an FA M with $\rho_c(O) = k < n$. Define Q as a column basis of O. Then a change of basis $\overline{x}^T = x^T Q$ results in FA $\overline{M}$ with: (1) same output; (2) states with a distinguishing accepting word.*

Proof. (1) Transitions $\overline{A}_\sigma = Q^{-1}[A_\sigma q_1 \ldots A_\sigma q_n]$, have $A_\sigma q_i$ as the σ-predecessors of states in q_i. The role of Q^{-1} is to represent $A_\sigma q_i$ in Q, and this never fails. (2) Dependent columns in O have been identified with their summands.

For example, consider the FA M_5 shown in Figure 3(b) and its associated observability matrix, shown below of Figure 3(b). No row-rank reduction applies, as $\rho_r(O) = 4$. However, as $\rho_c(O) = 3$, one can apply a column-basis reduction, with Q as the first three columns of O. The resulting FA is shown in Figure 3(c).

The column-basis transformation for $\rho_c^d(O)$ simplifies, as dependence reduces to identity. Moreover, in this case $\overline{M}$ can be constructed *on-the-fly*, as follows: Start with $Q, Q_n = [C]$. Then repeatedly remove the first state $q \in Q_n$, and add the transition $p \xrightarrow{\sigma} q$ to $\overline{A}$ for each $p \in Aq$. If $p \notin Q$, then also add p at the end of Q and Q_n. Stop when Q_n is empty. The resulting $\overline{M}^T$ is deterministic.

Theorem 7. *(Deterministic column transformation) Given FA M proceed as in Theorem 6 but using $\rho_c^d(O)$. Then $\overline{M}^T$ is a DFA with $|Q| \leq 2^n - 1$.*

Proof. Each row of $\overline{A}_\sigma^T$ has a single 1 for each input symbol $\sigma \in \Sigma$.

For example, consider the FA M_{11} shown in Figure 5(c). Construct the basis Q by selecting all columns in O. Using this basis in the equivalence transformation $\overline{x}^T = x^T Q$, results in the FA M_{12} shown in Figure 5(d).

5 Reachability Transformations of FA

The boolean semiring $\mathcal{B}$ is *commutative*, that is $ab = ba$ holds. When viewed as a semimodule, left linearity is therefore equivalent to right linearity, that is $\sum_{i \in I} x_i\, a_i = \sum_{i \in I} a_i\, x_i$. This in turn means that $(A\,B)^T = B^T\, A^T$.

Consequently, in $\mathcal{B}$ the *reachability* of an FA $M = [I, A, C]$ is *reducible* to the *observability* of the FA $M^T = [C^T, A^T, I^T]$, and all the results and transformations in Section 4, can be directly applied without any further proof!

For illustration, consider the FA M_9 shown in Figure 5(a). The reachability matrix $R^T(M_9)$ is given below. It is identical to $O(M_9^T)$.

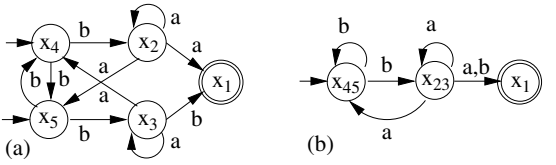


Fig. 6. (a) NFA M_{13}. (b) NFA M_{14}.

$$R^T(M_9) = \begin{array}{c} \begin{matrix} I & A_a^T I & A_b^T I & A_{aa}^T I & A_{bb}^T I & A_{ac}^T I & A_{bc}^T I \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \end{array} \begin{matrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{matrix} \qquad Q(M_9) = \begin{array}{c} \begin{matrix} q_1 & q_2 & q_3 & q_4 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

Each row i of R^T corresponds to state x_i. A column $A_w^T I$ of R^T is 1 in row i iff x_i is reachable from I with w^R, or dually, if state x_i accepts w in M^T.[7]

Row-basis transformations. These transformations utilize *sets of reachability-equivalent states in M* to build the independent states $q \in Q$ of $\overline{M}$. These states are, as discussed before, the observability equivalent states of M^T.

Theorem 8. *(Row reduction) Given an FA M, Theorem 4 applied to M^T results in an FA $\overline{M}$ with: (1) same output; (2) states with independent sets of reaching words.*

For example, in $R^T(M_9)$ above, row $x_4 = x_2 + x_3$. This determines the construction of the basis Q, also shown above. Using this basis in the equivalence transformation, results in the DFA M_{10} shown in Figure 5(b).

Identifying linearly dependent states with their generators and repairing lone σ-successors might preclude $\overline{M}^T$ to be a DFA, even if M^T was a DFA. Identifying only states with identical reachability however, ensures it.

Theorem 9. *(Deterministic row reduction) If M^T is a DFA, then Theorem 5 applied to M^T ensures that $\overline{M}^T$ is also a DFA.*

For example, let us apply Theorem 9 to the NFA M_{13} shown in Figure 6(a), the dual of the DFA M_2 shown in Figure 2(b). Hence, $M_{11}^T = M_2$ is a DFA. The reachability matrix $R^T(M_{13})$ is shown below. It is identical to $O(M_2)$.

$$R^T(M_{13}) = \begin{array}{c} \begin{matrix} I & A_b^T I & A_{ab}^T I & A_{bb}^T I \end{matrix} \\ \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \end{array} \begin{matrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{matrix} \qquad Q(M_{13}) = \begin{array}{c} \begin{matrix} q_1 & q_2 & q_3 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \end{array}$$

[7] We write w^R for the reversed form of w.

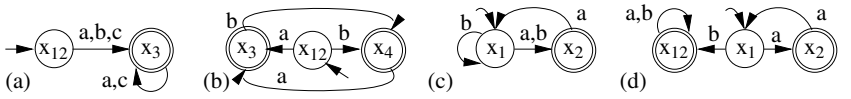


Fig. 7. (a) FA M_{15}. (b) FA M_{16}. (c) FA M_{17}. (d) FA M_{18}.

Rows $x_2 = x_3$ and $x_4 = x_5$. This determines the construction of the basis Q as shown above. Using this basis in the equivalence transformation, results in NFA M_{14}, shown graphically in Figure 6(c). The FA M_{14}^T is a DFA, and $M_{14}^T = M_3$.

Column-basis transformations. Given an FA M, these transformations construct FA $\overline{M}$ by choosing a column basis of R^T as the states Q of $\overline{M}$.

The general form of the transformations uses the full concept of linear dependency, in order to look for a column basis in R^T. Hence, this transformation computes the smallest possible column basis.

Theorem 10. *(Column reduction) Given FA M, Theorem 6 used on M^T results in FA $\overline{M}$ with: (1) same output; (2) states reached with a distinguishing word.*

Consider the NFA M_4 shown in Figure 3(a). Neither a row nor a column-basis observability reduction is applicable to M_4. However, one can apply a column-basis reachability reduction to M_4. The matrix $R^T(M_4)$ is given below.

$$R^T(M_4) = \begin{array}{c} \begin{matrix} I & A_a^T I & A_b^T I & A_c^T I \end{matrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \end{array} \begin{matrix} x_1 \\ x_2 \\ x_3 \end{matrix} \qquad Q(M_4) = \begin{array}{c} \begin{matrix} q_1 & q_2 \end{matrix} \\ \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \end{array}$$

Columns 1 and 2 form a basis for R^T. This determines the construction of Q as shown above. Using Q in $\overline{x}^T = x^T Q$ results in NFA M_{15}, shown in Figure 7(a).

In this case, the column-basis reachability transformation is identical to a row-basis reachability transformation. Consequently, the latter transformation would not require any automatic completion of the σ-successors $q^T A_\sigma$ of $q \in Q$.

Given an FA M, the deterministic column-basis transformation, with column rank $\rho_c^d(R^T)$, always constructs a DFA $\overline{M}$. This construction is dual to the deterministic column-basis observability transformation.

Theorem 11. *(Deterministic column transformation) Given an FA M, Theorem 7 applied to M^T results in the DFA $\overline{M}$.*

Consider for example the NFA M_{17} shown in Figure 7(c). Its reachability matrix $R^T(M_{17})$ is given below, where only the interesting columns are shown.

$$R^T(M_{17}) = \begin{array}{c} \begin{matrix} I & A_a^T I & A_b^T I \end{matrix} \\ \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \end{array} \begin{matrix} x_1 \\ x_2 \end{matrix} \qquad Q(M_{17}) = \begin{array}{c} \begin{matrix} q_1 & q_2 & q_3 \end{matrix} \\ \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \end{array}$$

As columns one and two form a basis for R^T, the general column-basis transformation is the identity. The deterministic one is not, as it includes all distinct

columns of R^T in Q, as shown above. Using Q in $\overline{x}^T = x^T Q$ results in the DFA M_{18} shown in Figure 7(d). This DFA has one more state, compared to M_{17}.

Applying a deterministic column-basis transformation to an FA M, does not necessarily increase the number of states of M. For example, applying such a transformation to NFA M_6 shown in Figure 3(c), results in the DFA M_{16} shown in Figure 7(b), which has the same number of states as M_6. Moreover, in this case, the general and the deterministic column-basis transformations coincide.

Corollary 2 (NFA determinization algorithm). *Theorem 11 is equivalent to the NFA determinization algorithm [2].*

6 The Pumping Lemma and FA Minimality

In previous sections we have shown that a control-theoretic approach to FA complements, and also allows to extend the reach of, the graph-theoretic approach. In this section we give two additional examples: An alternative proof of the pumping lemma [2]; A alternative approach to FA minimization. Both take advantage of the observability and reachability matrices.

Theorem 12 (Pumping Lemma). *If L is a regular set then there exists a constant p such that every word $w \in L$ of length $|w| \geq p$ can be written as xyz, where: (1) $0 < |y|$, (2) $|xz| \leq p$, and (2) $xy^i z \in L$ for all $i \geq 0$*

Proof. Consider a DFA M accepting L. Since M is deterministic, each column of R^T is a standard basis vector n_i, and there are at most n such distinct columns in R^T. Hence, for every word w of length greater than n, there are words $xyz = w$ satisfying (1) and (2) such that $I^T A_x = I^T A_{xy}$. Since $I^T A_{xy^i} = I^T A_{xy} A_{y^{i-1}}$, it follows that $I^T A_{xy^i z} C = I^T A_w C$, for all $i \geq 0$.

Canonical Forms. Row- and column-basis transformations are related to each other. Let $Q_c \in \mathcal{M}_{i \times j}(\mathcal{B})$, $Q_r \in \mathcal{M}_{i \times k}(\mathcal{B})$ be the observability column and row basis for an FA M. Let $A_c = Q_c^{-1} A Q_c$ and $A_r = Q_r^{-1} A Q_r$.

Theorem 13 (Row and column basis). *There is a matrix $R \in \mathcal{M}_{k \times j}(\mathcal{B})$ such that: (1) $Q_c = Q_r R$; (2) $A_c = R^{-1} A_r R$; (3) $A_r = R A_c R^{-1}$.*

Proof. (1) Let $B(m)$ be the index in O of the independent row of $q_m \in Q_r$ and $C(n)$ be the index in O of the independent column $q_n \in Q_c$, and define $R_{mn} = O_{B(m)C(n)}$, for $m \in [1,k]$, $n \in [1,j]$. Then $Q_c = Q_r R$; (2) As a consequence $A_c = (Q_r R)^{-1} A (Q_r R) = R^{-1} A_r R$; (3) This implies that $A_r = R A_c R^{-1}$.

Hence, A_r is obtained through a reachability transformation with column basis R after an observability transformation with column basis Q_c. Let $\mathcal{O}$ and $\mathcal{R}$ be the column basis observability and reachability transformations, respectively. We call $M_o = \mathcal{O}(\mathcal{R}(M))$ and $M_r = \mathcal{R}(\mathcal{O}(M))$ the *canonical observable* and *reachable* FAs of M, respectively.

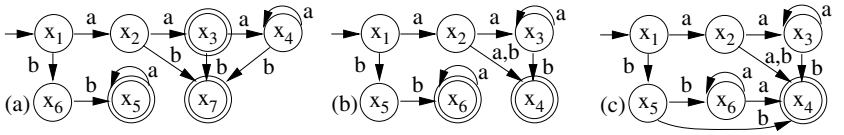


Fig. 8. (a) DFA M_{19}. (b) NFA M_{20}. (c) NFA M_{21}.

Theorem 14 (Canonical FA). *For any FA M, $\mathcal{R}(M_o)=M_r$ and $\mathcal{O}(M_r)=M_o$.*

Minimal FA. Canonical FAs are often *minimal* wrt. to the number of states. For example, M_{11} and M_{12} in Figure 5 are both minimal FAs. Moreover, FA M_{11} is canonical reachable and FA M_{12} is canonical observable.

For certain FAs however, the canonical FAs are not minimal. A *necessary condition* for the lack of minimality, is the existence of a weaker form of linear dependence among the basis vectors of the observability/reachability matrices: A set of vectors $Q=\{q_i \mid i \in I\}$ in $\mathcal{R}$ is called *weakly linearly dependent* if there are two disjoint subsets $I_1, I_2 \subset I$, such that $\sum_{i \in I_1} q_i = \sum_{i \in I_2} q_i$ [8].

For example, the DFA M_{19} in Figure 8(a) has the canonical reachable FA M_{20} shown in Figure 8(b), which is minimal. The observability matrix of M_{20} shown below, has 7 independent columns. The canonical observable FA of M_{19} and M_{20} has therefore 7 states! As a consequence, it is not minimal. Note however, that $A_bC + A_{bb}C = A_{ab}C + A_{ba}C$. Hence, the 7 columns are weakly dependent.

$$O(M_{20}) = \begin{array}{c} \begin{matrix} C & A_aC & A_bC & A_{aa}C & A_{ab}C & A_{ba}C & A_{bb}C \end{matrix} \\ \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{matrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{matrix} \end{array} \qquad Q(M_{20}) = \begin{array}{c} \begin{matrix} q_1 & q_2 & q_3 & q_4 & q_5 & q_6 \end{matrix} \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

Theorem 15 (Minimal FA). *Given the observability matrix O of an FA M, choose Q as a* set basis *[13] of O, such that AQ is representable in Q. Then the equivalence transformation $\overline{x}^T = x^T Q$ results in a minimal automaton.*

Alternatively, minimization can be reduced to computing the minimal boolean relation corresponding to O. For example, the Karnaugh blocks [9] in $O(M_{20})$ provide several ways of constructing Q. One such way is $Q(M_{20})$ shown above, where one block is the first column in $O(M_{20})$, and the other blocks correspond to its rows. The resulting FA is M_{21}. Both alternatives lead to NP-complete algorithms. Reachability is treated in a dual way, by manipulating R.

Since all equivalent FAs admit an equivalence transformation resulting in the same DFA, and since from this DFA one can obtain all other FAs through an equivalence transformation, *all FAs are related through an equivalence transformation!* This provides a cleaner way of dealing with the minimal FAs, when compared to the *terminal FA (incorporating all other FA)*, discussed in [1].

7 Conclusions

We have shown that regarding finite automata (FA) as discrete, time-invariant linear systems over semimodules, allows to unify DFA minimization, NFA determinization, DFA pumping and NFA minimality as various properties of observability and reachability transformations of FA. Our treatment of observability and reachability may also allow us to generalize the Cayley-Hamilton theorem to FA and derive a characteristic polynomial. In future work, we would therefore like to investigate this polynomial and its associated eigenvalues.

Acknowledgments. I would like to thank Gene Stark and Gheorghe Stefanescu for their insightful comments to the various drafts of this paper. This research was supported in part by NSF Faculty Early Career Award CCR01-33583.

References

1. Nivat, M., Arnold, A., Dicky, A.: A note about minimal nondeterministic automata. EATCS 45, 166–169 (1992)
2. Aho, A.V., Ullman, J.D.: The Theory of Parsing, Translation and Compiling. Prentice-Hall, Englewood Cliffs (1972)
3. Alur, R., Coucoubetis, C., Halbwachs, N., Henzinger, T.A., Ho, P.H., Nicolin, X., Olivero, A., Sifakis, J., Yovine, S.: The algorithmic analysis of hybrid systems. Theoretical Computer Sciences 138, 3–34 (1995)
4. Beasley, L.B., Guterman, A.E., Jun, Y.B., Song, S.Z.: Linear preservers of extremes of rank inequalities over semirings: Row and column ranks. Linear Algebra and its Applications 413, 495–509 (2006)
5. Chen, C.T.: Linear System Theory Design. Holt, Reinhart and Winston (1984)
6. Conway, J.H.: Regular Algebra and Finite Machines. Chapman and Hall, Boca Raton (1971)
7. Friedberg, S.H., Insel, A.J., Spence, L.E.: Linear Algebra. Prentice-Hall, Englewood Cliffs (1989)
8. Gondran, M., Minoux, M.: Graphs, Dioids and Semirings: New Models and Algorithms. Springer, Heidelberg (2008)
9. Karnaugh, M.: The map method for synthesis of combinational logic circuits. Transactions of American Institute of Electrical Engineers 72(9), 593–599 (1953)
10. Kuich, W., Salomaa, A.: Semirings, Automata, Languages. Springer, Heidelberg (1986)
11. Lynch, N., Segala, R., Vaandrager, F.: Hybrid I/O automata. Information and Computation 185(1), 103–157 (2003)
12. Salomaa, A.: Theory of Automata. Pergamon Press, Oxford (1969)
13. Stockmeyer, L.J.: The set basis problem is NP-complete. Technical Report RC-5431, IBM (1975)

Optimal Boundary Control of Convention-Reaction Transport Systems with Binary Control Functions

Falk M. Hante and Günter Leugering

Department Mathematik, Lehrstuhl für Angewandte Mathematik II, Universität Erlangen-Nürnberg, Martensstr. 3, 91058 Erlangen, Germany
{hante,leugering}@am.uni-erlangen.de

Abstract. We investigate a new approach for solving boundary control problems for dynamical systems that are governed by transport equations, when the control function is restricted to binary values. We consider these problems as hybrid dynamical systems embedded with partial differential equations and present an optimality condition based on sensitivity analysis for the objective when the dynamics are governed by semilinear convection-reaction equations. These results make the hybrid problem accessible for continuous non-linear optimization techniques. For the computation of optimal solution approximations, we propose using meshfree solvers to overcome essential difficulties with numerical dissipation for these distributed hybrid systems. We compare results obtained by the proposed method with solutions taken from a mixed inter programming formulation of the control problem.

1 Introduction

Dynamical transport processes governed by first order hyperbolic *partial differential equations* (PDEs), in particular on metric graphs, model a wide variety of complex problems in civil engineering such as gas or traffic flow, but also many problems in chemical engineering as well as communication, information and logistic areas [14]. Often these problems involve decisions for controlling these dynamical processes at the boundaries, for instance turning on/off compressors, switching valves or toggle traffic lights [10].

We consider these multiscale problems as hybrid dynamical systems embedded with PDEs in which the implementation of switching is merely on a faster time scale than the transportation. With very few exceptions, noting [4,13,1,12], these problems have not been considered in the context of hybrid systems, though they represent a potentially rich field of study [3].

In context of PDE constraint optimization, mixed integer programming is used for solving such control problems, e. g. for gas network optimization [15,11], not least because of their obvious capability to consider the decision variables. The drawback of mixed integer models is certainly their computational complexity when the problems become large. Continuous non-linear optimization techniques

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 209–222, 2009.

provide an alternative, though the treatment of discrete variables therein is not straightforward. Relaxation of the decision variables together with a penalty term homotopy provides a heuristic approach for solving such problems with non-linear optimization. It has for instance been applied for ramp metering of traffic flow [4], but in general this heuristic lacks convergence to the integer optimal solution [17]. We therefore investigate alternative methods for solving this hybrid control problem using continuous non-linear optimization that converge to (locally) optimal solutions.

Similar approaches are well-known for certain lumped parameter models that usually consist of switching among *ordinary differential equations* (ODEs) in a predefined sequence of active subsystems. Optimality conditions for piecewise defined solutions of ODEs were already developed in the 1960s in the context of ODE optimal control theory [7]. These optimality conditions were later considered for optimal control of hybrid dynamical systems governed by ODEs in [2,8].

The approach investigated here is based on a new optimality condition for switching boundary data when the system dynamics are governed by the semilinear transport partial differential equation. For the computation of optimal switching signals we propose to use meshfree solvers in order to overcome essential difficulties with numerical dissipation when the distributed system is discretized in space using standard fixed Eulerian grids. To demonstrate the feasibility of our approach, we compare numerical results of our method with solutions obtained from a mixed integer programming formulation of this hybrid optimal control problem.

The paper is organized as follows. In Section 2, we give a detailed formulation of the problem we consider. In Section 3, we present a first order optimality condition based on sensitivity analysis. In Section 4, we sketch the main ideas of an appropriate numerical method to compute optimal solution approximations. In Section 5, we present numerical results for two model problems. In Section 6, we conclude with final remarks and directions for future work.

2 Problem Formulation

Consider material flow governed by the well-known convection-reaction transport equation

$$\frac{\partial}{\partial t}u(t,s) + \frac{\partial}{\partial s}[a(t,s)u(t,s)] = f(t,s,u(t,s)), \quad s \in [0,1],\ t > 0 \tag{1}$$

for the unknown scalar function $u(t,s)$. Assuming that $a > 0$, the material inflow at $s = 0$ shall be given by boundary data

$$u(t,0) = \hat{u}(t;\mu(t)), \quad t \geq 0, \tag{2}$$

where the inflow $\hat{u}$ is controlled by a parameter $\mu(t)$. The material distribution at $t = 0$ shall be given by initial data

$$u(0,s) = \bar{u}(s), \quad 0 < s < 1. \tag{3}$$

We introduce hybridness into the problem in that the control $\mu(\cdot)$ of the material inflow $\hat{u}$ is a decision variable taking values in a discrete set. We will here assume for simplicity $\mu(t) \in \{0,1\}$, $t \geq 0$. An admissible control is a switching signal $\mu(\cdot)$ which has only finitely many switches $\mu \curvearrowright \mu'$ $(\mu \neq \mu' \in \{0,1\})$ with corresponding switching time τ_k in each finite time interval.

Embedding (1)–(3) into a graph setting, these equations model a variety of realistic network flow problems. For a given graph (E, N) with edges $e_i \in E$ and nodes $n_j \in N$, one can identify each edge with an interval $[0,1]$ and consider a PDE (1) along each of those edges. At multiple nodes n_j the boundary condition (2) is then to be replaced by a nodal condition, e. g. the sum of all in- and outflows equals a given nodal control $\hat{u}(t;\mu(t))$. See, e. g. [18] for details on such a network flow model applied to air traffic flow.

As performance index of the system over a finite time horizon $[0,T]$, we consider the integral of any continuous functional $g(\cdot)[\cdot,\cdot]$, e. g.

$$\int_0^T \int_0^1 g(u)[t,s]ds\,dt = \int_0^T \int_0^1 |u(t,s) - u_d(t,s)|^2 ds\,dt, \tag{4}$$

measuring the L^2-distance of the solution u to a desired solution u_d, together with costs $\gamma(\tau_k)$ for switching $\mu \curvearrowright \mu'$ at τ_k. As the optimal boundary control of the system (1)–(3) with continuous variables is well-understood, we consider here the discrete control μ as the only control variable. Thus the control task is to minimize

$$J = \int_0^T \int_0^1 g(u)[t,s]ds\,dt + \sum_{\tau_k} \gamma(\tau_k). \tag{5}$$

by specifying the switching function $\mu(\cdot)$ on $[0,T]$, where $u(\cdot,\cdot)$ solves the continuous transport equation (1)–(3).

It is easy to see that (1) together with possibly discontinuous boundary data (2) does not possess a classical, continuously differentiable solution. As common for conservation laws, we will therefore consider solutions in a broad sense having bounded variation as given by the method of characteristics. For any point $(\tau,\sigma) \in \Omega := \{(t,s) : t \geq 0, 0 \leq s \leq 1\}$, we denote by $t \mapsto s(t;\tau,\sigma)$ the characteristic curve passing through (τ,σ), i. e. the solution of the ODE initial value problem

$$\frac{d}{dt}s(t) = a(t,s(t)), \quad s(\tau) = \sigma. \tag{6}$$

If $s(t)$ solves (6) at any time t, one has

$$\frac{d}{dt}u(t,s(t)) = \frac{\partial}{\partial t}u + \frac{d}{dt}s(t)\,\frac{\partial}{\partial s}u = \frac{\partial}{\partial t}u + a(t,s)\,\frac{\partial}{\partial s}u = \tilde{f}(t,s(t),u), \tag{7}$$

where

$$\tilde{f}(t,s,u) = f(t,s,u) - \frac{\partial}{\partial s}a(t,s). \tag{8}$$

The value of the *broad solution* u of (1)–(3) at any point $(\tau,\sigma) \in \Omega$ is then defined [5] as the value at time τ of the ODE initial value problem

$$\frac{d}{dt}u = \tilde{f}(t,s(t;\tau,\sigma),u), \quad u(t^*) = data \tag{9}$$

where t^* denotes the time when the curve $s(\cdot;\tau,\sigma)$ intersects the boundary of Ω and *data* is the prescribed initial/boundary data there.

We consider this hybrid control problem under the following hypotheses:

(H_1) The initial data $\bar{u}(\cdot)$ and, for all modes μ fixed, the boundary data $\hat{u}(\cdot;\mu)$ is continuous.

(H_2) The convection term $a(\cdot,\cdot)$ is continuous in t and twice continuously differentiable in s, positive, bounded and bounded away from 0. Moreover, $a(\cdot,\cdot)$ satisfies a bound of the form $|a(t,s)| \leq C_1(1+|s|)$ uniformely in t.

(H_3) The reaction term $f(\cdot,\cdot,\cdot)$ is continuous in t,s and is Lipschitz continuous in u. Additionally, $f(\cdot,\cdot,\cdot)$ satisfies a bound of the form $|f(t,s,u)| \leq C_2(1+|u|)$ uniformely in t and s.

(H_4) The functional $g(\cdot)$ is continuous in u.

(H_5) The switching cost $\gamma(\cdot)$ is continuously differentiable, positive and bounded below by a constant $\underline{\gamma}$.

(H_6) For some reference control $\bar{\mu}$, the cost J is finite.

For details on wellposedness of the problem, in particular in the case of a network setting, we refer to [12]. We just note that standard results in the theory of ODEs imply that the solutions (in the sense of Carathéodory) of (6) and (9) exist, are bounded for bounded initial/boundary data and depend on the point (τ,σ) in a continuously differentiable way. Moreover, hypothesis (H_5) and (H_6) bound the number of switches for the optimal control by

$$K = \left\lceil \frac{J(\bar{\mu})}{\underline{\gamma}} \right\rceil . \tag{10}$$

Latter can be easily seen by assuming that there exists an optimal control μ^* with more than K switches. The optimal value then satisfies $J(\mu^*) > K\underline{\gamma}$ because $\underline{\gamma}$ is a lower bound of the positive switching cost. On the other hand, from the bound (10) we have $J(\bar{\mu}) = K\underline{\gamma}$, contradicting the optimality of μ^*. A compactness argument then yields the following result.

Theorem 1 ([12]). *There exists an optimal switching signal μ^* minimizing (5) subject to (1), (2) and (3).* □

Our goal is to use gradient based optimization methods to compute (locally) optimal μ^*. The key idea is to fix $\mu(0)$ by $\mu_0 \in \{0,1\}$ and thus all subsequent modes and, using the bound K given in (10), to obtain μ^* by considering τ_k as the new (continuous) optimization variables subject to appropriate inequality constaints, i. e. μ^* is obtained solving

$$\begin{aligned} \min_{0\leq\tau_1\leq\cdots\leq\tau_K\leq T} \; & J[u,\tau_1,\ldots,\tau_K] \\ \text{s. t. } & u \text{ solves (1), (2), (3).} \end{aligned} \tag{11}$$

Note that in problem (11) the continuous control $\hat{u}(t,\mu_k)$ for fixed μ_k on the interswitching intervals $[\tau_k,\tau_{k+1}]$ is not subject to optimization, but the switching times τ_k are. We present a first order optimality condition for this problem in the next section, noting that this is an essential subproblem of the two stage problem involving in addition the optimization of $\hat{u}(t,\mu_k)$.

3 Optimality Condition

The optimality condition for optimal τ_k in (11) is mainly based on the following sensitivity result.

Theorem 2. *Consider the problem (11) under the hypotheses (H_1)–(H_6) and let $0<\tau_1<\cdots<\tau_K<T$. Then, for all $k=1,\dots,K$,*

$$\frac{\partial}{\partial\tau_k}J=\int_{\tau_k}^{t^*(\tau_k)}\big(g(u^{\tau_k+})[t,s^*(t,\tau_k)]-g(u^{\tau_k-})[t,s^*(t,\tau_k)]\big)\,\frac{\partial}{\partial\tau_k}s^*(t,\tau_k)\,dt+\frac{d}{d\tau_k}\gamma(\tau_k),$$

where $s^*(\cdot,\tau_k)$ solves the characteristic equation (6) with $\tau=\tau_k$ and $\sigma=0$, $t^*(\tau_k)=\max\{t\in[0,T]: s^*(t,\tau_k)\le 1\}$ and where u^{τ_k+}, u^{τ_k-} denote the solutions of (1), (2), (3) with $u(\tau_k,0)=\hat{u}(\tau_k;\mu(\tau_k+))$, $u(\tau_k,0)=\hat{u}(\tau_k;\mu(\tau_k-))$, respectively.

Proof. Fix $k\in\{1,\dots,K\}$ and let $\tau_1,\dots,\tau_K$, $s^*(\cdot,\tau_k)$ and $t^*(\tau_k)$ be given as stated in Theorem 2. The cost function can be split up as follows

$$\begin{aligned}J=&\sum_{k=1}^{K}\gamma(\tau_k)+\int_0^{\tau_k}\int_0^1 g(u^{\tau_k-})[t,s]\,ds\,dt+\int_{t^*(\tau_k)}^{T}\int_0^1 g(u^{\tau_k+})[t,s]\,ds\,dt+\\&+\int_{\tau_k}^{t^*(\tau_k)}\int_0^{s^*(t,\tau_k)} g(u^{\tau_k+})[t,s]\,ds\,dt+\int_{\tau_k}^{t^*(\tau_k)}\int_{s^*(t,\tau_k)}^1 g(u^{\tau_k-})[t,s]\,ds\,dt.\end{aligned}$$

Thus, under hypotheses (H_1)–(H_6), we have

$$\begin{aligned}\frac{\partial}{\partial\tau_k}J=&\frac{d}{d\tau_k}\gamma(\tau_k)+\int_0^1 g(u^{\tau_k-})[\tau_k,s]\,ds-\int_0^1 g(u^{\tau_k+})[t^*(\tau_k),s]\,ds\frac{\partial}{\partial\tau_k}t^*(\tau_k)+\\&+\int_{\tau_k}^{t^*(\tau_k)} g(u^{\tau_k+})[t,s^*(t,\tau_k)]\,\frac{\partial}{\partial\tau_k}s^*(t,\tau_k)\,dt+\\&-\int_0^{s^*(\tau_k,\tau_k)} g(u^{\tau_k+})[\tau_k,s]\,ds+\int_0^{s^*(t^*(\tau_k),\tau_k)} g(u^{\tau_k+})[t^*(\tau_k),s]\,ds\,\frac{\partial}{\partial\tau_k}t^*(\tau_k)+\\&-\int_{\tau_k}^{t^*(\tau_k)} g(u^{\tau_k-})[t,s^*(t,\tau_k)]\,\frac{\partial}{\partial\tau_k}s^*(t,\tau_k)\,dt+\\&-\int_{s^*(\tau_k,\tau_k)}^1 g(u^{\tau_k-})[\tau_k,s]\,ds+\int_{s^*(t^*(\tau_k),\tau_k)}^1 g(u^{\tau_k-})[t^*(\tau_k),s]\,ds\,\frac{\partial}{\partial\tau_k}t^*(\tau_k)\\=&\int_{\tau_k}^{t^*(\tau_k)}\big(g(u^{\tau_k+})[t,s^*(t,\tau_k)]-g(u^{\tau_k-})[t,s^*(t,\tau_k)]\big)\,\frac{\partial}{\partial\tau_k}s^*(t,\tau_k)\,dt+\frac{d}{d\tau_k}\gamma(\tau_k),\end{aligned}$$

where the sum of all integrals in s vanish, using that $s^*(\tau_k, \tau_k) = 0$ and that $s^*(t^*(\tau_k), \tau_k) = 1$. □

Remark 1. From the equation for $\frac{\partial}{\partial \tau_k} J$ in Theorem 2, it is easy to see that the mapping $\tau \mapsto \sum_{k=1}^{K} \frac{\partial}{\partial \tau_k} J$ is continuous for all switching times τ_k satisfying $0 < \tau_1 < \cdots < \tau_K < T$ using (H_4) and that $\tau_k \mapsto \frac{\partial}{\partial \tau_k} s^*(t, \tau_k)$ is continuous due to (H_2). But in the case that $\tau_k = \tau_{k+1}$ for some k, the derivative $\frac{\partial}{\partial \tau_k} J$ is not defined. However, the mapping $\tau_k \mapsto \frac{\partial}{\partial \tau_k} J$ can be continuated in such points continuously by resticting the domain of $g(u^{\tau_k -})[t, s^*(t, \cdot)]$ to the single point $\tau_k = \tau_{k+1}$.

Based on Theorem 2 and Remark 1 the Karush-Kuhn-Tucker first order necessary optimality conditions taking into account the special structure of the constraints $0 \leq \tau_1 \leq \cdots \leq \tau_K \leq T$ can be stated as follows.

Proposition 1. *Let $\tau^* = (\tau_1^*, \ldots, \tau_K^*)$ be a local minimum of (11) under the hypotheses (H_1)–(H_6). Then, for all $k = 1, \ldots, K$,*

$$\sum_{i=\kappa(k)}^{k} \frac{\partial J(\tau^*)}{\partial \tau_i^*} \leq 0 \text{ unless } \tau_k^* = 0, \quad \text{and} \quad \sum_{i=k}^{\eta(k)} \frac{\partial J(\tau^*)}{\partial \tau_i^*} \geq 0 \text{ unless } \tau_k^* = T, \quad (12)$$

where $\kappa(k) = \min\{0 \leq \kappa \leq k : \tau_\kappa^* = \tau_k^*\}$, $\eta(k) = \max\{K + 1 \geq \eta \geq k : \tau_\eta^* = \tau_k^*\}$ with $\tau_0^* := 0$ and $\tau_{K+1}^* := T$.

Proof. See [8]. □

4 Computational Remarks

A major difficulty of this problem comes with the fact that, in order to evaluate the cost function J, one needs to discretize and solve the PDE constraint in space and time. In order to apply non-linear optimization techniques, it is necessary to ensure that the discretized solution $\tilde{u}(\cdot, \cdot)$ depends continuously on the optimization variables $\tau_1, \ldots, \tau_K$. Careless re-meshing in every step of the optimizer may easily destroy this property. For the problem (11) continuous dependence of the mapping $(\tau_1, \ldots, \tau_K) \mapsto \tilde{u}(\cdot, \cdot)$ can be achieved by using adaptive time steps Δt. However, for time steps Δt much smaller than the discretization step size h of any fixed Eulerian grid in space, the numerical dissipation, e. g.

$$\frac{1}{2}(a(t, s)\Delta t - h)a(t, s)\frac{\partial^2}{\partial s^2} u(t, s) \quad (13)$$

for upwind finite differencing discretization schemes, becomes large and causes inaccurate solution approximations.

We overcome this difficulty by using meshfree numerical solvers for such a hybrid transport problem. Points representing the solution are moved according to their characteristic velocity. These schemes are capable of propagating

discontinuities in the solution with correct speed and they are free of numerical dissipation. In case of a semilinear equation (1), the method is easy to implement but rarely used. We will briefly sketch the method here, noting that similar particle management for the case of non-linear conservation laws has been proposed recently [9].

Sketch of the meshfree solver

1. The initial solution is the approximation of the initial data (3) by a finite number of points $s_1 < \cdots < s_m \in (0,1)$ with function values $u_1, \ldots, u_m$ for some $m \in \mathbb{N}$.
2. The solution over time is found by
   (a) Moving each point s_i with speed $a(t,s)$ as suggested by (6).
   (b) Updating the function values u_i by solving an integral formulation of $\dot{u} = f(t,s,u)$, compare (7).
   (c) Inserting points where the distance between two points or their distance to $s = 0$ becomes unsatisfyingly large. When points are inserted at $s = 0$, their function value is taken from an approximation of the boundary data (2).
   (d) Dropping all points that are no longer needed, i. e. those with $s_i > 1$.

□

Many efficient adaptive sampling strategies for the initial and boundary data can be used because there is no requirement on the point distribution s_i. In particular, one may approximate the boundary data $\hat{u}$ at the switching times τ_k and at a fixed number of equidistant time instances during interswitching intervals $[\tau_k, \tau_{k+1}]$. This strategy ensures that the discretized solution depends continuously on the switching times as desired. The method is as accurate as the movement of s_i and the updates of u_i are realized. In particular, using explicit Euler methods makes the piecewise constant solution approximation $\tilde{u}(\cdot,\cdot)$ first order accurate everywhere and the solver in pseudo-code reads as follows.

Algorithm 1 (Meshfree solver)

Require: a, $\tilde{f}$, $\bar{u}$, $\tilde{u}$, $\tau_1, \ldots, \tau_K$.
Initialize: $\tau_0 := 0$, $\tau_{K+1} := 1$, $\Delta h := \frac{1}{m}$
  $[s] := [s_1, \ldots, s_m]$ *with* $s_i = i * \Delta h$
  $[u] := [u_1, \ldots, u_m]$ *with* $u_i = \bar{u}(s_i)$
for $k = 0, \ldots, K+1$ ***do***
  $\Delta t := (\tau_{k+1} - \tau_k)/N$
  for $j = 1, \ldots, N$ ***do***
    $t := \tau_k + j * \Delta t$
    Memorize: $\tilde{u}(t, [s]) := [u]$
    Move: $[s] := [s] + \Delta t * a(t, [s])$
    Update: $[u] := [u] + \Delta t * \tilde{f}(t, [s], [u])$
    for all i *such that* $s_{i+1} - s_i > \Delta h$ ***do***
      Insert: $[s] := [[s]_{\le i}, \frac{s_i + s_{i+1}}{2}, [s]_{\ge i+1}]$, $[u] := [[u]_{\le i}, \frac{u_i + u_{i+1}}{2}, [u]_{\ge i+1}]$
    end for

if $s_1 > \Delta h$ ***then***
Insert: $[s] := [0, [s]]$, $[u] := [\hat{u}(t; \mu(t)), [u]]$
end if
Drop: $[s] := [s]_{I \setminus J}$ *with* $I = \{1, \ldots, length([s])\}$, $J = \{i \in I : s_i > 1\}$
end for
end for

With the solution approximation $\tilde{u}(\cdot, \cdot)$ obtained from the meshfree solver not all information is readily available to evaluate $\frac{\partial}{\partial \tau_k} J$ as given in Theorem 2 for applying a gradient based optimization method solving the optimality system given in Proposition 1. Additionally, the solutions of $s^*(t, \tau_k)$ and $\frac{\partial}{\partial \tau_k} s^*(t, \tau_k)$ are needed. While the former can be directly computed using (6), the latter can be obtained using the following Lemma.

Lemma 1. *Let* $s^*(t, \tau_k)$ *solve the characteristic equation* (6) *with* $\tau = \tau_k$ *and* $\sigma = 0$. *Then,* $\frac{\partial}{\partial \tau_k} s^*(t, \tau_k) = \Phi(t, \tau_k)$, *where* $\Phi(\theta, \tau_k)$ *is the state transition matrix of the following linear time varying dynamical system*

$$\frac{d}{d\theta} z(\theta) = \frac{\partial}{\partial s} a(\theta, s)\big|_{s=s^*(\theta,\tau_k)} z(\theta) \tag{14}$$

Proof. Let $z(\theta) = \frac{\partial}{\partial \tau_k} s^*(\theta, \tau_k)$. Then, the statement of the Lemma follows from the derivation

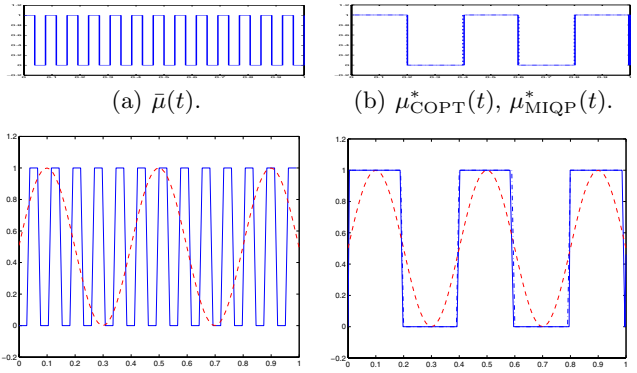
$$\begin{aligned} \frac{d}{d\theta} z(\theta) &= \frac{d}{d\theta} \frac{\partial}{\partial \tau_k} s^*(\theta, \tau_k) = \frac{\partial}{\partial \tau_k} \frac{d}{d\theta} s^*(\theta, \tau_k) = \frac{\partial}{\partial \tau_k} a(\theta, s^*(\theta, \tau_k)) \\ &= \frac{\partial}{\partial s} a(\theta, s)\big|_{s=s^*(\theta,\tau_k)} \frac{\partial}{\partial \tau_k} s^*(\theta, \tau_k) = \frac{\partial}{\partial s} a(\theta, s)\big|_{s=s^*(\theta,\tau_k)} z(\theta). \end{aligned}$$ □

In the following section, we compare numerical results for a gradient based optimization method established on the results presented in this section.

5 Numerical Results

We present numerical results for two model problems. For both we compare the following two methods to compute approximations of (locally) optimal binary control functions $\mu^*(\cdot)$ that minimize (5).

COPT. This methods applies continuous non-linear optimization techniques for the reformulated problem (11) using the results presented in Section 3 and Section 4. The system (1) is solved using a meshfree solver, which realizes the movement of s_i and the updates of u_i in our implementation by explicit Euler methods, see Algorithm 1. The cost function is approximated by the trapezoidal rule. The search for locally optimal τ_k (specifying $\mu(\cdot)$) is carried out by the MATLAB sequential quadratic programming solver *fmincon* [19]. The gradients for the BFGS updates are computed using the formula given in Theorem 2. Termination criterion is the first order optimality measure or the norm of the directional derivative falling below tolerance.



(a) $\bar{\mu}(t)$. (b) $\mu^*_{\mathrm{COPT}}(t)$, $\mu^*_{\mathrm{MIQP}}(t)$.

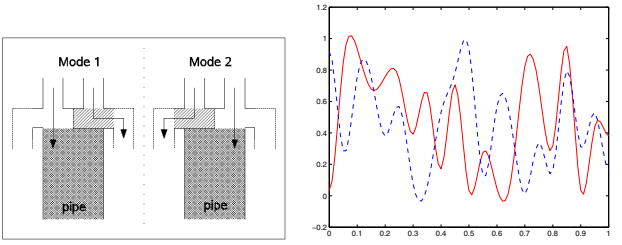
(c) $\bar{u}_{\mathrm{COPT}}(1,s)$. (d) $u^*_{\mathrm{COPT}}(1,s)$, $u^*_{\mathrm{MIQP}}(1,s)$.

Fig. 1. Initial and optimal bang-bang type solution of the traveling sine wave in Example 1. Fig. (a) and (c) show the initial switching signal with a final time plot of the corresponding solution. Fig. (b) and (d) show the computed optimal switching signals with corresponding solutions at $t = T = 1$ (COPT solutions solid, MIQP solutions dash-dotted). The dashed curve in Fig. (c) and (d) show the desired wave u_d plotted at final time $T = 1$.

MIQP. This method uses mixed integer programming on the original problem. The system dynamics (1) are transformed into a linear system constructed by upwind finite difference discretization on a fixed, equidistant Eulerian grid. For each timestep t_k, a binary variable represents $\mu(t_k)$. The cost function is approximated by the trapezoidal rule. We included the details on the MIQP reformulation in Appendix A, noting that more sophisticated MIQP reformulations of this problem are possible. The reformulation choosen here shall primarily serve as a verification of the proposed method above. The search for the obtained equality constraint mixed integer quadratic program is carried out by ILOG CPLEX [6]. The solver terminates when the gap between the best integer objective and the objective of the best node remaining in the branch-and-bound tree falls below tolerance.

The first very simple example serves as a verification of the proposed method COPT for computing approximations of optimal switching control functions.

Example 1. (Bang-bang type approximation of a traveling sine wave.) The control task consists of approximating a traveling sine wave u_d by switching $\hat{u}$ between the two extremal values of the wave $\hat{u}_1 = 0$, $\hat{u}_2 = 1$. We assume that the wave speed equals the transportation velocity, here taken for simplicity as $a(t,s) = 1$. We also assume constant switching costs $\gamma(\cdot) = 0.0075$ to avoid chattering. It should be clear that for this problem, we cannot expect exact controllability, but we are seeking for a binary control $\mu^*(\cdot)$ minimizing the L^2-distance (4) between u and u_d over the finite time horizon $[0,T]$ with $T = 1$. Also observe that the optimal control of the relaxed problem with $\hat{u} \in [0,1]$ is not



(a) Two mode inlet control. (b) Inflow data in mode 1 (solid) and mode 2 (dashed).

Fig. 2. Control task in Example 2

of bang-bang type. Thus, alternative methods like COPT or MIQP are required for the computation of optimal controls. For initialization of the continuous optimizer, we use $\bar{\mu}(\cdot)$ given by $\bar{\tau}_1, \ldots, \bar{\tau}_K$ with $K = 25$ equidistantly placed in $[0, T]$ as depicted in Figure 1 (a). The cost of the initial solution that is shown in Figure 1 (c) is $J(\bar{\mu}) = 0.1869$.

COPT terminates after 11 iterations with an optimal value of $J(\mu^*_{\mathrm{COPT}}) = 0.0299$ and first order optimality measure of 0.0092, where 20 of the 25 initial switching times coalesce in the optimal solution approximation. The integer optimal solution μ^*_{MIQP} for the upwind-discretized problem obtained by MIQP on a fixed grid with 2500 discretization points is qualitatively the same with $J(\mu^*_{\mathrm{MIQP}}) = 0.0298$.

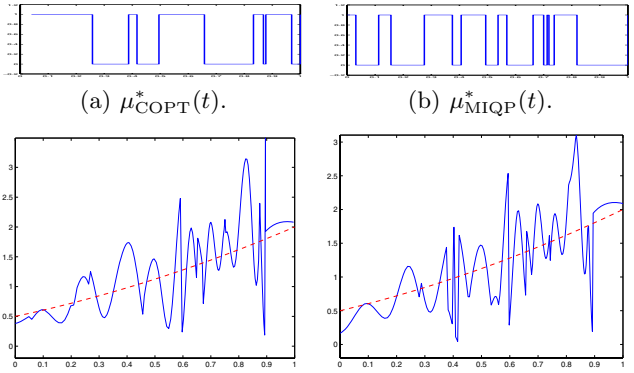
Plots of μ^*_{COPT} (solid line) and μ^*_{MIQP} (dashed line) are shown in Figure 1 (b) while the corresponding final time plots of the solution u at $t = T = 1$ are shown in Figure 1 (d). Note that the example is chosen such that at time T the complete history of the boundary control action $\mu(\cdot)$ is visible in $u(T, s)$.

The second example demonstrates that the method COPT may even outperform our mixed integer optimal programming implementation due to its inferiority in the discretization of the dynamical system.

Example 2. (2-mode plug flow regulation.) Consider a pipe that can be controlled at the inlet by choosing the inflow of material concentration either from $\hat{u}_1$ or from $\hat{u}_2$, compare Figure 2 (a). The plug flow in the pipe is assumed to satisfy the conservation law

$$\frac{\partial}{\partial t} u(t, s) + \frac{\partial}{\partial s}[a(s)u(t, s)] = 0 \tag{15}$$

with $a(s) = \frac{4}{3}(s-1)^2 + \frac{1}{2}$. The desired material distribution in the pipe is given by $u_d(t, s) = \frac{1}{2}(s+1)^2$. As in Example 1, we cannot expect exact controllability, but we are again seeking for a binary control $\mu^*(\cdot)$ minimizing the L^2-distance (4) between u and u_d over the finite time horizon $[0, 1]$ and include switching costs $\gamma(\cdot) = 0.0075$ to avoid chattering. For initialization of the continuous optimizer, we use $\bar{\mu}(\cdot)$ with 35 equidistantly placed switching times τ_k with a corresponding cost of $J(\bar{\mu}) = 0.1693$.



(a) $\mu^*_{\mathrm{COPT}}(t)$. (b) $\mu^*_{\mathrm{MIQP}}(t)$.

(c) $u^*_{\mathrm{COPT}}(1,s)$. (d) $u^*_{\mathrm{MIQP}}(1,s)$.

Fig. 3. Optimal control approximations for Example 2. Fig. (a) and (c) show the optimal switching signal computed with COPT and the final time plot of the corresponding solution u with $J(\mu^*_{\mathrm{COPT}}) = 0.0914$. Fig. (b) and (d) show the MIQP result with $J(\mu^*_{\mathrm{MIQP}}) = 0.2694$. The dashed line in Fig. (c) and (d) is the desired material distribution u_d.

COPT terminates after 17 iterations with an optimal value of $J(\mu^*_{\mathrm{COPT}}) = 0.0914$ and first order optimality measure of 0.01. The integer optimal solution μ^*_{MIQP} for the upwind-discretized problem obtained by MIQP on a fixed grid with 12800 discretization points has on optimal value of $J(\mu^*_{\mathrm{MIQP}}) = 0.2694$. The optimal control approximations μ^*_{COPT} and μ^*_{MIQP} with corresponding final time plots of the solutions at $t = T = 1$ are depicted in Figure 3.

We finally remark that the choice of the initial condition $\bar{\mu}(\cdot)$ is crucial for the proposed method since it searches for locally optimal controls only. A direct comparison with mixed integer programming, searching for globally optimal solutions on the discretized problem but at exponential cost, therefore is not feasible.

6 Conclusion

We presented a new approach for solving optimal boundary control problems for dynamical systems that are governed by semilinear transport equations when the control function is restricted to binary values. By considering this problem as a hybrid dynamical system embedded with partial differential equations, we derived an optimality condition similar to results known for hybrid systems governed by ordinary differential equations. This result makes the problem accessible for gradient based non-linear optimization methods.

For the computation of optimal solution approximations, we used meshfree solvers to overcome essential difficulties with numerical dissipation for these distributed hybrid systems. Our numerical results for model problems show that the proposed approach is a promising alternative compared to mixed integer programming.

Future work will be devoted to extend this approach to control problems that are governed by non-linear transport equations and multi-dimensional systems of equations.

Acknowledgments

This work has been supported by the Elite Network of Bavaria within the project #K-NW-2004-143.

References

1. Amin, S., Hante, F.M., Bayen, A.M.: On stability of switched linear hyperbolic conservation laws with reflecting boundaries. In: Egerstedt, M., Mishra, B. (eds.) HSCC 2008. LNCS, vol. 4981, pp. 602–605. Springer, Heidelberg (2008)
2. Xu, X., Antsaklis, P.: Optimal control of switched autonomous systems. In: Proc. IEEE Conf. Decision and Control, Las Vegas, NV (December 2002)
3. Barton, P.I.: Modeling, Simulation and Sensitivity Analysis of Hybrid Systems. In: Proc. of the IEEE Int. Symposium on Computer-Aided Control System Design, Anchorage, Alaska, September 25–27 (2000)
4. Bayen, A.M., Raffard, R.L., Tomlin, C.J.: Network congestion alleviation using adjoint hybrid control: Application to highways. In: Alur, R., Pappas, G.J. (eds.) HSCC 2004. LNCS, vol. 2993, pp. 95–110. Springer, Heidelberg (2004)
5. Bressan, A.: Hyperbolic Systems of Conservation Laws. Oxford University Press, New York (2000)
6. ILOG, Inc.: ILOG CPLEX Version 9.1, Sunnyvale, CA, USA (2007)
7. Dyer, P., McReynolds, S.R.: The Computation and Theorie of Optimal Control. Series Mathematics in Science and Engineering, vol. 65. Academic Press, New York (1970)
8. Egerstedt, M., Wardi, Y., Axelsson, H.: Transition-Time Optimization for Switched-Mode Dynamical Systems. IEEE Transactions on Automatic Control 51(1), 110–115 (2006)
9. Farjoun, Y., Seibold, B.: Solving One Dimensional Scalar Conservation Laws by Particle Management (January 2008) arXiv:0801.1495 [math.NA]
10. Fügenschuh, A., Herty, M., Klar, A., Martin, A.: Combinatorial and Continuous Models for the Optimization of Traffic Flows on Networks. SIAM Journal on Optimization 16, 1155–1176 (2006)
11. Geißler, B., Kolb, O., Lang, J., Leugering, G., Martin, A., Morsi, A.: Mixed Integer Linear Models for the Optimization of Dynamical Transport Networks (submitted, 2008)
12. Hante, F.M., Leugering, G., Seidman, T.I.: Modeling and Analysis of Modal Switching in Networked Transport Systems. Applied Mathematics and Optimization (in print) (2008) DOI:10.1007/s00245-008-9057-6
13. Kleinert, T., Lunze, J.: Modelling and state observation of Simulated Moving Bed processes based on an explicit functional wave form description. Mathematics and Computers in Simulation 68(3), 235–270 (2005)
14. Leugering, G.: Optimization and control of transport processes on networked systems. In: Conf. on Control of Physical Systems and Partial Differential Equations, Paris, June 16–20 (2008)

15. Martin, A., Möller, M., Moritz, S.: Mixed Integer Models for the Stationary Case of Gas Network Optimization. Math. Program., Ser. B 105, 563–582 (2006)
16. Quarteroni, A., Valli, A.: Numerical Approximation of Partial Differential Equations. 2. corr. printing. Springer, Berlin (1997)
17. Sager, S., Bock, H.G., Diehl, M., Reinelt, G., Schloder, J.P.: Numerical Methods for Optimal Control with Binary Control Functions Applied to a Lotka-Volterra Type Fishing Problem. In: Seeger, A. (ed.) Recent Advances in Optimization. LNEMS, vol. 563, pp. 269–289. Springer, Heidelberg (2006)
18. Sun, D., Strub, I., Bayen, M.: Comparison of the performance of four Eulerian network flow models for strategic air traffic flow management. Networks and Heterogeneous Media 2(4), 569–594 (2007)
19. The Mathworks, Inc.: Matlab Release 7.5.0 (R2007b), Natick, MA, USA (2007)

A Appendix: MIQP Formulation

For the sake of completeness we add the mixed integer formulation of the control problem (5) that was used for comparison in Example 1 and Example 2. In order to discretize a mixed initial boundary value problem of the type

$$\begin{aligned} &\frac{\partial}{\partial t}u(t,s) + a(t,s)\,\frac{\partial}{\partial s}u(t,s) = f(t,s)\,u(t,s) \quad s \in [0,1],\ t > 0 \\ &u(t,0) = \hat{u}(t;\mu(t)), \quad t \geq 0 \\ &u(0,s) = \bar{u}(s), \quad s \in (0,1) \end{aligned} \tag{16}$$

we use an explicit upwind finite difference scheme (see e. g. [16], Sec. 14.2), coupled with integer programming. The space-time domain is discretized by choosing a time step $\Delta t = 1/N$ and a mesh-width $\Delta s = 1/M$. The grid-points (t_j, s_i) are defined by

$$t_j = (j-1)\Delta t,\ j = 1,\dots,N, \quad s_i = (i-1)\Delta s,\ i = 1,\dots,M. \tag{17}$$

We replace the derivatives in the system (16) by upwind finite differences (using that $a(\cdot,\cdot) > 0$)

$$\frac{U_i(t_{j+1}) - U_i(t_j)}{\Delta t} + a(t_j,s_i)\frac{U_i(t_j) - U_{i-1}(t_j)}{\Delta s} = f(t_j,s_i)U_i(t_j) \tag{18}$$

for $i = 2,\dots,M$ and $j = 2,\dots,N$ and the initial condition becomes

$$U_i(t_1) = \bar{u}(s_i), \quad i = 2,\dots,M. \tag{19}$$

On the time grid the discrete control $\mu(\cdot)$ can be represented by N binary values $\mu_j \in \{0,1\}$ and thus the boundary conditions can be written as

$$U_1(t_j) = \mu_j\hat{u}(t;1) + (1-\mu_j)\hat{u}(t;0), \quad j = 1,\dots,N. \tag{20}$$

With $N\,M$ new continuous variables x_n given by

$$x_{(j-1)M+i} = U_i(t_j), \quad i = 1,\dots,M \text{ and } j = 1,\dots,N \tag{21}$$

and N additional binary variables

$$x_{NM+j} = \mu_j, \quad j = 1, \ldots, N \tag{22}$$

the equations (18), (19) and (20) can be written as a mixed integer linear equation system $Ax = b$ in $x_1, \ldots, x_{(N+1)M}$ with sparse coefficient matrix A. The integral part of the cost function (4) as used in the examples is approximated by

$$\sum_{i=1}^{NM} (x_i - z_i)^2 = (x - z)^\top (x - z) = x^\top x - 2z^\top x + z^\top z, \tag{23}$$

where z_i is a discretization of u_d. Using that $z^\top z$ is constant, these costs can be written as $x^\top Q x - c^\top x$ with $Q = 1$ and $c = 2z$. Moreover, the switching costs $\sum_{\tau_k} \gamma(\tau_k)$ for constant γ can be encoded in $Q = (q_{i,j})$ by setting $\kappa = \gamma/N$, $q_{i,i} = \kappa$, $q_{i+1,i} = -\frac{1}{2}\kappa$ and $q_{i,i+1} = -\frac{1}{2}\kappa$ for $i = 1 = NM+1, \ldots, (N+1)M$. We remark that for stability of the applied methods the above discretization scheme requires N and M choosen such that the CFL-condition holds.

Trajectory Based Verification Using Local Finite-Time Invariance*

A. Agung Julius and George J. Pappas

Department of Electrical and Systems Engineering
University of Pennsylvania
200 South 33rd Street, Philadelphia PA-19104
United States of America
{agung,pappasg}@seas.upenn.edu

Abstract. In this paper we propose a trajectory based reachability analysis by using local finite-time invariance property. Trajectory based analysis are based on the execution traces of the system or the simulation thereof. This family of methods is very appealing because of the simplicity of its execution, the possibility of having a partial verification, and its highly parallel structure.

The key idea in this paper is the construction of local barrier functions with growth bound in local domains of validity. By using this idea, we can generalize our previous method that is based on the availability of global bisimulation functions. We also propose a computational scheme for constructing the local barrier functions and their domains of validity, which is based on the S-procedure. We demonstrate that our method subsumes some other existing methods as special cases, and that for polynomial systems the computation can be implemented using sum-of-squares programming.

1 Introduction

One of the main problems in the field of hybrid systems is reachability analysis/safety verification. This type of problems is related to verifying that the state of a hybrid system does not enter a declared unsafe set in its execution trajectory. The domain of application of the problem is very wide, ranging from engineering design [1,2], air traffic management systems [3,4], to systems biology [5,6]. Understanding the importance of the problem, the hybrid systems community has put a lot of efforts in the research of reachability analysis and verification. We refer the reader to [7,8,9,10,11,12,13,14,15,16] for some of the earlier references in this topic[1].

* This work is partially funded by National Science Foundations awards CSR-EHS 0720518 and CSR-EHS 0509327.

[1] Given the breadth of research in this topic, this list is by no means exhaustive. However, it does capture a broad spectrum of techniques that have been developed in the community to answer the safety/reachability problems.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 223–236, 2009.

Among the different approaches to reachability analysis, there is a family of methods that is based on simulation or trajectory analysis. In some literature, this type of approach is also called *testing based* [17], referring to the possibility of generating the trajectories through actual executions (tests). This type of approach is very appealing because of several reasons [18]. One of the reasons is its simplicity. Running or simulating a system is generally much simpler than performing symbolic analysis on it. This is particularly true for systems with complex dynamics. Another reason why trajectory based verification is attractive is that its algorithm is highly parallelizable. Since simulation runs of the system do not depend one on another, they can be easily assigned to different processors, resulting in a highly parallel system. Trajectory based verification is also close to some actual practice in the industry where verification is done through "exhaustive" testing and/or simulation. Of course, formal exhaustive testing for continuous/hybrid systems is not possible, unless they are coupled with some notion of robustness, which is the central issue in this paper.

Within the family of trajectory based reachability analysis techniques itself there are different approaches. Some methods, for example, conduct state space exploration through randomized testing [19] or by using Rapidly exploring Random Trees (RRT) or its adaptations [20,21]. Methods based on linearization of the system's nonlinear dynamics along the execution trajectory have also been proposed, for example in [22,23]. Other related methods incorporate the notions of sensitivity [24], local gain/contraction analysis [25,26] and bisimulation function [27,17,28] to measure the difference between neighboring trajectories. The method that we present in this paper belongs to this class. In a sense, these methods combine two of the most successful analysis techniques for nonlinear dynamics, simulation and stability analysis. The difference between our approach and other approaches that use, for example, sensitivity analysis [24] and local gain/contraction analysis [25,26] is in the fact that we are not restricted to use a prespecified metric in the state space. In fact, the bisimulation/barrier functions induce a pseudometric that can be (locally) customized to best fit the application [17].

In this paper, we extend the approach reported in [17] (and in [18] for stochastic systems). An illustration of the approach proposed in this paper is shown in Figure 1. Suppose that we have a test trajectory that satisfies the safety condition. In the above mentioned references, we rely on the assumption of the availability of a bisimulation function for each mode of dynamics, which is valid globally, to bound the divergence of the trajectories resulting from nearby initial conditions. The contribution of this paper lies in the relaxation of this global assumption, allowing for more flexibility in the computation. Effectively, we construct a guarantee on the divergence of execution trajectories by piecing together multiple local finite-time invariance arguments. The idea is to link the domains of validity of these local invariance to cover a neighborhood of the test trajectory. In Figure 1 these domains are shown as Domain-1, 2, and 3. The local invariance arguments that we construct are similar to the barrier certificate as proposed in [14], except for the fact that the validity of the invariance property is finite time. In each of these domains, the shape of the level sets of the barrier function

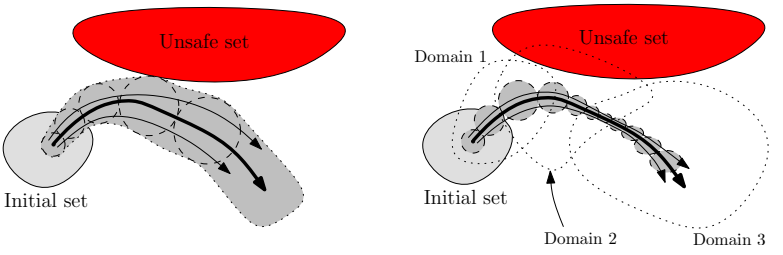


Fig. 1. The main idea presented in this paper. The test trajectory is shown as the thick curve. The use of global bisimulation function to bound the divergence of the trajectories is illustrated on the left side. Local finite-time invariance based analysis is illustrated on the right side.

and the stability property of the dynamics can be different. This is illustrated in Figure 1 by the changing of the shape and the size of the level sets.

The rest of the paper is organized as follows. In the next section, we present some basic results about local finite-time invariance of dynamical systems. The application of these results in safety verification is discussed in Section 3. In Section 4, we also propose a computational scheme to compute the barrier functions and their domains of validity based on the S-procedure [29]. We show that for affine systems, the method proposed in this paper coincides with that in [17]. We also show that our result captures, as a special case, the method based on local linearization of nonlinear systems. For polynomial systems we show that the computation can be implemented by using sum-of-squares (SOS) programming and demonstrate it with an example.

2 Local Finite-Time Invariance

We consider nonlinear dynamical system of the form

$$\dot{x} = f(x),\ x \in \mathcal{X}, \tag{1}$$

where $\mathcal{X} \subset \mathbb{R}^n$ is the state space of the system, and a differentiable function $\phi : \mathcal{X} \to \mathbb{R}_+$. We assume that the differential equation posed in (1) admits a unique solution for any initial condition in $\mathcal{X}$, during the time interval of interest, $\mathcal{T}$.

Notation. We denote the flow of the dynamical system at time t with initial condition $x(t)_{t=t_0} = x_0$ as $\xi(t; x_0, t_0)$. That is, $\xi(t; x_0, t_0)$ satisfies

$$\frac{d}{dt}\xi(t; x_0, t_0) = f(\xi(t; x_0, t_0)),$$
$$\xi(t_0; x_0, t_0) = x_0.$$

We have the natural semigroup property of the flow: $\xi(t; x_0, t_0) = \xi(t; x', t')$, where $x' := \xi(t'; x_0, t_0)$ for any $t' \in [t_0, t]$. We also have the time invariance property: $\xi(t; x_0, t_0) = \xi(t + \Delta; x_0, t_0 + \Delta)$ for any $\Delta \in \mathbb{R}$.

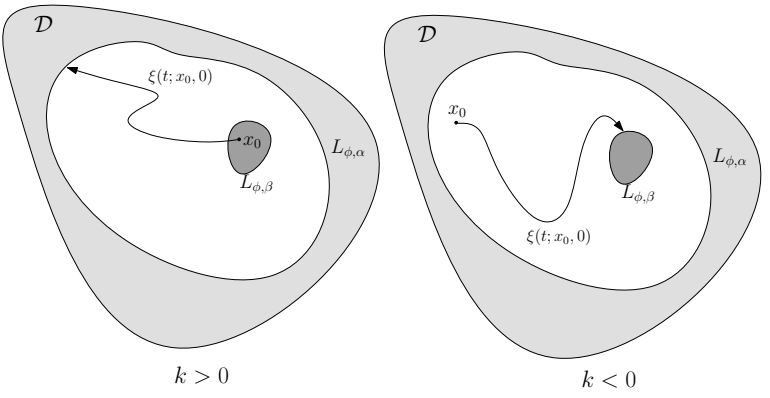


Fig. 2. Illustration for Propositions 1 and 2 for the case when $k > 0$ (left) and $k < 0$ (right)

Notation. We denote the level set of a function $\phi : \mathcal{X} \to \mathbb{R}$ as

$$L_{\phi,\alpha} := \{x \in \mathcal{X} \mid \phi(x) \leq \alpha\}. \tag{2}$$

In the subsequent discussion in this paper we need the following two results related to local finite-time invariance.

Proposition 1. *Suppose that the following relation holds for a subset $\mathcal{D} \subset \mathcal{X}$ and for some $k \in \mathbb{R}$,*

$$\nabla_x \phi(x) f(x) \leq k, \forall x \in \mathcal{D}. \tag{3}$$

Take any $\alpha, \beta \in \mathbb{R}$ such that $\beta < \alpha$ and $L_{\phi,\alpha} \subset \mathcal{D}$. The following results hold.
(i) If $k > 0$, then any trajectory of the system (1) that starts in $L_{\phi,\beta}$ remains in $L_{\phi,\alpha}$ for at least $\frac{\alpha-\beta}{k}$ time units, or mathematically

$$\xi(t;x_0,0) \in L_{\phi,\alpha},\ \forall x_0 \in L_{\phi,\beta}, \forall t \leq \frac{\alpha-\beta}{k}.$$

(ii) If $k < 0$, then any trajectory of the system (1) that starts in $L_{\phi,\alpha}$ enters $L_{\phi,\beta}$ after at most $\frac{\beta-\alpha}{k}$ time units, or

$$\xi(t;x_0,0) \in L_{\phi,\beta},\ \forall x_0 \in L_{\phi,\alpha}, \forall t \geq \frac{\beta-\alpha}{k}.$$

Proposition 2. *Suppose that the following relation holds for a subset $\mathcal{D} \subset \mathcal{X}$ and for some $k \in \mathbb{R}$,*

$$\nabla_x \phi(x) f(x) \leq k\phi(x), \forall x \in \mathcal{D}. \tag{4}$$

Take any $\alpha, \beta \in \mathbb{R}$ such that $\beta < \alpha$ and $L_{\phi,\alpha} \subset \mathcal{D}$. The following results hold.
(i) If $k > 0$, then any trajectory of the system (1) that starts in $L_{\phi,\beta}$ remains in $L_{\phi,\alpha}$ for at least $\frac{\ln\alpha-\ln\beta}{k}$ time units, or mathematically

$$\xi(t; x_0, 0) \in L_{\phi,\alpha},\ \forall x_0 \in L_{\phi,\beta}, \forall t \leq \frac{\ln \alpha - \ln \beta}{k}.$$

(ii) If $k < 0$, then any trajectory of the system (1) that starts in $L_{\phi,\alpha}$ enters $L_{\phi,\beta}$ after at most $\frac{\ln \beta - \ln \alpha}{k}$ time units, or

$$\xi(t; x_0, 0) \in L_{\phi,\beta},\ \forall x_0 \in L_{\phi,\alpha}, \forall t \geq \frac{\ln \beta - \ln \alpha}{k}.$$

Definition 1. *Hereafter, we call a function $\phi : \mathcal{X} \to \mathbb{R}$ that satisfies (3) or (4) a barrier function with constant and linear growth bound, respectively. The corresponding domain $\mathcal{D}$ is called the domain of validity of the barrier functions.*

Propositions 1 and 2 can be proved by using an argument similar to Lyapunov stability theory, which is a standard result in nonlinear system analysis (see, for example [30]). Effectively, the results above can be used to establish a barrier certificate that is valid for a finite time. Notice that if $\frac{\partial \phi}{\partial x} f(x)$ is continuous and $\mathcal{D}$ is a compact set, we can always find a finite bound k in (3). In a sense, this property guarantees that for any continuous function $f(x)$ and a compact domain $\mathcal{D}$, we can always construct a smooth barrier function with a finite constant growth bound.

3 Safety Verification

In this section, we extend the results in the previous section to the product of a dynamical system with itself. The goal is to establish a method for computing the robustness of test trajectories for systems with nonlinear dynamics. We consider dynamical systems in the form of (1), and suppose that there is an unsafe subset of the state space $\mathcal{X}$, which we denote by `Unsafe`. We want to verify that the execution trajectories of the system are *safe*. That is, they do not enter the unsafe set. The object of robustness computation is to establish a neighborhood around a test trajectory that is guaranteed to have the same safety property.

Consider a trajectory of the system with initial condition $x_i \in \mathcal{X}$ in the time interval $[0, T]$, as illustrated in Figure 3. Suppose that there exists a set $\mathcal{D} \subset \mathcal{X} \times \mathcal{X}$ such that the trajectory $(\xi(t; x_i, 0), \xi(t; x_i, 0)) \in \mathcal{D}$, and that the trajectory is safe, i.e. $\xi(t; x_i, 0) \notin \texttt{Unsafe}$, for all $t \in [0, T]$. Further, for any function $\phi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, we define a function

$$d(t) := \inf_{y \in \texttt{Avoid}(t)} \phi(\xi(t; x_i, 0), y), \forall t \in [0, T], \tag{5}$$

$$\texttt{Avoid}(t) := \{y \mid y \in \texttt{Unsafe}, (\xi(t; x_i, 0), y) \in \mathcal{D}\} \cup \{y \mid (\xi(t; x_i, 0), y) \notin \mathcal{D}\}, \tag{6}$$

and introduce the following notation.

Notation. We introduce the level set notation

$$L^x_{\phi,\alpha} := \{y \mid \phi(x, y) \leq \alpha\}. \tag{7}$$

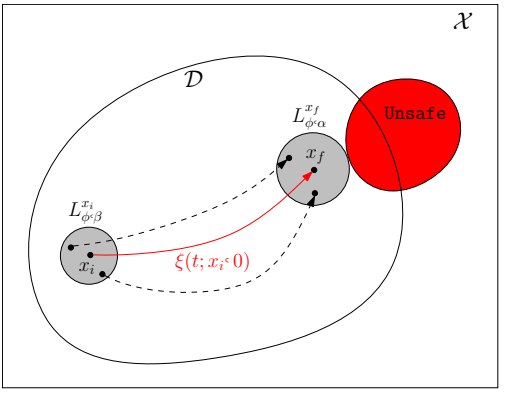


Fig. 3. Illustration for Proposition 3. The circles represent level sets of ϕ. For β that satisfies the condition in Proposition 3, any trajectory that starts in $L^{x_i}_{\phi,\beta}$ is guaranteed to possess the same safety property as $\xi(t; x_i, 0)$.

Proposition 3. *Suppose that for all $(x, y) \in \mathcal{D}$, there exists a $k \in \mathbb{R}$ such that*

$$\nabla_x \phi(x, y) f(x) + \nabla_y \phi(x, y) f(y) \leq k.$$

Let β and k' be such that

$$\beta + k' t \leq d(t), \forall t \in [0, T], \tag{8}$$
$$k' \geq k. \tag{9}$$

For any initial condition $x_0 \in L^{x_i}_{\phi,\beta}$, we have that

$$\xi(t; x_0, 0) \notin \texttt{Unsafe}, \tag{10}$$
$$(\xi(t; x_i, 0), \xi(t; x_0, 0)) \in \mathcal{D}, \tag{11}$$

for all $t \in [0, T]$.

Proof. By applying Proposition 1, we can show that for all $t \in [0, T]$,

$$\phi(\xi(t; x_i, 0), \xi(t; x_0, 0)) \leq \beta + k' t \leq d(t). \tag{12}$$

By definition, it implies that for all $t \in [0, T]$,

$$\phi(\xi(t; x_i, 0), \xi(t; x_0, 0)) \leq \inf_{y \in \texttt{Avoid}(t)} \phi(\xi(t; x_i, 0), y),$$

and therefore

$$\xi(t; x_0, 0) \notin \texttt{Avoid}(t).$$

By definition of $\texttt{Avoid}(t)$, this immediately implies the validity of (10 - 11).

A result similar to Proposition 3 for barrier functions with linear growth bound can be constructed as follows (the proof follows a similar construction).

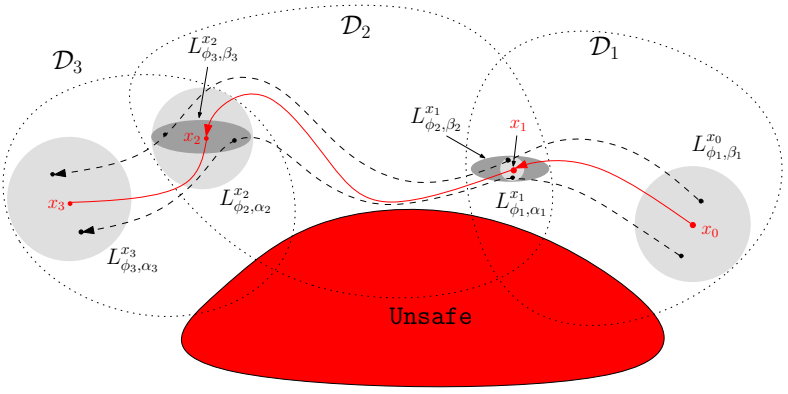


Fig. 4. Illustration of Theorem 1. The solid trajectory represents the test trajectory, while the dashed ones represent other trajectories with initial conditions in $L^{x_0}_{\phi_1,\beta_1}$.

Proposition 4. *Suppose that for all $(x, y) \in \mathcal{D}$, there exists a $k \in \mathbb{R}$ such that*

$$\nabla_x \phi(x, y) f(x) + \nabla_y \phi(x, y) f(y) \leq k\phi(x, y).$$

Let β and k' be such that

$$\ln \beta + k't \leq \ln d(t), \forall t \in [0, T], \tag{13}$$

$$k' \geq k. \tag{14}$$

For any initial condition $x_0 \in L^{x_i}_{\phi,\beta}$, we have that

$$\xi(t; x_0, 0) \notin \texttt{Unsafe}, \tag{15}$$

$$(\xi(t; x_i, 0), \xi(t; x_0, 0)) \in \mathcal{D}, \tag{16}$$

for all $t \in [0, T]$.

The results above establish a way to perform a local testing-based safety verification using a local bisimulation function/ Lyapunov function type argument, which is similar to [17]. Namely, we can guarantee the safety of all trajectories starting from a neighborhood $L^{x_i}_{\phi,\beta}$ of the nominal initial state x_i. The new contribution in this paper lies in the fact that the domain of validity of the function can be local. The locality of this analysis can be extended by linking multiple local analysis to cover a test trajectory. This idea is elucidated in the following theorem, and illustrated in Figure 4.

Theorem 1. *Consider a test trajectory $\xi(t; x_0, 0)$, $t \in [0, T]$. Suppose that for $i = 1, \ldots, N$, there exists a family of sets $\mathcal{D}_i \subset \mathcal{X} \times \mathcal{X}$, functions $\phi_i : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$, positive constants α_i and β_i, and time intervals $0 = t_0 < t_1 < \cdots < t_N = T$ such that*
(i) $(\xi(t; x_0, 0), \xi(t; x_0, 0)) \in \mathcal{D}_i$ for all $t \in [t_{i-1}, t_i]$,
(ii) for all $(x, y) \in \mathcal{D}_i$, there exists a $k_i \in \mathbb{R}$ such that

$$\nabla_x \phi_i(x, y) f(x) + \nabla_y \phi_i(x, y) f(y) \leq k_i, \tag{17}$$

(iii) there exists a $k_i' \geq k_i$ *such that*

$$\beta_i + k_i' t \leq d_i(t), \forall t \in [0, t_i - t_{i-1}],$$
$$\beta_i + k_i' (t_i - t_{i-1}) \leq \alpha_i,$$

where

$$d_i(t) := \inf_{y \in \texttt{Avoid}_i(t)} \phi_i(\xi(t; x_{i-1}, 0), y), \forall t \in [0, t_i - t_{i-1}],$$
$$\texttt{Avoid}_i(t) := \{y \mid y \in \texttt{Unsafe}, (\xi(t; x_{i-1}, 0), y) \in \mathcal{D}_i\} \cup \{y \mid (\xi(t; x_{i-1}, 0), y) \notin \mathcal{D}_i\},$$
$$x_{i-1} := x(t_{i-1}),$$

(iv) for $i = 1, \ldots, N-1$,

$$\alpha_i \leq \sup \left\{ \alpha \mid L^{x_i}_{\phi_i, \alpha} \subset L^{x_i}_{\phi_{i+1}, \beta_{i+1}} \right\}.$$

For any initial condition $\tilde{x}_0 \in L^{x_0}_{\phi_1, \beta_1}$, *we have that*

$$\xi(t; \tilde{x}_0, 0) \notin \texttt{Unsafe}, \tag{18}$$
$$(\xi(t; x_0, 0), \xi(t; \tilde{x}_0, 0)) \in \cup_{i=1}^N \mathcal{D}_i, \tag{19}$$

for all $t \in [0, T]$.

Proof. Consider the last interval of the trajectory, that is $t \in [t_{N-1}, T]$. By Proposition 3, we have that for any $\tilde{x}_{N-1} \in L^{x_{N-1}}_{\phi_N, \beta_N}$,

$$\xi(t; \tilde{x}_{N-1}, t_{N-1}) \notin \texttt{Unsafe}, \tag{20}$$
$$(\xi(t; x_{N-1}, 0), \xi(t; \tilde{x}_{N-1}, 0)) \in \cup_{i=1}^N \mathcal{D}_i, \tag{21}$$

for all $t \in [t_{N-1}, T]$. Also, for any $i = 1, \ldots, N-1$, using the same proposition, we can conclude that for any $\tilde{x}_{i-1} \in L^{x_{i-1}}_{\phi_i, \beta_i}$,

$$\xi(t; \tilde{x}_{i-1}, t_{i-1}) \notin \texttt{Unsafe}, \tag{22}$$
$$(\xi(t; x_{i-1}, t_{i-1}), \xi(t; \tilde{x}_{i-1}, t_{i-1})) \in \cup_{i=1}^N \mathcal{D}_i, \tag{23}$$
$$\xi(t_i; \tilde{x}_{i-1}, t_{i-1}) \in L^{x_i}_{\phi_i, \alpha_i}. \tag{24}$$

By construction, $L^{x_i}_{\phi_i, \alpha_i} \subset L^{x_i}_{\phi_{i+1}, \beta_{i+1}}$. Hence, from (24) we can obtain

$$\xi(t_i; \tilde{x}_{i-1}, t_{i-1}) \in L^{x_i}_{\phi_{i+1}, \beta_{i+1}}.$$

Therefore, by repeated application of Proposition 3, we can prove that this theorem holds.

The result given in Theorem 1 can be easily extended by replacing the barrier functions with constant growth bounds with those with linear growth bounds. In this case, the proof will follow Proposition 4.

4 Computation of Barrier Functions and the Domains of Validity

4.1 General Scheme

In the previous sections we have established some results that describe how to construct a finite-time safety/ reachability type guarantee based on the barrier function ϕ and its domain of validity $\mathcal{D}$. In this section, we propose a computational scheme to construct such barrier function and domain of validity.

Consider the dynamical system in (1).

Proposition 5. *Suppose that the functions $\phi(x)$ and $\gamma(x)$ satisfy*

$$\nabla_x \phi(x) f(x) - k \leq \varepsilon(x)\gamma(x), \tag{25}$$

for some strictly positive function $\varepsilon(x)$, and $k \in \mathbb{R}$. Then $\phi(x)$ is a barrier function with k as its constant growth bound and $\mathcal{D} := \{x \mid \gamma(x) \leq 0\}$ is its domain of validity,

Proof. This construction is based on the S-procedure. From (25), it follows that $\gamma(x) \leq 0$ implies

$$\nabla_x \phi(x) f(x) \leq k.$$

The linear growth bound version of this proposition can be found by replacing k in (25) with $k\phi(x)$. We can use this proposition to generate a barrier function ϕ for a given domain of validity $\mathcal{D}$.

$$\begin{aligned} &\text{Given } \gamma(x), \text{ find } \phi(x) \text{ and } \varepsilon(x) \text{ satisfying} \\ &\nabla_x \phi(x) f(x) - \varepsilon(x)\gamma(x) - k \leq 0,\ \varepsilon(x) \geq 0. \end{aligned} \tag{26}$$

Extending this scheme for safety verification amounts to finding a barrier function $\phi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that is valid in a domain given by $\gamma(x, y) \leq 0$. This can be done by solving the following problem.

$$\begin{aligned} &\text{Given } \gamma(x, y), \text{ find } \phi(x, y) \text{ and } \varepsilon(x, y) \text{ satisfying} \\ &\nabla_x \phi(x, y) f(x) + \nabla_y \phi(x, y) f(y) - \varepsilon(x, y)\gamma(x, y) - k \leq 0,\ \varepsilon(x, y) \geq 0. \end{aligned} \tag{27}$$

For a special class of systems, we can explicitly outline a computational technique that implements this general scheme, as described in the next subsection.

4.2 Affine Systems

If $f(x)$ in (1) is a linear function,

$$f(x) = Ax + b,\ x \in \mathbb{R}^n,\ A \in \mathbb{R}^{n \times n},\ b \in \mathbb{R}^{n \times 1}$$

we can constrain a barrier function to be a quadratic function

$$\phi(x, y) = (x - y)^T M (x - y),$$

for some $M > 0$. If the matrix A is Hurwitz, for barrier function with linear growth bound the domain of validity of the barrier function can be extended globally, by choosing $\gamma(x, y) = 0$. In this case, (27) becomes

$$\nabla_x \phi(x,y) f(x) + \nabla_y \phi(x,y) f(y) - k\phi(x,y) = (x-y)^T (MA + A^T M - kM)(x-y) \leq 0 \tag{28}$$

which is a Lyapunov equation that can be solved for $k \geq 2\lambda(A)$, where $\lambda(A)$ is the largest eigenvalue of A. Obviously, a similar approach also works for nonpositive constant growth bound.

If the matrix A is not Hurwitz, then for barrier function with linear growth bound (28) can still be solved if $k \geq 2\lambda(A)$. For barrier functions with positive constant growth bound, the domain of validity must be bounded. If we choose, for the domain of validity, an ellipsoidal set given by $\gamma(x, y) \leq 0$, where $\gamma(x, y) = (x-y)^T Q (x-y) - 1$, for some $Q > 0$, then (27) becomes finding M and $\varepsilon(x, y)$ satisfying

$$(x-y)^T \left(MA + A^T M - \varepsilon(x,y) Q\right)(x-y) + \varepsilon(x,y) - k \leq 0,\ \varepsilon(x,y) \geq 0, \tag{29}$$

which can be solved by taking $\varepsilon(x, y) = 1$ and M small enough such that $MA + A^T M \leq Q$. Once M is determined, we can find the tightest constant growth bound by solving the following optimization problem

$$\text{minimize } k \text{ subject to (29)},$$

with k and $\varepsilon(x, y)$ as the optimization variables. In this case, we can bound k as

$$k \leq \inf_{\varepsilon \in \mathbb{R}} \{\varepsilon \mid MA + A^T M \leq \varepsilon Q\}. \tag{30}$$

4.3 Locally Linearized Systems

For a locally linearized system $f(x)$ in (1) can be written as,

$$f(x) = Ax + b + \omega(x),\ x \in \mathcal{D} \subset \mathbb{R}^n.$$

Here $Ax + b$ is the linearized model and $\omega(x)$ is the residual term. Suppose that $\mathcal{D}$ is bounded and its diameter is given by

$$\rho(D) := \sup_{x,y \in \mathcal{D}} \|x - y\|,$$

and there exists a $\delta > 0$ such that $\|\omega(x)\| \leq \delta$, for all $x \in \mathcal{D}$. That is, we assume that we can bound the magnitude of the linearization residue in $\mathcal{D}$.

We propose to construct a quadratic barrier function in the form of $\phi(x, y) = (x-y)^T M (x-y)$, $M > 0$. In this case, we obtain

$$\begin{aligned}\nabla_x \phi(x,y) f(x) + \nabla_y \phi(x,y) f(y) &= (x-y)^T (MA + A^T M)(x-y) \\ &\quad + 2(x-y)^T M (\rho(x) - \rho(y)), \\ &\leq (x-y)^T (MA + A^T M)(x-y) + 4\|M\| \delta \rho(D),\end{aligned}$$

where $\|M\|$ is the largest singular value of M.

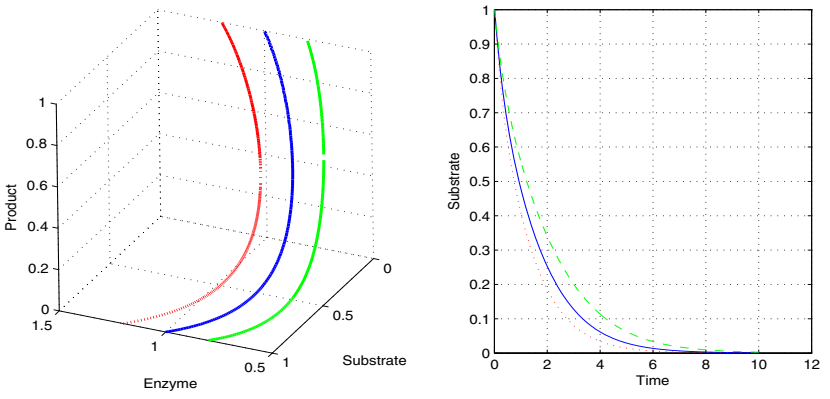


Fig. 5. Three trajectories of the system in Example 1 with varying enzyme availability. In the right panel we can see that smaller enzyme concentration implies slower consumption of the substrate.

If A is Hurwitz, then by following the same computation as in the previous subsection, we can construct a barrier function with constant growth bound by solving the Lyapunov equation $(MA + A^T M) \leq 0$ and the growth bound is $4\|M\|\delta\rho(D)$. If A is not Hurwitz, for any choice of M we can construct a positive constant growth bound for the barrier function by adding $4\|M\|\delta\rho(D)$ to an upper bound of $(x-y)^T (MA + A^T M)(x-y)$ for $x, y \in \mathcal{D}$. This can be done by using the technique described in the previous subsection, or by using the following (possibly conservative) bound

$$(x-y)^T (MA + A^T M)(x-y) \leq \rho(D)^2 \left\| MA + A^T M \right\|. \tag{31}$$

4.4 Polynomial Systems

If $f(x)$ in (1) is a polynomial, and if we assume that $\phi(x)$, $\varepsilon(x)$, and $\gamma(x)$ are polynomials, the semidefinite constraints in (26) can be recast as sum-of-squares constraints. Similar situation applies to (27) for safety verification. In this case, the computation can be implemented by using computational tools for sum-of-squares programming, such as SOSTOOLS [31].

Example 1. A standard model of the dynamics of an enzymatic reaction

$$\text{Enzyme} + \text{Substrate} \rightleftarrows \text{Enz.Sub} \rightarrow \text{Enzyme} + \text{Product}$$

is given by

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} -k_f x_1 x_2 + (k_b + k_m) x_3 \\ -k_f x_1 x_2 + k_b x_3 \\ k_f x_1 x_2 - (k_b + k_m) x_3 \\ k_m x_3 \end{bmatrix},$$

where the state variables are the concentrations of the enzyme, substrate, enzyme-substrate complex, and product, respectively. The constants k_f, k_b, and k_m are

reaction constants that determines the speed of the reactions. Several trajectories of this system are shown in Figure 5. In this simulation, we take $k_b = 0.1$ and $k_f = k_m = 1$. Consider the middle trajectory in Figure 5, which starts at the initial condition $(1, 1, 0, 0)$. Suppose that we take this trajectory as our test trajectory and we want to construct a local barrier function for this system for a given domain of validity. The circular domain of validity is expressed as

$$\gamma(x, y) := (x - c)^T (x - c) + (x - y)^T (x - y) - r^2 \leq 0,$$

where the vector $c = (0.70, 0.51, 0.30.0.19)^T$ defines the center of the circle in the state space and $r = 0.2$ is its radius. We assume that the barrier function can be written as

$$\phi(x, y) := \frac{1}{2}(x - y)^T M (x - y),$$

with M a 4×4 symmetric positive semidefinite matrix. Finding a suitable barrier function by using sum-of-squares programming can be cast as

$$\begin{gathered} \text{minimize } 0 \text{ subject to} \\ -\nabla_x \phi(x, y) f(x) - \nabla_y \phi(x, y) f(x) + \varepsilon(x, y)\gamma(x, y) + k = \text{sos}, \\ \phi(x, y) = \text{sos}, \ \varepsilon(x, y) = \text{sos}. \end{gathered}$$

Solving this problem with SOSTOOLS, we get

$$M = \begin{bmatrix} 0.28 & -0.07 & 0.21 & -0.07 \\ * & 0.19 & 0.11 & 0.19 \\ * & * & 0.16 & 0.11 \\ * & * & * & 0.19 \end{bmatrix}, \ k = 0.02.$$

Notice that we replace nonnegativity of the polynomials with sum-of-squares property, which is more restrictive and can lead to some conservativeness. However, through this step, the program can then be solved using available SOS computational tools.

5 Discussion

In this paper we propose a trajectory based reachability analysis using local finite-time invariance property. This method is a generalization of our previous results [17,18], where a global bisimulation is required for each mode of dynamics. We demonstrate that our method captures some other existing methods as special cases, and that for polynomial systems the computation can be implemented using sum-of-squares.

The extension of the method proposed in this paper to analysis of hybrid systems is relatively straightforward. The method proposed in [17] for hybrid systems performs the analysis on a hybrid test trajectory by piecing together trajectory segments between mode transitions in a way analogous to Theorem 1. We can therefore apply the local analysis based method to hybrid systems by extending Theorem 1 to handle transition guards in a way similar to Proposition 2 in [17].

In order to develop an effective implementation of the result posed in this paper, we still need to design a comprehensive test algorithm. There are a number of issues that need to be addressed along this direction. For example, the notion of test coverage and automatic test generation based on the coverage need to be developed to get an efficient testing procedure that can quickly cover the set of initial states. We also need to address the issue of optimal placement of the local domains of validity of the barrier functions. The goal is to design the segmentation of trajectories in a way that requires as few segments as possible. Another issue that we have not investigated is the use of constant and linear growth bounds. In the case where both bounds are available, we need to design an algorithm that can optimally choose which bound to use, in order to minimize the conservativeness of the bound.

References

1. Balluchi, A., Di Natale, F., Sangiovanni-Vincentelli, A., van Schuppen, J.H.: Synthesis for idle speed control of an automotive engine. In: Alur, R., Pappas, G.J. (eds.) HSCC 2004. LNCS, vol. 2993, pp. 80–94. Springer, Heidelberg (2004)
2. Platzer, A., Quesel, J.-D.: Logical verification and systematic parametric analysis in train control. In: Egerstedt, M., Mishra, B. (eds.) HSCC 2008. LNCS, vol. 4981, pp. 646–649. Springer, Heidelberg (2008)
3. Tomlin, C., Pappas, G.J., Sastry, S.: Conflict resolution for air traffic management: a study in multi-agent hybrid systems. IEEE Trans. Automatic Control 43, 509–521 (1998)
4. Hu, J., Prandini, M., Sastry, S.: Probabilistic safety analysis in three dimensional aircraft flight. In: Proc. 42nd IEEE Conf. Decision and Control, Maui, USA, pp. 5335–5340 (2003)
5. Belta, C., Schug, J., Dang, T., Kumar, V., Pappas, G.J., Rubin, H., Dunlap, P.: Stability and reachability analysis of a hybrid model of luminescence in the marine bacterium *vibrio fischeri*. In: Proc. IEEE Conf. Decision and Control, Orlando, Florida, pp. 869–874 (2001)
6. Ghosh, R., Amondlirdviman, K., Tomlin, C.: A hybrid systems model of planar cell polarity signaling in drosophila melanogaster wing epithelium. In: Proc. IEEE Conf. Decision and Control, Las Vegas (2002)
7. Kurzhanski, A.B., Varaiya, P.: Ellipsoidal technique for reachability analysis. In: Lynch, N.A., Krogh, B.H. (eds.) HSCC 2000. LNCS, vol. 1790, pp. 202–214. Springer, Heidelberg (2000)
8. Mitchell, I., Tomlin, C.J.: Level set methods in for computation in hybrid systems. In: Lynch, N.A., Krogh, B.H. (eds.) HSCC 2000. LNCS, vol. 1790, pp. 310–323. Springer, Heidelberg (2000)
9. Asarin, E., Bournez, O., Dang, T., Maler, O.: Approximate reachability analysis of piecewise-linear dynamical systems. In: Lynch, N.A., Krogh, B.H. (eds.) HSCC 2000. LNCS, vol. 1790, pp. 21–31. Springer, Heidelberg (2000)
10. Kapinski, J., Krogh, B.H.: A new tool for verifying computer controlled systems. In: Proc. IEEE Conf. Computer-Aided Control System Design, Glasgow, pp. 98–103 (2002)
11. Alur, R., Dang, T., Ivancic, F.: Reachability analysis of hybrid systems via predicate abstraction. In: Tomlin, C.J., Greenstreet, M.R. (eds.) HSCC 2002. LNCS, vol. 2289, pp. 35–48. Springer, Heidelberg (2002)

12. Stursberg, O., Krogh, B.H.: Efficient representation and computation of reachable sets for hybrid systems. In: Maler, O., Pnueli, A. (eds.) HSCC 2003. LNCS, vol. 2623, pp. 482–497. Springer, Heidelberg (2003)
13. Han, Z., Krogh, B.H.: Reachability analysis of hybrid control systems using reduced-order models. In: Proc. American Control Conference, pp. 1183–1189 (2004)
14. Prajna, S., Jadbabaie, A.: Safety verification of hybrid systems using barrier certificates. In: Alur, R., Pappas, G.J. (eds.) HSCC 2004. LNCS, vol. 2993, pp. 477–492. Springer, Heidelberg (2004)
15. Frehse, G.: PHAVer: Algorithmic verification of hybrid systems past HyTech. In: Morari, M., Thiele, L. (eds.) HSCC 2005. LNCS, vol. 3414, pp. 258–273. Springer, Heidelberg (2005)
16. Girard, A.: Reachability of uncertain linear systems using zonotopes. In: Morari, M., Thiele, L. (eds.) HSCC 2005. LNCS, vol. 3414, pp. 291–305. Springer, Heidelberg (2005)
17. Julius, A.A., Fainekos, G., Anand, M., Lee, I., Pappas, G.J.: Robust test generation and coverage for hybrid systems. In: Bemporad, A., Bicchi, A., Buttazzo, G. (eds.) HSCC 2007. LNCS, vol. 4416, pp. 329–342. Springer, Heidelberg (2007)
18. Julius, A.A., Pappas, G.J.: Probabilistic testing for stochastic hybrid systems. In: Proc. IEEE Conf. Decision and Control, Cancun, Mexico (2008)
19. Esposito, J.M.: Randomized test case generation for hybrid systems verification. In: Proc. 36th Southeastern Symposium of Systems Theory (2004)
20. Branicky, M.S., Curtiss, M.M., Levine, J., Morgan, S.: RRTs for nonlinear, discrete, and hybrid planning and control. In: Proc. IEEE Conf. Decision and Control, Hawaii, USA (2003)
21. Bhatia, A., Frazzoli, E.: Incremental search methods for reachability analysis of continuous and hybrid systems. In: Alur, R., Pappas, G.J. (eds.) HSCC 2004. LNCS, vol. 2993, pp. 142–156. Springer, Heidelberg (2004)
22. Asarin, A., Dang, T., Girard, A.: Reachability analysis of nonlinear systems using conservative approximation. In: Maler, O., Pnueli, A. (eds.) HSCC 2003. LNCS, vol. 2623, pp. 20–35. Springer, Heidelberg (2003)
23. Han, Z., Krogh, B.H.: Reachability analysis of nonlinear systems using trajectory piecewise linearized models. In: Proc. American Control Conference, Minneapolis (2006)
24. Donzé, A., Maler, O.: Systematic simulation using sensitivity analysis. In: Bemporad, A., Bicchi, A., Buttazzo, G. (eds.) HSCC 2007. LNCS, vol. 4416, pp. 174–189. Springer, Heidelberg (2007)
25. Lohmiller, W., Slotine, J.J.E.: On contraction analysis for nonlinear systems. Automatica 34, 683–696 (1998)
26. Tan, W., Packard, A., Wheeler, T.: Local gain analysis on nonlinear systems. In: Proc. American Control Conference, Minneapolis (2006)
27. Girard, A., Pappas, G.J.: Verification using simulation. In: Hespanha, J.P., Tiwari, A. (eds.) HSCC 2006. LNCS, vol. 3927, pp. 272–286. Springer, Heidelberg (2006)
28. Lerda, F., Kapinski, J., Clarke, E.M., Krogh, B.H.: Verification of supervisory control software using state proximity and merging. In: Egerstedt, M., Mishra, B. (eds.) HSCC 2008. LNCS, vol. 4981, pp. 344–357. Springer, Heidelberg (2008)
29. Boyd, S., El Ghaoui, L., Feron, E., Balakrishnan, V.: Linear Matrix Inequalities in Systems and Control Theory. SIAM, Philadelphia (1994)
30. Khalil, H.K.: Nonlinear Systems, 3rd edn. Prentice-Hall, Englewood Cliffs (2002)
31. Prajna, S., Papachristodoulou, A., Seiler, P., Parillo, P.A.: SOSTOOLS and its control application. In: Positive polynomials in control. Springer, Heidelberg (2005)

Synthesis of Trajectory-Dependent Control Lyapunov Functions by a Single Linear Program

Mircea Lazar and Andrej Jokic

Dept. of Electrical Eng., Eindhoven Univ. of Technology,
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
m.lazar@tue.nl, a.jokic@tue.nl

Abstract. Although control Lyapunov functions (CLFs) provide a mature framework for the synthesis of stabilizing controllers, their application in the field of hybrid systems remains scarce. One of the reasons for this is conservativeness of Lyapunov conditions. This article proposes a methodology that reduces conservatism of CLF design and is applicable to a wide class of discrete-time nonlinear hybrid systems. Rather than searching for global CLFs off-line, we focus on synthesizing CLFs by solving on-line an optimization problem. This approach makes it possible to derive a *trajectory-dependent CLF*, which is allowed to be locally non-monotone. Besides the theoretical appeal of the proposed idea, we indicate that for systems affine in control and CLFs based on infinity norms, the corresponding on-line optimization problem can be formulated as a single linear program.

1 Introduction

Control Lyapunov functions (CLFs) [1,2] represent perhaps the most popular tool for synthesizing control laws that achieve stability. The interested reader is referred to the surveys [3,4] for a complete historical account. The classical approach for smooth continuous-time systems is based on the design of an explicit feedback law off-line, which renders the derivative of a candidate CLF negative. Conditions under which these results can be extended to sampled-data nonlinear systems using their approximate discrete-time models can be found in [5]. An important article on control Lyapunov functions for discrete-time systems is [6]. Therein, classical continuous-time results regarding existence of smooth CLFs are reproduced for the discrete-time case.

Despite the popularity of CLFs within smooth nonlinear systems theory, there is still a significant gap in the usage of CLFs in stabilization of hybrid systems. One of the reasons for this is conservativeness of the sufficient conditions for Lyapunov asymptotic stability [7,8], which are employed by most methods for constructing CLFs. This makes classical CLFs overconservative for discontinuous nonlinear and hybrid systems, as observed in the seminal paper [9]. Ever since, the focus has been on designing less conservative types of Lyapunov functions for specific relevant classes of hybrid systems. For example, piecewise quadratic (PWQ) functions were exploited in stability analysis and synthesis

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 237–251, 2009.

problems for continuous-time and discrete-time piecewise affine (PWA) systems in [10], [11], [12]. Further relaxations were proposed in [13] for discrete-time switched linear systems, using parameter dependent PWQ Lyapunov functions. More recently, a hybrid CLF (which combines two different CLFs) was employed in [14] to stabilize hybrid systems with discrete dynamics (e.g., hybrid systems with discrete states and/or inputs).

To summarize, to some extent, the state-of-the-art methods for stability *analysis* of discrete-time hybrid systems (mostly PWA and switched linear systems) rely on the off-line search for globally defined PWQ Lyapunov functions. One of the most significant relaxations is that each quadratic function, which is part of the PWQ global function, is required to be positive definite and/or satisfy decreasing conditions only in a subset of the state-space, relaxation often referred to as the $\mathcal{S}$-procedure [10]. From a numerical point of view, the existing tools require solving a semidefinite programming problem. However, when it comes to *synthesis* of CLFs, which consists of simultaneously searching for a PWQ Lyapunov function and a state-feedback control law, the $\mathcal{S}$-procedure leads to a nonlinear matrix inequality that has not been solved systematically so far, although serious efforts have been put in this direction.

Next, we present a motivating example which suffers from this drawback.

1.1 Motivating Example

Consider the following piecewise linear (PWL) system from [8], Chapter 3:

$$x(k+1) = A_j x(k) + Bu(k) \quad \text{if} \quad E_j x(k) \geq 0, \quad k \in \mathbb{Z}_+, \tag{1}$$

where $j = \{1, 2, 3, 4\}$,

$$A_1 = \begin{bmatrix} 0.5 & 0.61 \\ 0.9 & 1.345 \end{bmatrix}, \; A_2 = \begin{bmatrix} -0.92 & 0.644 \\ 0.758 & -0.71 \end{bmatrix}, \; B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \; A_3 = A_1, \; A_4 = A_2.$$

The partitioning of the state-space is given by

$$E_1 = -E_3 = \begin{bmatrix} -1 & 1 \\ -1 & -1 \end{bmatrix}, \; E_2 = -E_4 = \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}.$$

As shown in [8] the synthesis problem[1] for this system in closed-loop with a PWL state-feedback law is not feasible for a common quadratic or a PWQ Lyapunov function without the $\mathcal{S}$-procedure relaxation. However, a solution to the synthesis problem for a PWQ Lyapunov function with the $\mathcal{S}$-procedure has been found in [8] at the expense of a significant computational complexity (i.e. a griding approach was used to solve a bilinear matrix inequality).

This indicates that there are even very simple classes of discrete-time hybrid systems for which a systematic and efficient synthesis method based on CLFs is not available.

[1] Notice that the above example is a "flower system" for the synthesis problem, similarly as the example introduced in [10] is a "flower system" for the analysis problem.

Remark 1. Existing on-line optimization based controllers, such as model predictive control algorithms, make use of the above-mentioned off-line synthesis methods to obtain an a priori stability guarantee, see, for example, [15, 16]. Hence, they are also affected by the limitations of these methods. □

Motivated by the above example, in this paper we propose a new methodology that reduces significantly the conservatism of CLF design for discrete-time systems. Rather than searching for global CLFs off-line, we focus on synthesizing time-variant CLFs by solving on-line an optimization problem. As such, trajectory-dependent CLFs that are allowed to be locally non-monotone can be derived. This approach offers the "least conservative" relaxation possible, in the sense that for a CLF with a fixed structure that incorporates some time-variant parameters, a possibly different value of these parameters is assigned to each measured state. Furthermore, the stabilization conditions that involve the CLF are only imposed along the closed-loop trajectory generated on-line. Numerically, we indicate that for piecewise continuous (PWC) nonlinear systems affine in control and CLFs based on infinity norms, the on-line optimization problem can be formulated as a single linear program. The effectiveness of the developed theory is demonstrated on the motivating example presented above.

2 Preliminaries

In this section we recall preliminary notions and fundamental stability results.

2.1 Basic Notions and Definitions

Let $\mathbb{R}$, $\mathbb{R}_+$, $\mathbb{Z}$ and $\mathbb{Z}_+$ denote the field of real numbers, the set of non-negative reals, the set of integer numbers and the set of non-negative integers, respectively. We use the notation $\mathbb{Z}_{\geq c_1}$ and $\mathbb{Z}_{(c_1,c_2]}$ to denote the sets $\{k \in \mathbb{Z}_+ \mid k \geq c_1\}$ and $\{k \in \mathbb{Z}_+ \mid c_1 < k \leq c_2\}$, respectively, for some $c_1, c_2 \in \mathbb{Z}_+$. For a set $\mathcal{S} \subseteq \mathbb{R}^n$, we denote by $\mathrm{int}(\mathcal{S})$ the interior and by $\mathrm{cl}(\mathcal{S})$ the closure of $\mathcal{S}$. A polyhedron (or a polyhedral set) in $\mathbb{R}^n$ is a set obtained as the intersection of a finite number of open and/or closed half-spaces. For a vector $x \in \mathbb{R}^n$, $[x]_i$ denotes the i-th element of x. A vector $x \in \mathbb{R}^n$ is said to be nonnegative (nonpositive) if $[x]_i \geq 0$ ($[x]_i \leq 0$) for all $i \in \{1, \dots n\}$, and in that case we write $x \geq 0$ ($x \leq 0$). For a vector $x \in \mathbb{R}^n$ let $\|\cdot\|$ denote an arbitrary p-norm. Let $\|x\|_\infty := \max_{i=1,\dots,n} |[x]_i|$, where $|\cdot|$ denotes the absolute value. In the Euclidean space $\mathbb{R}^n$ the standard inner product is denoted by $\langle \cdot, \cdot \rangle$ and the associated norm is denoted by $\|\cdot\|_2$, i.e. for $x \in \mathbb{R}^n$, $\|x\|_2 = \langle x, x \rangle^{\frac{1}{2}} = (x^\top x)^{\frac{1}{2}}$. For a matrix $Z \in \mathbb{R}^{m \times n}$, $[Z]_{ij}$ denotes the element in the i-th row and j-th column of Z. Given $Z \in \mathbb{R}^{m \times n}$ and $I \subseteq \{1, \dots, m\}$, we write $[Z]_{I\bullet}$ to denote a submatrix of Z formed by rows I of Z. For a matrix $Z \in \mathbb{R}^{m \times n}$ let $\|Z\| := \sup_{x \neq 0} \frac{\|Zx\|}{\|x\|}$ denote its corresponding induced matrix norm. It is well known that $\|Z\|_\infty = \max_{1 \leq i \leq m} \sum_{j=1}^{n} |[Z]_{ij}|$.

A function $\varphi : \mathbb{R}_+ \to \mathbb{R}_+$ *belongs to class* $\mathcal{K}$ if it is continuous, strictly increasing and $\varphi(0) = 0$. A function $\varphi : \mathbb{R}_+ \to \mathbb{R}_+$ *belongs to class* $\mathcal{K}_\infty$ if $\varphi \in \mathcal{K}$ and it is radially unbounded (i.e. $\lim_{s \to \infty} \varphi(s) = \infty$). A function $\beta : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}_+$

belongs to class $\mathcal{KL}$ if for each fixed $k \in \mathbb{R}_+$, $\beta(\cdot, k) \in \mathcal{K}$ and for each fixed $s \in \mathbb{R}_+$, $\beta(s, \cdot)$ is decreasing and $\lim_{k\to\infty} \beta(s, k) = 0$.

2.2 Lyapunov Asymptotic Stability

Consider the discrete-time autonomous nonlinear system

$$x(k+1) \in \Phi(x(k)), \quad k \in \mathbb{Z}_+, \tag{2}$$

where $x(k) \in \mathbb{R}^n$ is the state at the discrete-time instant k and the mapping $\Phi : \mathbb{R}^n \hookrightarrow \mathbb{R}^n$ is an arbitrary nonlinear, possibly discontinuous, set-valued function. For simplicity, we assume that the origin is an equilibrium of (2), i.e. $\Phi(0) = \{0\}$. The following definitions give a strong characterization of invariance and stability for the difference inclusion (2), in the sense that these properties are required to hold for all possible trajectories generated by (2), and not just for one of them.

Definition 1. *We call a set* $\mathcal{P} \subseteq \mathbb{R}^n$ positively invariant (PI) *for system* (2) *if for all* $x \in \mathcal{P}$ *it holds that* $\Phi(x) \subseteq \mathcal{P}$.

Definition 2. *Let* $\mathbb{X}$ *with* $0 \in \text{int}(\mathbb{X})$ *be a subset of* $\mathbb{R}^n$. *We call system* (2) *AS(*$\mathbb{X}$*) if there exists a* $\mathcal{KL}$*-function* $\beta(\cdot, \cdot)$ *such that, for each* $x(0) \in \mathbb{X}$ *it holds that all corresponding state trajectories of* (2) *satisfy* $\|x(k)\| \leq \beta(\|x(0)\|, k)$, $\forall k \in \mathbb{Z}_+$. *We call system* (2) *globally asymptotically stable if it is AS(*$\mathbb{R}^n$*).*

Theorem 1. *Let* $\mathbb{X}$ *be a PI set for* (2) *with* $0 \in \text{int}(\mathbb{X})$. *Furthermore, let* $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$, $\rho \in \mathbb{R}_{[0,1)}$ *and let* $V : \mathbb{Z}_+ \times \mathbb{R}^n \to \mathbb{R}_+$ *be a function such that:*

$$\alpha_1(\|x\|) \leq V(k, x) \leq \alpha_2(\|x\|), \ \forall x \in \mathbb{X}, \forall k \in \mathbb{Z}_+, \tag{3a}$$

$$\forall x(0) \in \mathbb{X}, \quad V(k+1, x^+) \leq \rho V(k, x(k)) \tag{3b}$$

for all $x^+ \in \Phi(x(k))$, $k \in \mathbb{Z}_+$. *Then system* (2) *is AS(*$\mathbb{X}$*).*

The proof of the above theorem can be obtained *mutatis mutandis* from the proofs given in [17, 8] by replacing the difference *equation* with the difference *inclusion* as in (2). It is worth to point out that if $V(\cdot)$ is a continuous and time-invariant function, the above theorem can be recovered from Theorem 2.8 of [18], which gives sufficient conditions for robust $\mathcal{KL}$-stability of difference inclusions. We call a function $V(\cdot, \cdot)$ that satisfies (3) a *time-variant Lyapunov function.*

3 Trajectory-Dependent CLFs for Discrete-Time Systems

Consider the discrete-time constrained nonlinear system

$$x(k+1) = \phi(x(k), u(k)), \quad k \in \mathbb{Z}_+, \tag{4}$$

where $x(k) \in \mathbb{X} \subseteq \mathbb{R}^n$ is the state and $u(k) \in \mathbb{U} \subseteq \mathbb{R}^m$ is the control input. $\phi : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is an arbitrary nonlinear function, possibly discontinuous, with $\phi(0, 0) = 0$. We assume that $0 \in \text{int}(\mathbb{X})$ and $0 \in \text{int}(\mathbb{U})$.

Remark 2. Since we allow $\phi(\cdot,\cdot)$ to be discontinuous, the following fairly wide class of hybrid systems is accounted for; *Piecewise continuous nonlinear systems*:

$$x(k+1) = \phi(x(k), u(k)) := \phi_j(x(k), u(k)) \text{ if } x(k) \in \Omega_j, \quad k \in \mathbb{Z}_+, \tag{5}$$

where each $\phi_j : \Omega_j \times \mathbb{U} \to \mathbb{R}^n$, $j \in \mathcal{S}$, is a continuous, possibly nonlinear function in x and $\mathcal{S} := \{1, 2, \ldots, s\}$ is a finite set of indices. The collection $\{\Omega_j \subseteq \mathbb{R}^n \mid j \in \mathcal{S}\}$ defines a partition of $\mathbb{X}$, meaning that $\cup_{j\in\mathcal{S}}\Omega_j = \mathbb{X}$ and $\Omega_i \cap \Omega_j = \emptyset$, with the sets Ω_j not necessarily closed. In most sections of the paper we will omit the explicit reference to the functions $\phi_j(\cdot,\cdot)$ and the partition $\{\Omega_j\}_{j\in\mathcal{S}}$ for brevity. □

Definition 3. *Let $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$ and let $\rho \in \mathbb{R}_{[0,1)}$. A function $V : \mathbb{Z}_+ \times \mathbb{R}^n \to \mathbb{R}_+$ that satisfies*

$$\alpha_1(\|x\|) \le V(k,x) \le \alpha_2(\|x\|), \quad \forall x \in \mathbb{X}, \forall k \in \mathbb{Z}_+ \tag{6}$$

and for which there exists a control law $u : \mathbb{X} \to \mathbb{U}$ such that for any $x(0) \in \mathbb{X}$

$$V(k+1, \phi(x(k), u(x(k)))) \le \rho V(k, x(k)) \quad \text{for all } k \in \mathbb{Z}_+$$

is called a time-variant control Lyapunov function (tvCLF) in $\mathbb{X}$ for system (4).

Next, based on Definition 3, we formulate the following optimization problem.

Problem 1. Let $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$, $\rho \in \mathbb{R}_{[0,1)}$ and the structure of a candidate tvCLF $V(\cdot,\cdot)$ be fixed such that (6) holds for all $x \in \mathbb{X}$ and all $k \in \mathbb{Z}_+$. At time $k \in \mathbb{Z}_+$ measure $x(k)$ and calculate $V(k, x(k))$ and a control action $u(k)$ such that:

$$u(k) \in \mathbb{U}, \quad \phi(x(k), u(k)) \in \mathbb{X}, \tag{7a}$$
$$V(k, \phi(x(k), u(k))) \le \rho V(k, x(k)), \tag{7b}$$
$$V(k, x(k)) \le V(k-1, x(k)) \text{ if } k \in \mathbb{Z}_{\ge 1}. \tag{7c}$$

□

The reasoning employed to construct the constraints in Problem 1 is graphically depicted in Figure 1, first plot from left to right. Let $\pi(x(k)) := \{u(x(k)) \mid \exists V(k,\cdot) \text{ s.t. } (6) - (7) \text{ hold}\}$ and let $\phi_{\text{cl}}(x, \pi(x)) := \{\phi(x,u) \mid u \in \pi(x)\}$. Notice that for a given $x(0) \in \mathbb{X}$, the inequalities (7) generate, besides a sequence of sets of feasible control actions $\{\pi(x(k))\}_{k\in\mathbb{Z}_+}$, also a sequence of sets of feasible realizations of a tvCLF, i.e. $\mathcal{V}(V(k-1), x(k)) := \{V(k,\cdot) \mid \exists u(x(k) \text{ s.t. } (6) - (7) \text{ hold})\}$ for any $k \in \mathbb{Z}_{\ge 1}$. Implicitly, $\pi(x(k))$ also depends on $V(k-1,\cdot)$, but we omitted this dependency for brevity of notation. At $k = 0$, both $\pi(x(0))$ and $\mathcal{V}(x(0))$ depend on $x(0)$ only and their definition is recovered by removing (7c) in the definitions given above for $k \in \mathbb{Z}_{\ge 1}$.

Theorem 2. *Let $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$ be given. Suppose that Problem 1 is feasible for all states x in $\mathbb{X}$. Then the difference inclusion*

$$x(k+1) \in \phi_{cl}(x(k), \pi(x(k))), \quad k \in \mathbb{Z}_+, \tag{8}$$

is AS($\mathbb{X}$).

Proof. Let $x(k) \in \mathbb{X}$ for some $k \in \mathbb{Z}_+$. Then, feasibility of Problem 1 ensures that $x(k+1) \in \phi_{\text{cl}}(x(k), \pi(x(k))) \subseteq \mathbb{X}$ due to constraint (7a). Hence, Problem 1 remains feasible and thus, $\mathbb{X}$ is a PI set for system (8). As $V(k,x)$ satisfies (6) for all $x \in \mathbb{X}$ and all $k \in \mathbb{Z}_+$ by assumption and hence, it satisfies (3a), we only need to show that $V(k, x(k))$ also satisfies inequality (3b) for all $x(0) \in \mathbb{X}$ and all $k \in \mathbb{Z}_+$. At time $k = 0$, for any $x(0) \in \mathbb{X}$ we have that $V(0, \phi(x(0), u(0))) \leq \rho V(0, x(0))$ for all $u(0) \in \pi(x(0))$. Furthermore, at time $k = 1$ it holds that $V(1, x(1)) \leq V(0, x(1)) = V(0, \phi(x(0), u(0))) \leq \rho V(0, x(0))$ for all $u(0) \in \pi(x(0))$. Thus, inequality (3b) holds for system (8) for $k = 0$ and all $x(0) \in \mathbb{X}$. We will show next that inequality (3b) holds for any $k \in \mathbb{Z}_{\geq 1}$. Due to positive invariance of $\mathbb{X}$, for any $x(0) \in \mathbb{X}$ we have that inequality (7b) is feasible at time k and inequality (7c) is feasible at time $k+1$ for any $k \in \mathbb{Z}_{\geq 1}$. Hence,

$$V(k+1, x(k+1)) \leq V(k, x(k+1)) = V(k, \phi(x(k), u(k))) \leq \rho V(k, x(k)),$$

for all $u(k) \in \pi(x(k))$, $k \in \mathbb{Z}_{\geq 1}$ and all $x(0) \in \mathbb{X}$. Then, AS($\mathbb{X}$) of system (8) follows from Theorem 1. □

Notice that the result of Theorem 2 is of the type "feasibility implies stability" and as such, we have assumed that Problem 1 is feasible for all $x \in \mathbb{X}$. For a given $x(0) \in \mathbb{X}$, by solving Problem 1 on-line in a receding horizon fashion (assuming that it remains feasible at all future instances), one does not obtain a tvCLF in $\mathbb{X}$, but only a tvCLF valid for the corresponding closed-loop state trajectory $\{x(k)\}_{k \in \mathbb{Z}_+}$. Therefore, it makes sense to introduce the following formal definition.

Definition 4. *Consider Problem 1. For any $x(0) \in \mathbb{X}$ such that the sets $\pi(x(0))$, $\pi(x(k))$, $\mathcal{V}(x(0))$ and $\mathcal{V}(V(k-1,\cdot), x(k))$ are non-empty for all $k \in \mathbb{Z}_{\geq 1}$, we call a sequence $\{V(k,\cdot)\}_{k \in \mathbb{Z}_+}$ with $V(0,\cdot) \in \mathcal{V}(x(0))$, $V(k,\cdot) \in \mathcal{V}(V(k-1,\cdot), x(k))$ for all $k \in \mathbb{Z}_{\geq 1}$ a* trajectory-dependent control Lyapunov function (tdCLF).

It is interesting to point out that a trajectory-dependent CLF can be interpreted as an approximation along a particular trajectory of a possibly very complex global time-invariant CLF. The tdCLF concept can also be extended to deal with a set of trajectories that originate from a particular set of initial conditions of interest.

Furthermore, observe that the $\mathcal{S}$-procedure relaxation proposed in [10] for PWA systems is recovered as a particular case of the design presented in this section, i.e. for $V(k, x(k))$ time invariant as long as $x(k) \in \Omega_j$ for some $j \in \mathcal{S}$.

In the next subsection we will briefly discuss the possibility of enlarging the feasible domain of Problem 1 considerably.

3.1 Non-monotone tdCLFs

The inequalities (7) can be further significantly relaxed by allowing the candidate tdCLF to be locally non-monotone. This can be done by replacing the inequalities (7b) and (7c) with:

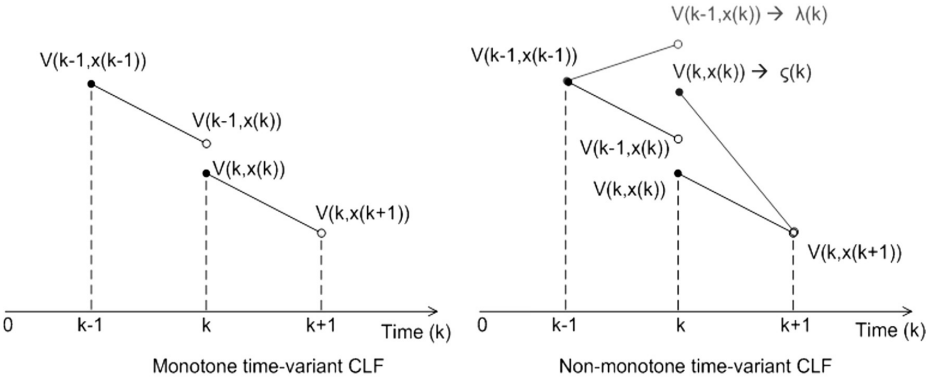


Fig. 1. A graphical illustration of tvCLFs

$$V(k, \phi(x(k), u(k))) \leq \rho V(k, x(k)) + \lambda(k), \tag{9a}$$
$$V(k, x(k)) \leq V(k-1, x(k)) + \zeta(k) \text{ if } k \in \mathbb{Z}_{\geq 1}, \tag{9b}$$

respectively, where $\lambda(k) \in \mathbb{R}_+$ and $\zeta(k) \in \mathbb{R}_+$ are additional variables. For a graphical illustration see Figure 1, the second plot from left to right. Whenever $\lambda(k) \to 0$ and $\zeta(k) \to 0$ as $k \to \infty$ is a priori guaranteed, the closed-loop asymptotic stability result of Theorem 2 still holds. An appealing solution to guarantee this property is to define $\lambda(k)$ and $\zeta(k)$ as outputs of an artificial dynamical system. Then the behavior of $\lambda(k)$ and $\zeta(k)$ can be kept non-monotone, which implies non-monotonicity of $V(\cdot, \cdot)$, while $\lim_{k\to\infty} \lambda(k) = 0$ and $\lim_{k\to\infty} \zeta(k) = 0$ can be ensured through partial stability [19] of the artificial system. The construction of such an artificial system is the object of undergoing research. Alternatively, $\lambda(k) \in \mathbb{R}_+$ and $\zeta(k) \in \mathbb{R}_+$ can be set as optimization variables. Then, adding a suitably defined[2] cost function $J(\lambda(k), \zeta(k))$ to Problem 1 and minimizing over $J(\cdot, \cdot)$ for a given $x(k)$ results in optimizing the trade-off between *(i)* feasibility of Problem 1 and *(ii)* stabilization.

Remark 3. The relaxation proposed in (9b) recovers as a particular case the one proposed in [9], where it is allowed for the Lyapunov function not to decrease when the system switches from one mode to another, i.e. when $x(k-1) \in \Omega_j$ and $x(k) \in \Omega_i$ for some $(i, j) \in \mathcal{S} \times \mathcal{S}, i \neq j$. Furthermore, observe that the two additional variables allow two types of non-monotone behavior of the tdCLF: λ allows non-monotonicity of the tdCLF for fixed $k \in \mathbb{Z}_+$, while ζ allows non-monotonicity of the tdCLF for fixed $x(k)$. The solution based on λ was also used in [14] to stabilize hybrid systems with discrete dynamics (e.g., with discrete states and/or inputs) via non-monotone time-invariant CLFs. □

In the remainder of the paper, for simplicity of exposition, we no longer consider non-monotone tdCLFs. However, all the derivations presented in the next section for Problem 1 trivially apply also to the case when (7b)-(7c) are replaced by (9a)-(9b), as λ and ζ, respectively, enter the latter inequalities linearly.

[2] $J(\cdot, \cdot) : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}_+$ is radially unbounded and $J(0, 0) = 0$.

4 Synthesis of tdCLFs by a Single Linear Program

In this section we consider candidate tvCLFs of a fixed structure and with a set of unknown parameters to be determined on-line, which yields a tdCLF. More precisely, we restrict our attention to cases where $V(k, x(k)) = V(p(k), x(k))$, $V(\cdot, \cdot)$ is *a priori* defined and $p(k)$ is a vector of parameters. For example $V(k, x(k)) = \|P(k)x(k)\|$ or $V(k, x(k)) = x^\top(k)P(k)x(k)$, where in both cases the elements of $P(k)$ are the unknown parameters which are to be determined on-line so that (6)-(7) hold. Furthermore, to make solving Problem 1 on-line tractable, it is desirable that the inequalities (6)-(7) are convex in both the control input $u(k)$ and the parameters $p(k)$. In this respect we point out to some features of Problem 1 as follows. For a given $x(k)$, the lower bound inequality in (6) imposes convex constraints on the parameters $p(k)$ if and only if $V(p(k), x(k))$ is *concave* in $p(k)$, while the upper bound inequality in (6) imposes convex constraints on the parameters $p(k)$ if and only if $V(p(k), x(k))$ is *convex* in $p(k)$. Similarly, suppose that $\phi(x(k), u(k))$ and $x(k)$ are fixed. Then inequality (7b) imposes convex constraints on $p(k)$ if and only if $V(p(k), x(k))$ is both convex and concave in $p(k)$, i.e. if it is *affine* or *linear* in $p(k)$. To summarize, the inherent feature of Problem 1 is that, in general, it is a nonconvex problem.

In the remainder of this section we present a complete convexification procedure for the following fairly general case. In terms of the class of systems, we restrict our attention to PWC nonlinear systems that are affine in the control input, i.e.:

$$\begin{aligned} x(k+1) = \phi(x(k), u(k)) &= \phi_j(x(k), u(k)) \text{ if } x(k) \in \Omega_j \\ &= f_j(x(k)) + g_j(x(k))u(k) \text{ if } x(k) \in \Omega_j, \quad k \in \mathbb{Z}_+ \end{aligned}$$

where $f_j(\cdot)$ and $g_j(\cdot)$ denote suitably defined continuous nonlinear functions. For brevity, let $f(x) := f_j(x)$ and $g(x) := g_j(x)$ if $x \in \Omega_j$, respectively. Observe that PWA systems are a subclass of the above system. Also, we assume that the sets $\mathbb{X}$ and $\mathbb{U}$ are polyhedra. In terms of candidate tvCLFs, we restrict our attention to functions defined using the infinity norm, i.e.:

$$V(k, x(k)) := \|P(k)x(k)\|_\infty,$$

where $P(k) \in \mathbb{R}^{p \times n}$ is to be computed on-line so that (6)-(7) hold.

4.1 Construction of the Lower and Upper Bound

For a fixed $x(k) \in \mathbb{X}$ let

$$\mathcal{P}(x(k)) := \{\, y \in \mathbb{R}^n \mid \langle y, x(k) \rangle \geq 0 \,\}, \tag{10}$$

and let $\mathcal{P}_i(x(k)) \subset \mathbb{R}^n$ for $i \in \{1, \ldots, p\} =: \mathcal{I}$, $p \geq n$, be compact sets. Furthermore, define

$$\mathit{\Pi}(x(k)) := \{P \in \mathbb{R}^{p \times n} \;\mid\; [P]_{i\bullet} \in \mathcal{P}_i(x(k)),\ \forall i \in \mathcal{I}\}, \tag{11}$$

and suppose that the collection of sets $\{\mathcal{P}_i(x(k))\}_{i\in\mathcal{I}}$ is such that:

$$\mathcal{P}_i(x(k)) \subset \mathcal{P}(x(k)), \quad \forall i \in \mathcal{I}, \tag{12a}$$
$$P(k) \in \Pi(x(k)) \quad \Rightarrow \quad \operatorname{rank}(P(k)) = n, \tag{12b}$$
$$\mathcal{P}_i(x(k)) \cap \mathcal{B}_{r_1(k)} = \emptyset, \quad \mathcal{P}_i(x(k)) \subset \mathcal{B}_{r_2(k)}, \quad \forall i \in \mathcal{I}, \tag{12c}$$

where $\mathcal{B}_{r_i(k)} := \{z \in \mathbb{R}^n \mid \|z\|_2 < r_i(k)\}$, $i = 1, 2$, for some $r_1(k), r_2(k) \in \mathbb{R}_{>0}$, $r_1(k) < r_2(k)$ for all $k \in \mathbb{Z}_+$.

Lemma 1. *Let $x(k) \in \mathbb{X}$, $k \in \mathbb{Z}_+$, be fixed and let $\{\mathcal{P}_i(x(k))\}_{i\in\mathcal{I}}$ satisfy (12). Then*
(i) $P(k)x(k) \geq 0$ for all $P(k) \in \Pi(x(k))$;
(ii) $\exists \alpha_1 \in \mathcal{K}_\infty$ such that $\|P(k)z\|_\infty \geq \alpha_1(\|z\|_\infty)$, $\forall z \in \mathbb{R}^n$, $\forall P(k) \in \Pi(x(k))$;
(iii) $\exists \alpha_2 \in \mathcal{K}_\infty$ such that $\|P(k)z\|_\infty \leq \alpha_2(\|z\|_\infty)$, $\forall z \in \mathbb{R}^n$, $\forall P(k) \in \Pi(x(k))$.

Proof. (*i*) Follows directly from (12a) and the definitions of $\mathcal{P}(x(k))$ and $\Pi(x(k))$.
(*ii*) Let

$$c(k) := \max_{i\in\mathcal{I}} \min_{z\neq 0} \min_{y\in\mathcal{P}_i(x(k))} \frac{|\langle y, z\rangle|}{\|y\|_2\|z\|_2}. \tag{13}$$

Note that $c(k)$ is well defined, as from (12c) we have that for each $i \in \mathcal{I}$, $y \in \mathcal{P}_i(x(k)) \Rightarrow y \not\equiv 0$. For any $z \in \mathbb{R}^n \setminus \{0\}$, $\max_{i\in\mathcal{I}} \min_{y\in\mathcal{P}_i(x(k))} \frac{|\langle y,z\rangle|}{\|y\|_2\|z\|_2} > 0$, since from (12b) it follows that there always exists a $j \in \mathcal{I}$ such that $y \in \mathcal{P}_j(x(k)) \Rightarrow \langle y, z\rangle \neq 0$. Hence, $c(k) \neq 0$ and thus, $c(k) > 0$. Now, let $P(k) \in \Pi(x(k))$ and for notational convenience let $p_i(k) := [P(k)]_{i\bullet}^\top$. Then we can write the following sequence of equalities

$$\|P(k)z\|_\infty = \|(\langle p_1(k), z\rangle, \ldots, \langle p_p(k), z\rangle)^\top\|_\infty = \max_{i\in\mathcal{I}} |\langle p_i(k), z\rangle|. \tag{14}$$

Furthermore, for any fixed $p_i(k)$ and any $z \neq 0$ we have

$$\max_{i\in\mathcal{I}} \frac{|\langle p_i(k), z\rangle|}{\|p_i(k)\|_2\|z\|_2} \geq \max_{i\in\mathcal{I}} \min_{\tilde{z}\neq 0} \min_{\tilde{y}\in\mathcal{P}_i(x(k))} \frac{|\langle \tilde{y}, \tilde{z}\rangle|}{\|\tilde{y}\|_2\|\tilde{z}\|_2} = c(k). \tag{15}$$

Therefore, using (12c) and $\|z\|_\infty \leq \|z\|_2$, yields:

$$\max_{i\in\mathcal{I}} |\langle p_i(k), z\rangle| \geq c(k)\|p_i(k)\|_2\|z\|_2 \geq c(k)r_1(k)\|z\|_\infty,$$

which together with (14) implies $\|P(k)z\|_\infty \geq c(k)r_1(k)\|z\|_\infty$. Since $P(k)$ is an arbitrary matrix in $\Pi(x(k))$, we conclude that the desired inequality holds with $\alpha_1(\|z\|_\infty) := \inf_{k\in\mathbb{Z}_+} c(k)r_1(k)\|z\|_\infty$.
(*iii*) For any $P(k) \in \Pi(x(k))$ we have that

$$\|P(k)z\|_\infty \leq \|P(k)\|_\infty\|z\|_\infty = \max_{i\in\mathcal{I}} \|p_i(k)\|_1\|z\|_\infty. \tag{16}$$

Using the fact that $\|p_i(k)\|_1 \leq n\|p_i(k)\|_2$ and the property (12c), inequality (16) further implies that $\|P(k)z\|_\infty \leq n\, r_2(k)\, \|z\|_\infty$. Since $P(k)$ is an arbitrary matrix in $\Pi(x(k))$, we conclude that the desired inequality holds with $\alpha_2(\|z\|_\infty) := n \sup_{k\in\mathbb{Z}_+} r_2(k)\|z\|_\infty$. □

Notice that the upper and lower bounds established in the proof of Lemma 1 can be derived explicitly as follows. Some tuning parameters $R_1, R_2 \in \mathbb{R}_{>0}$ can always be a priori chosen such that $R_1 \leq r_1(k) < r_2(k) \leq R_2$ for all $k \in \mathbb{Z}_+$. This will be illustrated in Section 4.3, where it is also shown how to derive a number $C \in \mathbb{R}_{>0}$ such that $c(k) \geq C$ for all $k \in \mathbb{Z}_+$. Furthermore, therein we present an approach to the derivation of the collection of sets $\{\mathcal{P}_i(x(k))\}_{i\in\mathcal{I}}$ such that (12) holds. Another clarifying point is that the result of Lemma 1-(i) will be instrumental in the convexification of inequality (7b).

4.2 Convexification of Problem 1

Next, let $D(x(k))$ and $d(x(k))$ be a matrix and a vector of appropriate dimensions such that at each time k the inequalities (7a) are equivalently written as $D(x(k))u(k) \leq d(x(k))$. Note that with the hypothesis that $\mathbb{X}$ and $\mathbb{U}$ are polyhedra, this can always be done for PWC nonlinear systems affine in control.

Problem 2. Let $C, R_1, R_2 \in \mathbb{R}_{>0}$ with $R_1 < R_2$ be given such that $C \leq c(k)$ and $R_1 \leq r_1(k) < r_2(k) \leq R_2$ for all $k \in \mathbb{Z}_+$. At time $k \in \mathbb{Z}_+$ let $x(k)$ be the measured state. Determine the partition $\{\mathcal{P}_i(x(k))\}_{i\in\mathcal{I}}$, $\mathcal{I} = \{1, \ldots, p\}$, $p \geq n$, such that (12) holds for some $r_1(k), r_2(k) \in \mathbb{R}_{>0}$, $r_1(k) < r_2(k)$. Then, compute $P(k) \in \mathbb{R}^{p\times n}$, $\tau(k) \in \mathbb{R}^m$ and $\xi(k) \in \mathbb{R}$ such that

$$D(x(k))\tau(k) \leq \xi(k)d(x(k)), \tag{17a}$$
$$\|\xi(k)f(x(k)) + g(x(k))\tau(k)\|_\infty - \rho[P(k)x(k)]_{i\bullet} \leq 0, \quad \forall i \in \mathcal{I}, \tag{17b}$$
$$\|P(k)\|_\infty \leq \xi(k), \tag{17c}$$
$$[P(k)]_{i\bullet} \in \mathcal{P}_i(x(k)), \quad \forall i \in \mathcal{I}, \tag{17d}$$
$$\|P(k)x(k)\|_\infty \leq \|P(k-1)x(k)\|_\infty, \quad \text{if} \quad k \in \mathbb{Z}_{\geq 1}. \tag{17e}$$

□

Lemma 2. *Let $P(k)$, $\tau(k)$ and $\xi(k)$ denote a feasible solution of Problem 2 for state $x(k)$ at time $k \in \mathbb{Z}_+$ and let $[u(k)]_i := \frac{[\tau(k)]_i}{\xi(k)}$ for $i = 1, \ldots, m$. Then $V(k, x) := \|P(k)x\|_\infty$ and $u(k)$ are a feasible solution of Problem 1 for state $x(k)$ at time $k \in \mathbb{Z}_+$.*

Proof. Since (17d) implies that $P(k) \not\equiv 0$, we obtain $\|P(k)\|_\infty \neq 0$ and thus, from (17c) it follows that $\xi(k) > 0$. This implies that $u(k)$ is indeed well-defined, and that we can pull out $\xi(k)$ from the norm in (17b). By Lemma 1-(i), we have that (17d) $\Rightarrow P(k)x(k) \geq 0$. Furthermore, from with (17c) and (17b) we obtain:

$$\begin{aligned} 0 &\geq \xi(k)\|f(x(k)) + g(x(k))\frac{\tau(k)}{\xi(k)}\|_\infty - \rho\|P(k)x(k)\|_\infty \\ &\geq \|P(k)\|_\infty\|f(x(k)) + g(x(k))u(k)\|_\infty - \rho\|P(k)x(k)\|_\infty \\ &\geq \|P(k)(f(x(k)) + g(x(k))u(k))\|_\infty - \rho\|P(k)x(k)\|_\infty, \end{aligned} \tag{18}$$

i.e. (7b) holds. Furthermore, from (17a) we have that

$$D(x(k))\frac{\tau(k)}{\xi(k)} = D(x(k))u(k) \leq d(x(k)),$$

and therefore (7a) is satisfied. Using Lemma 1, the inequality (17d) and $R_1 \leq r_1(k) < r_2(k) \leq R_2$ for all $k \in \mathbb{Z}_+$ we obtain (6) with $\alpha_1(\|x\|_\infty) := CR_1\|x\|_\infty$ and $\alpha_2(\|x\|_\infty) := nR_2\|x\|_\infty$. The proof is concluded by observing that (17e) is just (7c). □

Remark 4. Suppose that each $\mathcal{P}_i(x(k))$ in Problem 2 is a convex set. Then Problem 2 amounts to finding a feasible solution to a set of *convex* inequalities, and it implicitly solves a non-convex optimization problem, i.e. Problem 1. □

4.3 Construction of a Collection of Polyehedral Sets $\{\mathcal{P}_i(x(k))\}_{i\in\mathcal{I}}$

In parallel with the general description of the procedure we will refer to the following example for illustrative purposes. Suppose that $x \in \mathbb{R}^2$. For a fixed $x(k)$, $k \in \mathbb{Z}_+$, Figure 2 illustrates a possible choice of the collection of sets $\{\mathcal{P}_i(x(k))\}_{i\in\mathcal{I}}$, where each set is a polyhedron. It is easy to verify that these sets satisfy the conditions (12). For example, by restricting the 3 non-zero vectors into the cones indicated by the angles φ in Figure 2, it necessarily holds that at least two of these vectors are linearly independent. Hence, the full column rank condition (12b) is ensured.

Next we illustrate how to explicitly calculate the value of $c(k)$. Recall that $c(k) := \max_{i\in\mathcal{I}} \min_{z\neq 0} \min_{y\in\mathcal{P}_i(x(k))} \frac{|\langle y,z\rangle|}{\|y\|_2\|z\|_2}$ and for any $x, y \in \mathbb{R}^n$ the value $\frac{|\langle x,y\rangle|}{\|x\|_2\|y\|_2}$ defines the angle β between the two vectors [20]. More precisely $\beta = \cos^{-1}(\frac{|\langle x,y\rangle|}{\|x\|_2\|y\|_2})$, $0 \leq \beta \leq \frac{\pi}{2}$. As such, the value $c(k)$ is in fact the maximum of the smallest possible $\cos(\beta_i)$, where β_i denotes the angle between z and $y \in \mathcal{P}_i$. For the example of partition in Figure 2, we have that $c(k) = \cos(\varphi) = \cos(\frac{\pi}{3}) = 0.5$ for all $k \in \mathbb{Z}_+$ and thus, C can be taken equal to 0.5.

Next, we briefly describe an algorithm for constructing the sets $\{\mathcal{P}_i(x(k))\}_{i\in\mathcal{I}}$ as polyhedra, which consists of an off-line part and a very simple on-line adjustment procedure.

Off-line part: Construct an initial collection of polyhedral sets $\{\mathcal{P}_i^0\}_{i\in\mathcal{I}}$ for an arbitrary fixed $x(k) = x^0$ (for example, in Figure 2 we have chosen $x^0 := (1, 0)^\top$) such that (12) holds. Note that this is always possible. In particular observe that the condition (12b) is satisfied if and only if $p \geq n$ and there does not exist a hyperplane in $\mathbb{R}^n$ which contains the origin and intersects all the sets $\{\mathcal{P}_i^0\}_{i\in\mathcal{I}}$. Since each $\mathcal{P}_i^0$ is a polyhedron, there exist matrices H_i^0 and vectors h_i^0 such that $y \in \mathcal{P}_i^0 \Leftrightarrow H_i^0 y \leq h_i^0$ for all $i \in \mathcal{I}$ (for the example of Figure 2, $\mathcal{I} = \{1, 2, 3\}$).

On-line part: Let $x(k)$ be measured and let $\alpha(x(k))$ be the angle between $x(k)$ and x^0, see Figure 2 for a graphical illustration. Then construct $\{\mathcal{P}_i(x(k))\}_{i\in\mathcal{I}}$ as follows: $H_i(x(k)) := H_i^0 M(\alpha(x(k)))$ and $h_i(x(k)) = h_i^0$, where $y \in \mathcal{P}_i(x(k)) \Leftrightarrow H_i(x(k))y \leq h_i(x(k))$ for all $i \in \mathcal{I}$ and the matrix $M(\alpha)$ is a suitably defined rotational matrix [20], which can be chosen off-line. For the example of Figure 2, $M = \begin{pmatrix} \cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & \cos(\alpha) \end{pmatrix}$.

Remark 5. The on-line part mentioned above can be completely removed by replacing the terms $\rho[P(k)x(k)]_{i\bullet}$, $i \in \mathcal{I}$, in (17b) with the corresponding lower

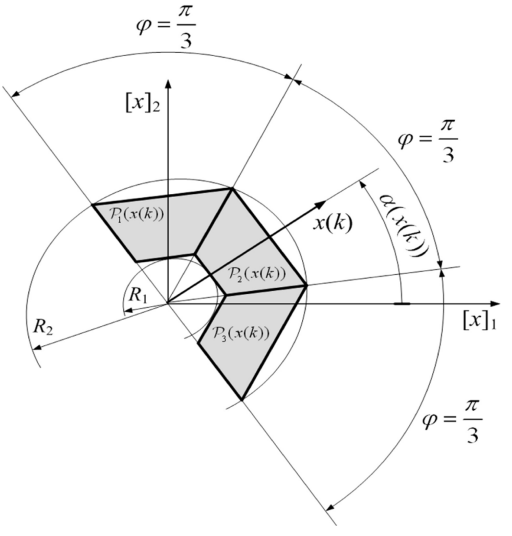


Fig. 2. Example of sets $\{\mathcal{P}_i(x(k))\}_{i\in\mathcal{I}}$ in the two dimensional case

bound. Also, observe that the derivation of $c(k)$ depends only on the off-line partition $\{\mathcal{P}_i^0\}_{i\in\mathcal{I}}$ and therefore, $c(k) = C$ can be fully determined off-line. □

Note that now Problem 2 can be formulated as a single linear program as follows. Constraint (17a) is linear as $\mathbb{X}$ and $\mathbb{U}$ are polyhedra and the system is affine in the control input. Constraint (17d) is now a set of linear constraints as each set $\mathcal{P}_i(x(k))$ is a polyhedron for the measured state $x(k)$. Furthermore, note that that for any matrix $Z \in \mathbb{R}^{r\times l}$ the condition $\|Z\|_\infty \leq c$ for some $c \in \mathbb{R}_{>0}$ is equivalent to $\pm[Z]_{i1} \pm [Z]_{i2} \ldots \pm [Z]_{il} \leq c$, $i = 1, \ldots, r$. Thus, as $x(k)$ is known at each $k \in \mathbb{Z}_+$, (17b), (17c) and (17e) can be specified through a finite number of linear inequalities in $\xi(k)$, $\tau(k)$ and in the elements of $P(k)$ without introducing any conservatism. Therefore, by Lemma 2, a solution to Problem 1 can be found by solving a single linear program at each sampling instant $k \in \mathbb{Z}_+$.

Remark 6. The hybrid nature of the system dynamics is inherently embedded in inequality (17b), which is equivalent to

$$\|\xi(k)f_j(x(k)) + g_j(x(k))\tau(k)\|_\infty - \rho[P(k)x(k)]_{i\bullet} \leq 0, \quad \forall i \in \mathcal{I}, \quad \text{if } x(k) \in \Omega_j.$$

However, as $x(k)$ is known at every time instant $k \in \mathbb{Z}_+$, it implies that the index $j \in \mathcal{S}$ is also known. That is why it is possible to solve Problem 2 by a single linear program (LP). Moreover, even in the case of mode uncertainty (possibly due to measurement noise), one can impose the above inequality in a robust way, i.e. for all dynamics indexed by $j \in \overline{\mathcal{S}}(x(k)) \subseteq \mathcal{S}$, where $\overline{\mathcal{S}}(x(k))$ collects all the indexes corresponding to the regions Ω_j that need to be accounted for in the case of mode uncertainty. Then, Problem 2 still can be formulated as a single linear program, while the method presented in Section 3.1 can be employed to decrease conservativeness considerably. □

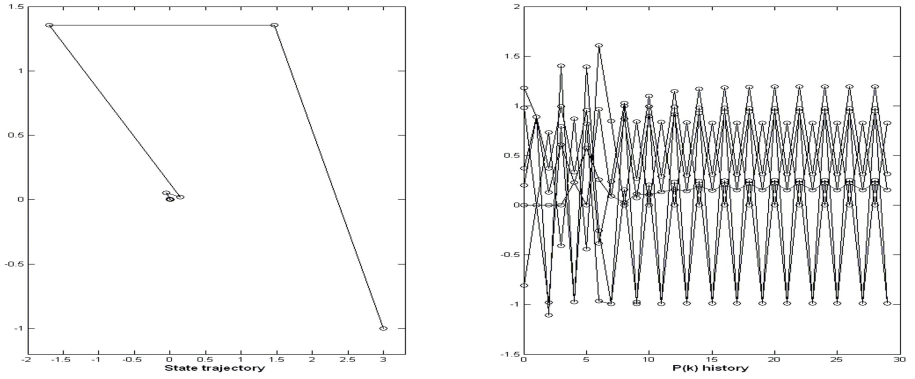


Fig. 3. Closed-loop simulation results: State-trajectory and $P(k)$ history

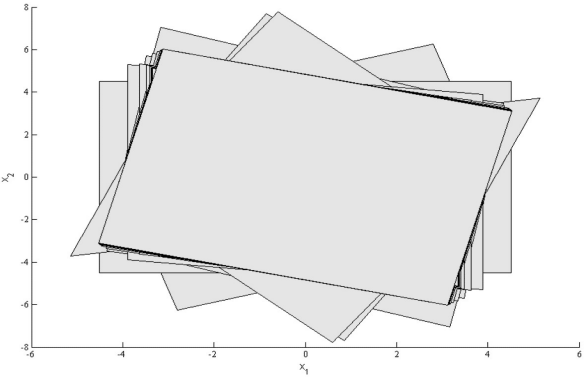


Fig. 4. Sublevel sets $\{x \mid V(k,x) \leq 4\}$ for $k \in \mathbb{Z}_{[0,30]}$

5 Illustrative Example

Consider system (1) with all the numerical details presented in Section 1.1. As stated therein, the synthesis problem for a PWQ Lyapunov function and a PWL state-feedback control law is not feasible unless the $\mathcal{S}$-procedure [10] is applied. This makes the synthesis of a stabilizing control law a challenging problem. We have followed the procedure described in the previous section to formulate Problem 2 as a single LP. We have fixed the dimensions of $P(k)$ to $\mathbb{R}^{3\times 2}$. The initial partition $\{\mathcal{P}_i^0\}_{i\in\mathcal{I}}$ has been chosen as illustrated in Figure 2, which yields $c(k) = C = 0.5$ for all $k \in \mathbb{Z}_+$. Furthermore, we have chosen the constants $R_1 = 1$ and $R_2 = 2$. The rotational matrix M for finding the matrices $H_i(x(k))$, $i = 1, 2, 3$, on-line is also taken as specified in Section 4.3. Finally, the tuning parameter ρ is taken equal to 0.9. The resulting LP has 8 optimization variables (the 6 elements of $P(k)$, $\xi(k)$ and $\tau(k)$) and 44 constraints. The time spent to calculate the matrices $H_i(x(k))$, which update certain constraints in the LP, is

negligible compared to the time required to solve the LP. The overall worst-case time required by the algorithm was less than 5 milliseconds.

Figure 3 shows the closed-loop simulation results for initial state $x(0) = [3 \ -1]^\top$. The proposed method successfully stabilizes the PWL system, while satisfying state and input constraints ($\mathbb{X} := \{x \in \mathbb{R}^2 \mid \|x\|_\infty \leq 5\}$ and $\mathbb{U} := \{u \in \mathbb{R} \mid |u| \leq 2\}$). Figure 4 presents a plot of the sublevel sets $\{x \mid V(k,x) = \|P(k)x\|_\infty \leq 4\}$ for $k \in \mathbb{Z}_{[0,30]}$. It is worth to point out that the closed-loop trajectory keeps on switching between two modes even very close to the origin, which in turn yields a different matrix $P(k)$ at two successive sampling instants. This can be observed in Figure 3, in the plot showing the history of all the 6 elements of $P(k)$, which still switch between two different values even when the state is very close to the origin. This demonstrates that the theoretical set-up proposed in this paper can effectively deal with non-trivial stabilization problems encountered in hybrid systems.

6 Conclusions

In this article we have proposed a new methodology that reduces significantly the conservatism of CLF design for discrete-time systems. Rather than searching for global CLFs off-line, we focused on synthesizing time-variant CLFs by solving on-line an optimization problem. As such, trajectory-dependent CLFs that are allowed to be locally non-monotone were derived. This approach offers a less conservative relaxation when compared to the classical $\mathcal{S}$-procedure approach. Regarding efficiency, we indicated that for PWC nonlinear systems affine in control and CLFs based on infinity norms, the on-line optimization problem can be formulated as a single linear program.

Acknowledgements. Research supported by the Veni grant "Flexible Lyapunov Functions for Real-time Control", grant number 10230, awarded by STW (Dutch Science Foundation) and NWO (The Netherlands Organization for Scientific Research).

References

1. Artstein, Z.: Stabilization with relaxed controls. Nonlinear Analysis 7, 1163–1173 (1983)
2. Sontag, E.D.: A Lyapunov-like characterization of asymptotic controllability. SIAM Journal of Control and Optimization 21, 462–471 (1983)
3. Sontag, E.D.: Stability and stabilization: Discontinuities and the effect of disturbances. In: Clarke, F.H., Stern, R.J. (eds.) Nonlinear Analysis, Differential Equations, and Control, pp. 551–598. Kluwer Academic Publishers, Dordrecht (1999)
4. Kokotović, P., Arcak, M.: Constructive nonlinear control: a historical perspective. Automatica 37(5), 637–662 (2001)
5. Grüne, L., Nesić, D.: Optimization based stabilization of sampled-data nonlinear systems via their approximate discrete-time models. SIAM Journal of Control and Optimization 42(1), 98–122 (2003)

6. Kellett, C.M., Teel, A.R.: Discrete-time asymptotic controllability implies smooth control-Lyapunov function. Systems & Control Letters 52, 349–359 (2004)
7. Jiang, Z.-P., Wang, Y.: Input-to-state stability for discrete-time nonlinear systems. Automatica 37, 857–869 (2001)
8. Lazar, M.: Model predictive control of hybrid systems: Stability and robustness. PhD thesis, Eindhoven University of Technology, The Netherlands (2006)
9. Branicky, M.S., Borkar, V.S., Mitter, S.K.: A unified framework for hybrid control: model and optimal control theory. IEEE Transactions on Automatic Control 43(1), 31–45 (1998)
10. Johansson, M., Rantzer, A.: Computation of piecewise quadratic Lyapunov functions for hybrid systems. IEEE Transactions on Automatic Control 43(4), 555–559 (1998)
11. Johansson, M.: Piecewise linear control systems. PhD thesis, Lund Institute of Technology, Sweden (1999)
12. Ferrari-Trecate, G., Cuzzola, F.A., Mignone, D., Morari, M.: Analysis of discrete-time piecewise affine and hybrid systems. Automatica 38(12), 2139–2146 (2002)
13. Daafouz, J., Riedinger, P., Iung, C.: Stability analysis and control synthesis for switched systems: A switched Lyapunov function approach. IEEE Transactions on Automatic Control 47, 1883–1887 (2002)
14. Di Cairano, S., Lazar, M., Bemporad, A., Heemels, W.P.M.H.: A Control Lyapunov Approach to Predictive Control of Hybrid Systems. In: Egerstedt, M., Mishra, B. (eds.) HSCC 2008. LNCS, vol. 4981, pp. 130–143. Springer, Heidelberg (2008)
15. Grieder, P., Kvasnica, M., Baotic, M., Morari, M.: Stabilizing low complexity feedback control of constrained piecewise affine systems. Automatica 41(10), 1683–1694 (2005)
16. Lazar, M., Heemels, W.P.M.H., Weiland, S., Bemporad, A.: Stabilizing model predictive control of hybrid systems. IEEE Transactions on Automatic Control 51(11), 1813–1818 (2006)
17. Jiang, Z.-P., Wang, Y.: A converse Lyapunov theorem for discrete-time systems with disturbances. Systems & Control Letters 45, 49–58 (2002)
18. Kellett, C.M., Teel, A.R.: On the robustness of $\mathcal{KL}$-stability for difference inclusions: Smooth discrete-time Lyapunov functions. SIAM Journal on Control and Optimization 44(3), 777–800 (2005)
19. Vorotnikov, V.I.: Partial stability and control. Birkhäuser, Boston (1998)
20. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, United Kingdom (1985)

Uniform Consensus among Self-driven Particles

Ji-Woong Lee

Department of Electrical Engineering
Pennsylvania State University
University Park, PA 16802
jiwoong@psu.edu

Abstract. A nonconservative stability theory for switched linear systems is applied to the convergence analysis of consensus algorithms in the discrete-time domain. It is shown that the uniform-joint-connectedness condition for asymptotic consensus in distributed asynchronous algorithms and multi-particle models is in fact necessary and sufficient for uniform exponential consensus.

1 Introduction

We consider teams of mobile agents working together towards the common goal of reaching consensus asymptotically [1,2]. These agents are often modeled as spatially distributed self-driven particles whose states (e.g., positions and velocities) evolve according to the information received from their neighbors. Each agent has its own neighbor set, and the collection of such neighbor sets over all agents determines a communication topology of a team. As the agents' states evolve, their neighbor sets are updated over time, and the team's communication topology undergoes changes as well. Since the number of agents is finite, the number of all possible communication topologies is finite. Therefore, as argued in [3], the behavior of these mobile agents can be described by a switched, or hybrid, dynamical system whose mode of operation jumps from one to another within a finite set according to the underlying, possibly nondeterministic, switching structure [4,5,6,7].

The purpose of this paper is to use switched system stability theory and establish a condition for teams of mobile agents to reach consensus in the discrete-time domain. Existing work in the literature [8,3,9,10] builds on Markov chain and Lyapunov stability theories. However, despite the apparent connection between the area of switched systems and that of multi-agent teams, not much work has been done at the intersection of the two areas. This is partly because seeking a common quadratic Lyapunov function does not work for the latter [3], which is discouraging, and because a relevant nonconservative stability analysis for the former was discovered only very recently [11,12,13]. This paper presents a convergence analysis that fully exploits the connection between switched systems and multi-agent models.

One of the nice things that comes from the use of switched system theory is that the notion of uniform exponential consensus arises as a natural notion of

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 252–261, 2009.

convergence. Uniform exponential consensus requires the existence of a single rate at which the agents' states converge to a common value regardless of the initial time. This uniformity requirement guarantees that asymptotic consensus will occur against a disturbance that causes a sudden change in the agents' states at an arbitrary time instant. This robustness property against disturbance is not guaranteed under the notion of mere asymptotic consensus. Moreover, our convergence condition is equivalent to a well-known sufficient condition for asymptotic consensus (i.e., the uniform-joint-connectedness condition in [3, Theorem 2]), which turns out to be not only sufficient but also necessary for uniform exponential consensus.

In summary, the novelty of this work lies in the following aspects:

- The connection between switched systems and multi-agent models is fully exploited;
- The common notion of asymptotic consensus is replaced with the stronger but more useful notion of uniform exponential consensus;
- The condition that the communication topology be uniformly jointly connected is shown to be an exact condition for uniform exponential consensus.

The main result is presented in Section 2, and its proof is given in Section 3. Concluding remarks are made in Section 4.

Notation

The n-dimensional real Euclidean space is denoted by $\mathbb{R}^n$. The Euclidean vector norm $\|\cdot\|$ on $\mathbb{R}^n$ is defined by $\|x\| = \sqrt{x^{\mathrm{T}}x}$ for $x \in \mathbb{R}^n$. The spectral norm on $\mathbb{R}^{n\times n}$ is denoted by $\|\cdot\|$ as well, and is defined by

$$\|\mathbf{X}\| = \sup\left\{\sqrt{\lambda}\colon \lambda \text{ is an eigenvalue of } \mathbf{X}^{\mathrm{T}}\mathbf{X}\right\}$$

for $\mathbf{X} \in \mathbb{R}^{n\times n}$. If $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^{n\times n}$ are symmetric (i.e., $\mathbf{X} = \mathbf{X}^{\mathrm{T}}$ and $\mathbf{Y} = \mathbf{Y}^{\mathrm{T}}$) and $\mathbf{X} - \mathbf{Y}$ is negative definite (i.e., $x^{\mathrm{T}}(\mathbf{X} - \mathbf{Y})x < 0$ whenever $x \neq 0$), we write either $\mathbf{X} < \mathbf{Y}$ or $\mathbf{X} - \mathbf{Y} < \mathbf{0}$.

2 Main Result

Let $\mathbb{S}$ be the set of all symmetric stochastic matrices in $\mathbb{R}^{n\times n}$ with positive diagonal entries. (That is, $\mathbf{F} = (f_{ij}) \in \mathbb{S}$ if and only if $f_{ij} = f_{ji}$, $f_{ij} \geq 0$, $f_{ii} > 0$, and $\sum_{k=1}^{n} f_{ik} = 1$ for all $i, j \in \{1, \ldots, n\}$.) Associated with each $\mathbf{F} = (f_{ij}) \in \mathbb{S}$ is a graph $G \subset \{1, \ldots, n\} \times \{1, \ldots, n\}$ such that $(i, j) \in G$ if and only if $f_{ij} > 0$ and $i \neq j$. (Note that these graphs are identified with sets of edges as they share the common set of vertices given by $\{1, \ldots, n\}$.)

Definition 1. *A graph $G \subset \{1, \ldots, n\} \times \{1, \ldots, n\}$ is said to be connected if between every pair of distinct vertices i, $j \in \{1, \ldots, n\}$ there exists a path $(i_0, i_1, \ldots, i_L) \in \{1, \ldots, n\}^{L+1}$ such that $i_0 = i$, $i_L = j$, and $(i_k, i_{k+1}) \in G$ for $k = 0, \ldots, L-1$. A set of graphs $\{G_j\colon j \in J\}$ is said to be jointly connected if its union $\bigcup_{j\in J} G_j$ is connected.*

A finite set

$$\mathcal{F} = \{\mathbf{F}_1, \ldots, \mathbf{F}_N\} \subset \mathbb{S} \tag{1}$$

defines a discrete linear inclusion (i.e., a discrete-time switched linear system under arbitrary switching) whose state-space representation is of the form

$$x(t+1) = \mathbf{F}_{\theta(t)} x(t) \tag{2}$$

for each switching sequence $\theta = (\theta(0), \theta(1), \ldots) \in \{1, \ldots, N\}^\infty$. For each $i \in \{1, \ldots, N\}$, let G_i be the graph associated with $\mathbf{F}_i$.

Definition 2. *Let $\mathcal{F}$ be as in (1). Let G_i be the graph associated with $\mathbf{F}_i$ for $i = 1, \ldots, N$. A switching sequence $\theta \in \{1, \ldots, N\}^\infty$ is said to yield uniformly jointly connected graphs if there exists an integer $T \geq 0$ such that the set of graphs $\{G_{\theta(t)}, \ldots, G_{\theta(t+T)}\}$ is jointly connected for all $t = 0, 1, \ldots$.*

Associated with each $\mathbf{F}_i \in \mathcal{F}$ is a unique matrix $\mathbf{A}_i \in \mathbb{R}^{(n-1)\times(n-1)}$ such that

$$\begin{bmatrix} 1 & \cdots & 0 & -1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & -1 \end{bmatrix} \mathbf{F}_i = \mathbf{A}_i \begin{bmatrix} 1 & \cdots & 0 & -1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & -1 \end{bmatrix}, \quad i = 1, \ldots, N.$$

Then

$$\mathcal{A} = \{\mathbf{A}_1, \ldots, \mathbf{A}_N\}$$

defines a discrete linear inclusion whose state-space description is given by

$$\hat{x}(t+1) = \mathbf{A}_{\theta(t)} \hat{x}(t) \tag{3}$$

for all switching sequences $\theta \in \{1, \ldots, N\}^\infty$. As argued in [3], the state equation (2) satisfies

$$\lim_{t\to\infty} x(t) = x_0 \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \tag{4}$$

for each $x(0) \in \mathbb{R}^n$, where $x_0 \in \mathbb{R}$ is a constant that depends on $x(0)$ (i.e., θ achieves asymptotic consensus for $\mathcal{F}$), if and only if the state equation (3) satisfies

$$\lim_{t\to\infty} \hat{x}(t) = 0 \tag{5}$$

for all $\hat{x}(0) \in \mathbb{R}^{n-1}$ (i.e., θ is asymptotically stabilizing for $\mathcal{A}$).

Definition 3. *Let $\mathcal{F}$ be as in (1). A switching sequence $\theta \in \{1, \ldots, N\}^\infty$ is said to achieve uniform exponential consensus for $\mathcal{F}$ if there exist $c > 0$ and $\lambda \in (0, 1)$ such that the state-space equation (3) satisfies*

$$\|\hat{x}(t)\| \leq c\lambda^{t-t_0} \|\hat{x}(t_0)\| \tag{6}$$

for all $t_0 \geq 0$, $t \geq t_0$, and $\hat{x}(t_0) \in \mathbb{R}^{n-1}$.

The following is the main result that establishes an exact condition under which a given switching sequence θ achieves uniform exponential consensus for $\mathcal{F}$.

Theorem 4. *Let $\mathcal{F}$ be as in (1). A switching sequence $\theta \in \{1, \ldots, N\}^\infty$ achieves uniform exponential consensus for $\mathcal{F}$ if and only if it yields uniformly jointly connected graphs.*

The proof of this theorem is deferred to Section 3. The result is applicable to a large class of distributed algorithms and multi-agent networks; e.g., some of the linear discrete-time consensus algorithms studied in [8,3,9,10]. In particular, Vicsek et al.'s multi-particle model [14] employs a nearest neighbor rule with parameter $r > 0$ for n agents moving at a common speed. Here, a real-valued state $x_i(t)$ of the i-th agent (i.e., the heading of the i-th agent) is updated according to

$$x_i(t+1) = \frac{1}{|N_i(t)|} \sum_{j \in N_i(t)} x_j(t), \quad i = 1, \ldots, N, \quad t = 0, 1, \ldots, \tag{7}$$

where

$$N_i(t) = \{j \in \{1, \ldots, n\} : \text{position of agent } j \text{ at time } t \text{ is within radius } r \text{ from position of agent } i \text{ at time } t,\ j = 1, \ldots, n\}$$

is the set of nearest neighbors of agent i (including agent i itself), and where $|N_i(t)|$ is the cardinality of $N_i(t)$. This update rule gives rise to a state equation of the form

$$x(t+1) = \mathbf{F}(t)x(t)$$

with $x(t) = [x_1(t) \ \cdots \ x_n(t)]^{\mathrm{T}}$ and $\mathbf{F}(t) \in \mathbb{S}$ for all t. Since the number N of distinct network topologies $\{N_1(t), \ldots, N_n(t)\}$ that can occur over all initial states $x(0) \in \mathbb{R}^n$ and over all time instants t is finite, we can label these topologies from 1 to N and obtain the state equation (2) with $\mathbf{F}(t) = \mathbf{F}_{\theta(t)}$, $\theta(t) \in \{1, \ldots, N\}$. Jadbabaie et al.'s sufficient condition [3] for asymptotic consensus states that, if there exists a τ and time instants $0 < t_1 < t_2 < \cdots$ such that $t_{k+1} - t_k \leq \tau$ for all k and such that the sets of graphs

$$\{G_{\theta(0)}, \ldots, G_{\theta(t_1-1)}\}, \quad \{G_{\theta(t_1)}, \ldots, G_{\theta(t_2-1)}\}, \quad \ldots$$

are all jointly connected, then the nearest neighbor rule (7) is guaranteed to yield asymptotic consensus; that is, all headings $x_i(t)$ approach a common value x_0 as $t \to \infty$. Putting $T = 2\tau$, this condition implies that the set $\{G_{\theta(t)}, \ldots, G_{\theta(t+T)}\}$ is jointly connected for all $t = 0, 1, \ldots$. Thus Theorem 4 asserts this sufficient condition for asymptotic consensus is in fact necessary and sufficient for uniform exponential consensus.

3 Proof of Main Result

3.1 Lemmas

There are a few lemmas required to prove Theorem 4. This subsection is devoted to summarizing them.

In the contexts of distributed asynchronous algorithms and multi-particle models, where each initial state leads to a deterministic switching sequence, it is known that asymptotic convergence of the state variables to a common value is guaranteed if the switching sequence yields uniformly jointly connected graphs [8,14,3,15].

Lemma 5. *If the switching sequence θ yields uniformly jointly connected graphs, then the state equation (2) satisfies (4) for each $x(0) \in \mathbb{R}^n$, where $x_0 \in \mathbb{R}$ is a constant that depends on $x(0)$.*

Proof. The result is due to $\mathcal{F}$ being a finite subset of $\mathbb{S}$. See, e.g., [3, Theorem 2].

On the other hand, in the context of discrete inclusions and switched systems under nondeterministic switching, it is known that the discrete linear inclusion $\mathcal{A}$ is asymptotically stable under arbitrary switching if and only if the generalized spectral radius of $\mathcal{A}$ is less than one, or equivalently, there exists a sub-multiplicative norm $\|\cdot\|_{\mathcal{A}}$ such that $\|\mathbf{A}_i\|_{\mathcal{A}} < 1$ for all $i \in \{1, \ldots, N\}$ [16,17,18,19]. The following lemma is a simple consequence of this, and says that asymptotically stable discrete linear inclusions are in fact uniformly exponentially stable.

Lemma 6. *The state equation (3) satisfies (5) for all $\hat{x}(0) \in \mathbb{R}^{n-1}$ and $\theta \in \{1, \ldots, N\}^\infty$ if and only if there exist $c > 0$ and $\lambda \in (0, 1)$ such that (3) satisfies (6) for all $t_0 \geq 0$, $t \geq t_0$, $\hat{x}(t_0) \in \mathbb{R}^{n-1}$, and $\theta \in \{1, \ldots, N\}^\infty$.*

Proof. In fact, the result holds for any finite subset $\mathcal{A}$ of $\mathbb{R}^{(n-1)\times(n-1)}$. See, e.g., [11, Proposition 8].

Recent advances in the stability analysis of discrete-time switched linear systems give a characterization of uniformly exponentially stabilizing switching sequences. This characterization plays a crucial role in establishing our result, and hence is described here. For each integer $L \geq 0$, tuples of integers of the form $(i_0, \ldots, i_L) \in \{1, \ldots, N\}^{L+1}$ are called L-paths. Following the terminology used in [13], a finite set $\mathcal{N}$ of L-paths shall be said to be admissible if for each $(i_0, \ldots, i_L) \in \mathcal{N}$ there exists an integer $M > 0$ such that $(i_0, \ldots, i_L) = (i_{M-L}, \ldots, i_M)$ and such that $(i_t, \ldots, i_{t+L}) \in \mathcal{N}$ for all $t = 0, \ldots, M - L$. Likewise, an admissible set $\mathcal{N}$ of L-paths shall be called $\mathcal{A}$-admissible if there exist symmetric positive definite matrices $\mathbf{X}_{(j_1,\ldots,j_L)} \in \mathbb{R}^{(n-1)\times(n-1)}$ satisfying the coupled Lyapunov inequalities

$$\mathbf{A}_{i_L}^{\mathrm{T}} \mathbf{X}_{(i_1,\ldots,i_L)} \mathbf{A}_{i_L} - \mathbf{X}_{(i_0,\ldots,i_{L-1})} < \mathbf{0}$$

for all L-paths $(i_0, \ldots, i_L) \in \mathcal{N}$. Given a switching sequence $\theta \in \{1, \ldots, N\}^\infty$ and an integer $L \geq 0$, let $\mathcal{N}_L(\theta)$ be the largest admissible subset of

$$\{(\theta(0), \ldots, \theta(L)), (\theta(1), \ldots, \theta(L+1)), \ldots\}.$$

Then we have the following result:

Lemma 7. *There exist $c > 0$ and $\lambda \in (0,1)$ such that the state equation (3) satisfies (6) for all $t_0 \geq 0$, $t \geq t_0$, and $\hat{x}(t_0) \in \mathbb{R}^{n-1}$, if and only if there exists an integer $L \geq 0$ such that $\mathcal{N}_L(\theta)$ is $\mathcal{A}$-admissible.*

Proof. As in the proof of Lemma 6, the result holds for any finite subset $\mathcal{A}$ of $\mathbb{R}^{(n-1)\times(n-1)}$. See [12, Corollary 3.4].

Suppose $\mathcal{N}$ is an $\mathcal{A}$-admissible set of L-paths. If the smallest $\mathcal{A}$-admissible subset of $\mathcal{N}$ is $\mathcal{N}$ itself, then $\mathcal{N}$ is called $\mathcal{A}$-minimal. As argued in [13], associated with each $\mathcal{A}$-minimal set of L-paths is a periodic uniformly exponentially stabilizing switching sequence for $\mathcal{A}$; moreover, each $\mathcal{A}$-admissible set is a finite union of $\mathcal{A}$-minimal sets. For switching sequences $\theta \in \{1,\dots,N\}^\infty$ and integers $L \geq 0$, define $\mathcal{N}_L^\infty(\theta)$ as the set of L-paths $(i_0,\dots,i_L)$ such that for any $t_0 \geq 0$ there exists a $t > t_0$ satisfying $(\theta(t),\dots,\theta(t+L)) = (i_0,\dots,i_L)$. Then $\mathcal{N}_L^\infty(\theta)$ contains the L-paths that occur infinitely many times in θ; it is nonempty because the set $\{1,\dots,N\}^{L+1}$ of all L-paths is finite. In summary, we have the following lemma:

Lemma 8. *Suppose that there exists an integer $L \geq 0$ such that $\mathcal{N}_L(\theta)$ is $\mathcal{A}$-admissible. Then the following hold:*

(a) The set $\mathcal{N}_L^\infty(\theta)$ is $\mathcal{A}$-admissible and is identical to $\mathcal{N}_L((\theta(t_0),\theta(t_0+1),\dots))$ for some integer $t_0 \geq 0$.
(b) The set $\mathcal{N}_L^\infty(\theta)$ is a finite union of $\mathcal{A}$-minimal sets of L-paths.

Proof. Part (b) is an immediate consequence of part (a), so it suffices to show part (a) holds true. Suppose $\mathcal{N}_L^\infty(\theta)$ is not admissible. Then, there exists an L-path $(i_0,\dots,i_L) \in \mathcal{N}_L^\infty(\theta)$ such that, whenever $M > 0$ and $i_{L+1}, \dots, i_M \in \{1,\dots,N\}$ satisfy $(i_{M-L},\dots,i_M) = (i_0,\dots,i_L)$, there exists a $t \in \{0,\dots,M-L\}$ such that $(i_t,\dots,i_{t+L})$ does not belong to $\mathcal{N}_L^\infty(\theta)$. That is, whenever we form a cycle of L-paths that contains $(i_0,\dots,i_L)$, the cycle contains an L-path that does not occur infinitely many times in θ. Therefore, $(i_0,\dots,i_L)$ cannot occur infinitely many times in θ. This contradicts the fact that $(i_0,\dots,i_L) \in \mathcal{N}_L^\infty(\theta)$. Thus $\mathcal{N}_L^\infty(\theta)$ is admissible. Moreover, $\mathcal{N}_L^\infty(\theta)$ is $\mathcal{A}$-admissible because $\mathcal{N}_L^\infty(\theta)$ is an admissible subset of $\mathcal{N}_L(\theta)$, which is $\mathcal{A}$-admissible. To complete the proof, it remains to show that $\mathcal{N}_L^\infty(\theta) = \mathcal{N}_L((\theta(t_0),\theta(t_0+1),\dots))$ for some $t_0 \geq 0$. Since $\mathcal{N}_L(\theta)$ is finite, the set difference $\mathcal{N}_L(\theta) \setminus \mathcal{N}_L^\infty(\theta)$ is finite. For each $(i_0,\dots,i_L)$ in $\mathcal{N}_L(\theta) \setminus \mathcal{N}_L^\infty(\theta)$, let τ be the largest integer such that $(\theta(\tau-1),\dots,\theta(\tau+L-1)) = (i_0,\dots,i_L)$. Then letting t_0 be the maximum of such τ's over all L-paths in the finite set $\mathcal{N}_L(\theta) \setminus \mathcal{N}_L^\infty(\theta)$ yields the desired result.

3.2 Sufficiency

To prove sufficiency of Theorem 4, suppose a switching sequence θ yields uniformly jointly connected graphs. If G_i are the graphs associated with $\mathbf{F}_i$ for $i = 1, \dots, N$, then there exists a $T \geq 0$ such that $\{G_{\theta(t)},\dots,G_{\theta(t+T)}\}$ is jointly connected for all $t = 0, 1, \dots$. Given such a T, define

$$\mathcal{S} = \left\{ (i_0,\dots,i_T) \in \{1,\dots,N\}^{T+1} \colon \bigcup\nolimits_{t=0}^{T} G_{i_t} \text{ is connected} \right\},$$

so that $\mathcal{S}$ is the set of all T-paths over which the associated graphs are jointly connected. Define

$$\widetilde{\mathbf{F}}_{(i_0,\dots,i_T)} = \mathbf{F}_{i_T}\cdots\mathbf{F}_{i_0} \quad \text{and} \quad \widetilde{\mathbf{A}}_{(i_0,\dots,i_T)} = \mathbf{A}_{i_T}\cdots\mathbf{A}_{i_0}$$

for $(i_0,\dots,i_T)\in\mathcal{S}$, and let

$$\widetilde{\mathcal{F}} = \{\widetilde{\mathbf{F}}_{(i_0,\dots,i_T)}\colon (i_0,\dots,i_T)\in\mathcal{S}\},$$
$$\widetilde{\mathcal{A}} = \{\widetilde{\mathbf{A}}_{(i_0,\dots,i_T)}\colon (i_0,\dots,i_T)\in\mathcal{S}\}.$$

By construction, $\widetilde{\mathcal{F}}$ forms a discrete linear inclusion whose elements $\widetilde{\mathbf{F}}_{(i_0,\dots,i_T)}$ are associated with connected graphs

$$\widetilde{G}_{(i_0,\dots,i_T)} = \bigcup_{t=0}^{T} G_{i_t}, \quad (i_0,\dots,i_T)\in\mathcal{S}.$$

By Lemma 5 we have that, for every sequence of T-paths $\tilde\theta = (\tilde\theta(0),\tilde\theta(1),\dots)$ such that $\tilde\theta(t)\in\mathcal{S}$, $t=0,1,\dots$, the state equation

$$\bar x(t+1) = \widetilde{\mathbf{F}}_{\tilde\theta(t)}\bar x(t)$$

satisfies $\lim_{t\to\infty}\bar x(t) = \bar x_0\,[1\ \cdots\ 1]^{\mathrm{T}}$ for each $\bar x(0)\in\mathbb{R}^n$, with some constant $\bar x_0$ depending on $\bar x(0)$. That is, the state equation

$$\tilde x(t+1) = \widetilde{\mathbf{A}}_{\tilde\theta(t)}\tilde x(t) \tag{8}$$

satisfies $\lim_{t\to\infty}\tilde x(t) = 0$ for all $\tilde x(0)\in\mathbb{R}^{n-1}$ and for all $\tilde\theta = (\tilde\theta(0),\tilde\theta(1),\dots)$ with $\tilde\theta(t)\in\mathcal{S}$, $t=0,1,\dots$. Then, by Lemma 6, there exist $\tilde c>0$ and $\tilde\lambda\in(0,1)$ such that the state equation (8) satisfies $\|\tilde x(t)\|\le\tilde c\tilde\lambda^{t-t_0}\|\tilde x(t_0)\|$ for all $t_0\ge 0$, $t\ge t_0$, $\tilde x(t_0)\in\mathbb{R}^{n-1}$, and $\tilde\theta = (\tilde\theta(0),\tilde\theta(1),\dots)$ with $\tilde\theta(s)\in\mathcal{S}$, $s=0,1,\dots$. In particular, the given switching sequence $\theta = (\theta(0),\theta(1),\dots)$ can be identified with a sequence of T-paths $\tilde\theta = (\tilde\theta(0),\tilde\theta(1),\dots)$ via

$$\tilde\theta(t) = \big(\theta(t(T+1)),\dots,\theta(t(T+1)+T)\big), \quad t=0,1,\dots,$$

and it yields a state equation of the form (3) that satisfies

$$\|\hat x(\tau(T+1))\|\le\tilde c\tilde\lambda^{\tau-\tau_0}\|\hat x(\tau_0(T+1))\| \tag{9}$$

whenever $\tau\ge\tau_0\ge 0$ and $\hat x(\tau_0(T+1))\in\mathbb{R}^{n-1}$.

It remains to convert (9) to an inequality of the form (6). Let $\lambda\in(0,1)$ be such that $\tilde\lambda = \lambda^{T+1}$, and let $M = \max_{1\le i\le N}\|\mathbf{A}_i\|/\lambda$. Whenever $t\ge t_0\ge 0$, let τ be the largest integer such that $t\ge\tau(T+1)$, and let τ_0 be the smallest integer such that $\tau_0(T+1)\ge t_0$. Then it follows from (9) that

$$\|\hat x(t)\|\le\begin{cases} M^{t-t_0}\lambda^{t-t_0}\|\hat x(t_0)\| & \text{if } \tau_0>\tau;\\ \tilde c M^{(t-\tau(T+1))+(\tau_0(T+1)-t_0)}\lambda^{t-t_0}\|\hat x(t_0)\| & \text{if } \tau_0\le\tau.\end{cases}$$

If $\tau_0 > \tau$, then $t - t_0 \le T$. Similarly, if $\tau_0 \le \tau$, then $t - \tau(T+1) \le T$ and $\tau_0(T+1) - t_0 \le T$. Thus

$$\|\hat{x}(t)\| \le \begin{cases} \max\{1, M\}^T \lambda^{t-t_0} \|\hat{x}(t_0)\| & \text{if } \tau_0 > \tau; \\ \tilde{c} \max\{1, M\}^{2T} \lambda^{t-t_0} \|\hat{x}(t_0)\| & \text{if } \tau_0 \le \tau. \end{cases}$$

Now, letting $c = \max\{1, \tilde{c}\} \max\{1, M\}^{2T}$ yields that (6) holds for all $t_0 \ge 0$, $t \ge t_0$, and $\hat{x}(t_0) \in \mathbb{R}^{n-1}$. Therefore, θ achieves uniform exponential consensus for $\mathcal{F}$. This completes the proof of the sufficiency part of Theorem 4.

3.3 Necessity

To prove necessity of Theorem 4, suppose a switching sequence θ achieves uniform exponential consensus for $\mathcal{F}$. Then the state equation (3) satisfies (6) whenever $t \ge t_0 \ge 0$ and $\hat{x}(t_0) \in \mathbb{R}^{n-1}$. By Lemma 7 there exists a nonnegative integer L such that $\mathcal{N}_L(\theta)$ is $\mathcal{A}$-admissible, and hence by Lemma 8 the set $\mathcal{N}_L^\infty(\theta)$ is $\mathcal{A}$-admissible and is a finite union of $\mathcal{A}$-minimal sets of L-paths.

Choose an $\mathcal{A}$-minimal set $\mathcal{N}_{\min}$ of L-paths and the associated periodic switching sequence

$$\theta_{\min} = (i_0, \dots, i_M, i_0, \dots, i_M, \dots),$$

where the period $M+1$ equals the cardinality of $\mathcal{N}_{\min}$. We will first show that $\theta_{\min}$ yields uniformly jointly connected graphs. Since $\mathcal{N}_{\min}$ is an $\mathcal{A}$-admissible set of L-paths, by Lemma 7 there exist $c > 0$ and $\lambda \in (0, 1)$ such that the state equation (3), with θ replaced by $\theta_{\min}$, satisfies (6) whenever $t \ge t_0 \ge 0$ and $\hat{x}(t_0) \in \mathbb{R}^{n-1}$. That is, $\theta_{\min}$ achieves uniform exponential consensus for $\mathcal{F}$. Suppose $\theta_{\min}$ does not yield uniformly jointly connected graphs. Then, since $\theta_{\min}$ is periodic with period $M+1$, we have that the union $G = \bigcup_{t=0}^{M} G_{i_t}$, where G_i is the graph of $\mathcal{F}_i$, is not connected. That is, we can partition the set of vertices $\{1, \dots, n\}$ into two disjoint sets $V_1, V_2 \subset \{1, \dots, n\}$ such that $(i, j) \notin G$ whenever $(i, j) \in V_1 \times V_2$. Now, choose two distinct $x_1, x_2 \in \mathbb{R}$, and let $x(0) = (x_1(0), \dots, x_n(0)) \in \mathbb{R}^n$ be such that

$$x_i(0) = \begin{cases} x_1 & \text{if } i \in V_1; \\ x_2 & \text{if } i \in V_2. \end{cases}$$

Because V_1 and V_2 remain disconnected under $\theta_{\min}$, and because the matrices $\mathbf{F}_i$ are stochastic, the state equation (2) will have that $x(t) = x(0)$ for all t under $\theta_{\min}$. This contradicts $\theta_{\min}$ achieving uniform exponential consensus for $\mathcal{F}$, and hence proves that $\theta_{\min}$ indeed yields uniformly jointly connected graphs.

Now that we have shown each $\mathcal{A}$-minimal set leads to a periodic switching sequence that yields uniformly jointly connected graphs, we are ready to show that the given θ, which achieves uniform exponential consensus for $\mathcal{F}$, yields uniformly jointly connected graphs. Let τ be the cardinality of $\mathcal{N}_L^\infty(\theta)$. By Lemma 8, there exists a t_0 such that, for each $t \ge t_0$, there exists an L-path $(i_0, \dots, i_L)$ that occur more than once in the switching path $(\theta(t), \dots, \theta(t+\tau+L))$; that is, for some t_1, $t_2 \in \{t, \dots, t+\tau\}$ such that $t_1 < t_2$, we have

$$(\theta(t_1), \dots, \theta(t_1+L)) = (\theta(t_2), \dots, \theta(t_2+L)) = (i_0, \dots, i_L).$$

Then it is clear that the set

$$\mathcal{N} = \{(\theta(t_1), \ldots, \theta(t_1 + L)), \ldots, (\theta(t_2 - 1), \ldots, \theta(t_2 + L - 1))\} \qquad (10)$$

forms an $\mathcal{A}$-admissible set of L-paths. Since $\mathcal{N}$ contains an $\mathcal{A}$-minimal set of L-paths, we have that the union $\bigcup_{t=t_1}^{t_2-1} G_{\theta(t)}$ is connected. This is true for each $t \geq t_0$, and so the union $\bigcup_{s=t}^{t+\tau-1} G_{\theta(s)}$ is connected for all $t \geq t_0$. Therefore, putting $T = t_0 + \tau$ gives that the set of graphs $\{G_{\theta(t)}, \ldots, G_{\theta(t+T)}\}$ is jointly connected for all $t = 0, 1, \ldots$. This concludes the proof of the necessity part of Theorem 4.

4 Conclusions

Multi-agent consensus algorithms were studied via a nonconservative stability theory for switched systems, and a well-known sufficient condition for asymptotic consensus was shown to be necessary and sufficient for uniform exponential consensus. Possible extensions of this work include consideration of more general classes of consensus algorithms and incorporation of the state-dependent switching structure.

Acknowledgment

The author would like to thank Supratim Ghosh, discussions with whom inspired this work.

References

1. Olfati-Saber, R., Fax, J.A., Murray, R.M.: Consensus and cooperation in networked multi-agent systems. Proceedings of the IEEE 95(1), 215–233 (2007)
2. Ren, W., Beard, R.W., Atkins, E.M.: Information consensus in multivehicle cooperative control. IEEE Control Systems Magazine 27(2), 71–82 (2007)
3. Jadbabaie, A., Lin, J., Morse, A.S.: Coordination of groups of mobile autonomous agents using nearest neighbor rules. IEEE Transactions on Automatic Control 48(6), 988–1001 (2003)
4. Branicky, M.S.: Multiple Lyapunov functions and other analysis tools for switched and hybrid systems. IEEE Transactions on Automatic Control 43(4), 475–482 (1998)
5. Liberzon, D., Morse, A.S.: Basic problems in stability and design of switched systems. Control Systems Magazine 19(5), 59–70 (1999)
6. DeCarlo, R.A., Branicky, M.S., Pettersson, S., Lennartson, B.: Perspectives and results on the stability and stabilizability of hybrid systems. Proceedings of the IEEE 88(7), 1069–1082 (2000)
7. Sun, Z., Ge, S.S.: Analysis and synthesis of switched linear control systems. Automatica 41(2), 181–195 (2005)
8. Tsitsiklis, J.N., Bertsekas, D.P., Athans, M.: Distributed asynchronous deterministic and stochastic gradient optimization algorithms. IEEE Transactions on Automatic Control 31(9), 803–812 (1986)

9. Ren, W., Beard, R.W.: Consensus seeking in multiagent systems under dynamically changing interaction topologies. IEEE Transactions on Automatic Control 50(5), 655–661 (2005)
10. Moreau, L.: Stability of multiagent systems with time-dependent communication links. IEEE Transactions on Automatic Control 50(2), 169–182 (2005)
11. Lee, J.W., Dullerud, G.E.: Uniform stabilization of discrete-time switched and Markovian jump linear systems. Automatica 42(2), 205–218 (2006)
12. Lee, J.W., Dullerud, G.E.: Optimal disturbance attenuation for discrete-time switched and Markovian jump linear systems. SIAM Journal on Control and Optimization 45(4), 1329–1358 (2006)
13. Lee, J.W., Dullerud, G.E.: Uniformly stabilizing sets of switching sequences for switched linear systems. IEEE Transactions on Automatic Control 52(5), 868–874 (2007)
14. Vicsek, T., Czirók, A., Ben-Jacob, E., Cohen, I., Shochet, O.: Novel type of phase transition in a system of self-driven particles. Physical Review Letters 75(6), 1226–1229 (1995)
15. Bertsekas, D.P., Tsitsiklis, J.N.: Comments on coordination of groups of mobile autonomous agents using nearest neighbor rules. IEEE Transactions on Automatic Control 52(5), 968–969 (2007)
16. Daubechies, I., Lagarias, J.C.: Sets of matrices all infinite products of which converge. Linear Algebra and its Applications 161, 227–263 (1992)
17. Berger, M.A., Wang, Y.: Bounded semigroups of matrices. Linear Algebra and its Applications 166, 21–27 (1992)
18. Gurvits, L.: Stability of discrete linear inclusion. Linear Algebra and its Applications 231, 47–85 (1995)
19. Daubechies, I., Lagarias, J.C.: Corrigendum/addendum to: Sets of matrices all infinite products of which converge. Linear Algebra and its Applications 327, 69–83 (2001)

Optimization of Multi-agent Motion Programs with Applications to Robotic Marionettes

Patrick Martin and Magnus Egerstedt

Georgia Institute of Technology, Atlanta, GA 30332, USA
{pmartin,magnus}@ece.gatech.edu

Abstract. In this paper, we consider the problem of generating optimized, executable control code from high-level, symbolic specifications. In particular, we construct symbolic control programs using strings from a motion description language with a nominal set of motion parameters, such as temporal duration and energy, embedded within each mode. These parameters are then optimized over, using tools from optimal switch-time control and decentralized optimization of separable network problems. The resulting methodology is applied to the problem of controlling robotic marionettes, and we demonstrate its operation on an example scenario involving symbolic puppet plays defined for multiple puppets.

1 Introduction

In order to manage the complexity associated with many emerging controls applications, various abstraction-based formalisms have been advanced for specifying, modeling, and controlling such systems. Examples include linear temporal logic specifications (e.g. [13,20]), Maneuver Automata for capturing symmetries [10,11], and Motion Description Languages (MDL) for symbolic control (e.g. [6,7,14,16]). These different formalisms have been designed with alternative goals in mind. As such, they have different strengths, but common to them all is that they use varying degrees of abstraction to achieve desired levels of control code granularity [5]. However, there is always a choice to be made when mapping these high-level programs onto executable code. This mapping is the main question under consideration in this paper. In particular, we investigate how to turn such high-level control descriptions into *optimized*, *executable* low-level control software modules, or *control code*, for a particular hardware platform.

In this paper we choose the MDL framework, as originally formulated in [6], to break up the control task into "strings" of individual controller-interrupt pairs. However, we use a slightly modified MDL structure for the motion programs in that they support energy parameterized motions as well as novel spatio-temporal motion constraints. In particular, this work is applied to the problem of robotic puppetry. Puppeteers script plays that designate a string of motions for each character within a structured environment; consequently, the use of MDL strings for *specifying* plays is natural as observed in [8]. As such we script plays using the MDL formalism and take the resulting nominal symbolic descriptions of the play and generate optimized, executable programs based on the system dynamics and an associated cost criterion.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 262–275, 2009.

The resulting optimization problem is not unique to puppetry, since MDL-based abstractions of hybrid systems may need to optimize their motion programs in order to account for system dynamics and constraints in a number of other applications. We approach the solution to this problem by drawing from recent results in switched-time optimization [3,9,19,21], focusing on the scheduling of discrete transitions in a hybrid system by adjusting the timing parameters or mode order of the program.

This paper expands previous work on robotic puppetry [8,15] by fully incorporating spatial and temporal constraints into the hybrid optimization engine. We do so by applying classical results in separable programming to generate an algorithm for hybrid optimization under *networked* constraints. The result of this effort is a tool (the *MDL compiler*) that is able to accept MDL strings for a collection of puppets and generate optimal timing and energy parameters under temporal and spatial constraints. Moreover, we validate this MDL compilation framework with numerical simulations involving multiple puppets with spatial and temporal constraints.

The remainder of this paper is organized as follows: In Section 2 we introduce the MDL structure and derive an optimal control-based MDL compiler. Section 3 showcases the application of this MDL compiler by optimizing motion programs involving multiple agents and spatial constraints. Section 4 discusses an application of decentralized nonlinear programming techniques for handling motion programs with timing constraints between agents. We conclude with a brief summary in Section 5.

2 Background

In this section we discuss the background work for generating control code from high-level specifications. Figure 1 illustrates the general flow of this control code generation process. In particular we modify the "standard" MDL formalism to enable the specification of motion programs for a collection of agents typically encountered in puppetry. Furthermore, we derive the necessary optimality conditions for the program's switch times and scaling parameters, which are then used in the *MDL Compiler* block. Finally, we illustrate the application of this MDL compiler for the special case of specifying puppet motion programs.

2.1 Motion Description Language Compiler

In order to script a motion program we describe a special MDL that accounts for four important properties of multi-agent motion programs: *who* should act,

Fig. 1. An illustration of the process of turning high-level MDL programs into executable control code

what motion should they do, *where* should they operate, and *when* should the action occur. We assume that the agents are identified by $i \in \mathcal{M}$, where $\mathcal{M} = \{1, \cdots, m\}$, and each agent has the dynamics,

$$\dot{x}^i = f(x^i, u^i),\ x^i \in \mathbb{R}^n,\ u^i \in \mathbb{R}^p, \tag{1}$$

where we use the superscript i to denote agent i.

We define the input to this model as one in a collection of possible feedback laws, i.e. $u^i = \kappa_j(x^i, t, \alpha_j)$, with κ_j, for some j, coming from a finite set of control laws $\mathcal{K} = \{\kappa_1, \cdots, \kappa_C\}$; additionally, α_j is an "energy"-scaling parameter that could affect speed, amplitude, or some other property of the control mode. When applying a controller of this form, we get the resulting closed-loop system dynamics $\dot{x}^i = f(x^i, t, \kappa_j(x^i, t, \alpha_j))$.

"Standard" MDL combines the controllers from $\mathcal{K}$ with a time-driven interrupt, denoted τ, that dictates the time at which the control mode interrupts, resulting in controller-interrupt pairs of the form (κ, τ). However, to allow for the specification of multi-agent programs, we add in an element for agent identification, i, and a spatially defined location, r, where the agent performs its control κ. These locations in the environment come from a set $\mathcal{R} = \{r_1, \cdots, r_l\}$. Using these additional elements, we thus define our multi-agent MDL mode as the tuple (i, κ, r, τ).

For example, if agent-i is using the two mode MDL string

$$(i, \kappa_1(\alpha_1), r_1, \tau_1)(i, \kappa_2(\alpha_2), r_1, \tau_2)$$

it must complete the motion κ_1, scaled by α_1, within region r_1 until time τ_1. (Note here that even though κ_j is a function of x^i, t, and α_j, we specify it symbolically through the α_j dependency alone.) Once this mode terminates, the second mode will execute κ_2 with scaling α_2, also in region r_1, until τ_2, which in this case signals the end of the play.

Now that we have modified MDL for composing multi-agent motion programs, we focus on developing a process for tweaking the timing and scaling parameters. For instance, an undesirable MDL mode would use a control law that potentially drives the system out of its intended region. It would be better to adjust the timing and scaling of the mode so that this is prevented. We approach this problem using calculus of variations to design a MDL compiler that accepts a nominal motion program and outputs control code based on the system dynamics, under spatio-temporal constraints.

Say we are given a single-agent (agent i) program with N modes over the time interval $[t_0, t_f]$, and we denote all switch time parameters as the vector $\bar{\tau}^i = [\tau_1^i\ \cdots\ \tau_{N-1}^i]$ and the scaling parameters as $\bar{\alpha}^i = [\alpha_1^i\ \cdots\ \alpha_N^i]$. Then the cost functional for optimizing this agent's program could take the form,

$$\min_{\bar{\tau}^i, \bar{\alpha}^i} J(\bar{\tau}^i, \bar{\alpha}^i) = \int_{t_0}^{t_f} L(x^i, t)dt + \sum_{j=1}^{N} C_j(\alpha_j^i) + \sum_{k=1}^{N-1} \left(\Psi_k(x^i(\tau_k^i)) + \Delta_k(\tau_k^i)\right). \tag{2}$$

The interpretation here is that the agent has a trajectory cost, $L(x^i, t)$, associated with the execution of the motion program. Since scaling controller speed or

amplitude requires more energy, we penalize the energy usage of each mode with the $C_j(\alpha_j^i)$ functions. We also encode the spatial constraint for each mode through the spatial cost term, $\Psi_k(x^i(\tau_k^i))$, that penalizes the distance of the agent from the location of the specified region. Finally, to prevent large deviations of a particular switch-time τ_k^i, we add the temporal cost function $\Delta_k(\tau_k^i)$.

To determine the first order necessary optimality conditions, we perturb all switch times and energy parameters as $\tau_k^i \to \tau_k^i + \varepsilon\theta_k^i$ and $\alpha_k^i \to \alpha_k^i + \varepsilon a_k^i$. In [8], the derivation for a two mode program was given and we generalize this without proof since it is a direct generalization of the of the derivation in [8]:

$$\begin{aligned}
&\frac{\partial J}{\partial \tau_k^i} = \lambda^i(\tau_k^{i-}) f_k(x^(\tau_k^i)) - \lambda^i(\tau_k^{i+}) f_{k+1}(x^i(\tau_k^i)) + \frac{\partial \Delta_k^i}{\partial \tau_i} = 0,\ k = 1, \cdots, N-1 \\
&\frac{\partial J}{\partial \alpha_k^i} = \mu^i(\tau_{k-1}^{i+}) = 0,\ k = 1, \cdots, N
\end{aligned} \tag{3}$$

where we use the short-hand notation $f_k(x^i(t))$ to denote $f(x^i, t, \kappa_k(x^i, t, \alpha_k^i))$ and where τ_k^{i-} and τ_k^{i+} are the left and right limits, respectively. Moreover, the discontinuous costates (λ^i, μ^i) satisfy the costate dynamics,

$$\begin{aligned}
\dot{\lambda}^i &= -\frac{\partial J}{\partial x^i} - \lambda^i \frac{\partial f_k}{\partial x^i}, t \in (\tau_{k-1}, \tau_k) \\
\dot{\mu}^i &= \lambda^i \frac{\partial f_k}{\partial x^i}
\end{aligned}$$

and boundary conditions,

$$\begin{aligned}
\lambda^i(\tau_N^i) &= \frac{\partial \Psi_N}{\partial x^i}(x^i(\tau_N^i)) \\
\mu^i(\tau_N^i) &= \frac{\partial C_N}{\partial \alpha_N^i} \\
\lambda^i(\tau_k^{i-}) &= \lambda^i(\tau_k^{i+}) + \frac{\partial \Psi_k}{\partial x^i}(x^i(\tau_k^i)) \\
\mu^i(\tau_k^{i-}) &= \frac{\partial C_k}{\partial \alpha_k^i}
\end{aligned}$$

for $k = 1, \cdots, N-1$ and where we let we let $\tau_N = t_f$ and $\tau_0 = t_0$. We can now use these optimality conditions to implement a gradient descent algorithm for determining the optimized $\bar{\tau}^i$ and $\bar{\alpha}^i$ (initialized at the nominal values) as was discussed in [15]. This construction is in fact the fundamental tool in the MDL compilation process shown in the second block of Figure 1.

2.2 Languages for Puppet Plays

We use the MDL compiler developed in Section 2.1 for coordinating multiple puppets (such as the puppet in Figure 2), with specifications written in a special MDL for puppetry, MDLp, that mimics how puppeteers compose puppet

(a) (b) (c)

Fig. 2. An image sequence of the puppet executing a *wave* followed by a *walk* mode

plays. In fact, real puppet plays are written in a special script that enables the specification of puppet motion that must be choreographed with music and other puppets. Each line in a puppet play combines the agents involved, their motions, and the timing and spatial requirements [4,8]. For example, this excerpt from *Rainforest Adventures*[1] describes the motion for three puppets Female (`F`), Male 1 (`1`) and Male 2 (`2`):

```
4. F 1 2 fly up and stay and drop fast
5. F hops in place, 1 hops 4 SR turns hops 4 SL
```

The left column of the script displays the *count* number, which denotes the timing for motions for the agents listed in the line. In this case, the agents `F`, `1`, and `2` will perform several actions during the fourth count of this scene. Note that the `drop` motion is parameterized by a relative speed: `fast`. We interpret this modifier as the energy parameter α, described in Section 2.1. Another important element of the play specification is the designated regions seen in count `5`: `SR` ("stage-right") and `SL` ("stage-left"). We use these stage descriptions as the regions in the set $\mathcal{R}$.

Accordingly, for our puppet platform, we can create several motions for the set $\mathcal{K}$ and break up the stage into the same regions used in puppetry. For example, a MDLp mode for "walking" could be written as $(1, \texttt{walk}(\alpha_1), r_2, 3)$, which is interpreted as "puppet 1 walks at speed α_1 within region 2 for 3 counts." These puppetry specification language details are used when we program example plays and use the MDL compiler to generate modified control code for multi-puppet plays with spatial constraints.

3 Spatial MDL Optimization

As mentioned before, we want to specify motion programs for multiple agents, where each agent must execute the action within some region. For example, if the

[1] Courtesy of Jon Ludwig, Artistic Director of the Center for Puppetry Arts, Atlanta, Georgia, `http://www.puppet.org`.

agent is in $\mathbb{R}^2$, with its coordinates denoted by (x, y), we could use *hard* spatial constraints to keep the x position of the agent within a set interval, i.e. $[x_1, x_2]$. However, this approach would lead to increased computational complexity as the number of agents grows, since each agent's spatial constraints have to be enforced. Alternatively, we could use *soft* spatial constraints by making the costs $L(x,t)$ and $\Psi(x(\tau))$ in (2) penalize (or, benefit) the spatial location of the agent. Consequently, these compiler costs are tuned depending on the particular task that each agent must achieve. In other words, given the same example agent in $\mathbb{R}^2$, we could alter the costs to completely ignore the y position, opting instead to weight only the agent's x position. Using these design considerations, we formulate an example motion program for puppets, developing cost functions for insertion into (2), and subsequently optimizing a play with respect to this cost functional.

3.1 Example: Spatial Optimization for Multiple Puppets

In this section we demonstrate the MDL compiler proposed in Section 2.1 by scripting a play with MDLp. Although the actual puppet dynamics is quite complex (e.g. [12]), the spatial location of the puppet can be handled without taking the joint angles into account. Instead we envision a system in which the gross spatial actuation of the puppet takes on planar unicycle dynamics. We denote each agent's planar state with the dynamics,

$$\dot{z} = \begin{bmatrix} \alpha v \sin\gamma \\ \alpha v \cos\gamma \\ u_\gamma \end{bmatrix}.$$

The α in these dynamics is the scaling parameter discussed before and v is a constant maximum speed. Additionally, γ represents the heading angle of the puppet and is driven directly by some signal u_γ. Also, the joint angles of the arms and legs are represented by the vector $q = [\theta_r\ \phi_r\ \theta_l\ \phi_l\ \psi_r\ \psi_l]^T$, where the r and l subscripts denote right and left, respectively. The motion of these arm and leg joint angles is modeled kinematically with rigid strings. Thus, the model for the joint angle motion is of the form $\dot{q} = Iu$, where I is the identity matrix and u is the chosen input signal. We concatenate the z and q states, and denote the puppet's state as $\bar{x} = [z\ q]^T$. (Note that we choose a simplified model for this puppet for less intensive algorithm computations. For a deeper examination of puppet models see [12,22].)

In this example, we want the puppets to stay as close to the center of their designated regions as possible; therefore, we use a quadratic cost for $L(\bar{x}, t)$ and $\Psi(\bar{x}(\tau))$ in (2). Additionally, the desired trajectory terms in these costs, denoted by $\bar{x}^d$, will depend on the regions specified in the MDLp script.

Using these cost design choices, we let $L(\bar{x}, t) = (\bar{x} - \bar{x}^d)^T P(\bar{x} - \bar{x}^d)$, where P is a 9×9 positive definite weight matrix. The other cost function that accounts for spatial penalties is $\Psi(\bar{x})$. We define this function as,

$$\Psi_k(x(\tau_k)) = (\bar{x}(\tau_k) - \bar{x}^d(\tau_k))^T Z(\bar{x}(\tau_k) - \bar{x}^d(\tau_k)),\ k = 1, \cdots, N-1,$$

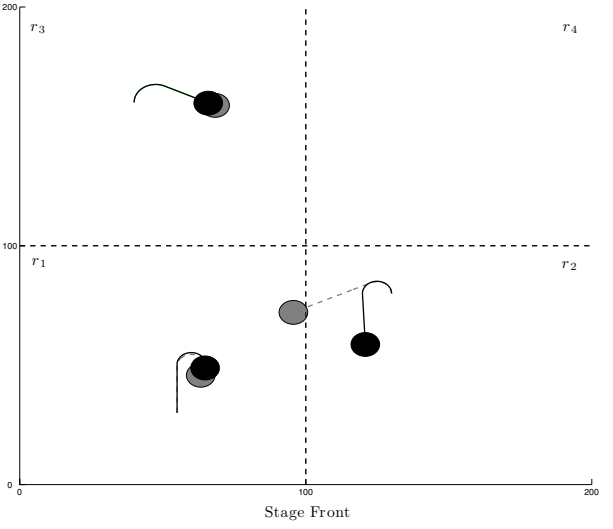


Fig. 3. Image of the puppet motions before (gray) and after (black) the MDL compilation process

where $Z \succ 0$ is another weight matrix. This function is similar to $L(\bar{x}, t)$; however, its weight matrix penalizes *only* the position of the agent, and it is evaluated only at the switch times, τ_k. Finally, we penalize the scaling factors and time deviations in the same way as in [15]: $C_j = \rho_j \alpha_j^2$, for $j = 1, \cdots, N$, and $\Delta_k(\tau_k) = w_k(T_k - \tau_k)^2$ for $k = 1, \cdots, N-1$.

As an example, we implemented a small collection of controls, $\mathcal{K} = \{\kappa_1 = \texttt{waveLeft}, \kappa_2 = \texttt{walk}, \kappa_3 = \texttt{walkInCircles}\}$. Using these controllers, we constructed the following MDLp play:

$$(p^1, \kappa_1(1.2), r_1, 2.5)(p^1, \kappa_2(1.3), r_1, 3)(p^1, \kappa_3(1), r_1, 4)$$
$$(p^2, \kappa_1(1.2), r_3, 2.5)(p^2, \kappa_3(1.5), r_3, 2)(p^2, \kappa_2(1.3), r_3, 3)$$
$$(p^3, \kappa_3(1), r_2, 2)(p^3, \kappa_2(1.5), r_2, 4).$$

The initial run of this MDLp play is illustrated by the *gray* lines and shapes in Figure 3. Note that puppets 1 and 2 (located in r_1 and r_3 in the figure) behave relatively well under their nominal plays. However, puppet 3 breaches the boundary between r_1 and r_2 while its MDLp string requires it to remain in r_2.

After running the MDL compiler on these strings, the improved runtime behavior is illustrated by the black lines and shapes in Figure 3. Puppet 3's trajectory is now correctly within r_2, as prescribed in the original MDLp string. Also, all three puppets reduce their cost, as shown in Figure 4. Note that puppet 3 takes the longest, computing 100 iterations before minimizing its cost. This iteration count shows how bad the nominal program was at satisfying the cost functional (2). Additionally, our algorithm uses a conservative, fixed-step gradient descent to limit the amount of numerical error, which will slow down

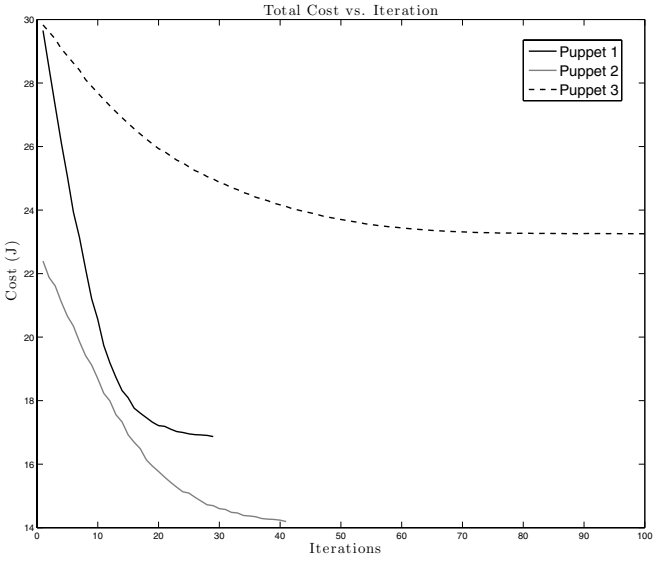


Fig. 4. This figure shows the costs as a function of the MDL compiler algorithm iteration when compiling a play for three puppets with *spatial* constraints. Puppet 1 completed in 29 iterations, Puppet 2 completed in 41 iterations, and Puppet 3 took 100 iterations.

convergence as the derivatives (3) get closer to 0. If a dynamic step size were used (such as Armijo step-size [1]) then convergence would be faster. This work demonstrates that we can solve the problem of improving the multi-agent motion program given spatial costs. We now turn to the problem of generating optimized control code under networked timing constraints.

4 Constrained Timing Optimization

The work in Section 3 dealt with optimizing the MDL strings of multiple agents, considering each agent's dynamics and spatial costs. Additionally, many systems require *hard* timing constraints, such as terminating one particular mode *before* some other agent's mode completes. In a puppet play, missing these timing constraints may lead to benign issues, such as awkward character placement, to serious problems, such as collisions and string tangling. This section considers generating optimized MDL programs under timing inequality constraints by distributing the constraint among the agents.

In this problem, we again assume that the motion program has m puppets, each operating under their own dynamics. Additionally, each puppet switches between C_i control modes, with the terminal time denoted by $t_f = \tau_{C_i},\ i \in \mathcal{M}$. In other words, a direct modification to the formulation described in Section 2.1 gives that each puppet be governed by the dynamics,

$$\dot{x}^i(t) = \begin{cases} f_1(x^i(t)), \ t \in [0, \tau_1^i) \\ f_2(x^i(t)), \ t \in [\tau_1^i, \tau_2^i) \\ \vdots \\ f_{C_i}(x^i(t)), \ t \in [\tau_{C_{i-1}}^i, \tau_{C_i}^i] \end{cases}$$

for agents $i = 1, \cdots, m$. Let moreover the cost functional be defined as

$$J(\bar{\tau}^1, \ldots, \bar{\tau}^m) = \int_0^T \sum_{i=1}^m D^i(x^i, t) \ dt = \sum_{i=1}^m J^i(\bar{\tau}^i) \tag{4}$$

where $D^i(x^i, t)$ is the cost associated with operating system i for a particular control mode's time duration, without taking the other systems into account. To illustrate the way in which the temporal constraints show up, we assume, without loss of generality, that the temporal constraint only affects the d^{th} switch for systems j and k, where $j, k \in \mathcal{M}$, as $\tau_d^j - \tau_d^k \leqslant 0$. Note that this minimization formulation results in a *separable* optimization problem, since the function to be minimized (4) and the timing constraint depend additively on their domains [17].

This optimization problem can be solved by augmenting the cost with a Lagrangian term $\nu(\tau_d^j - \tau_d^k)$, and then jointly solving it across all the switching times for all the puppets. However, we do not want to use this centralized solution, since the ultimate goal is to have several autonomous agents (or in this case, puppets) optimize their plays in a decentralized fashion. Since we have already noted that the problem is separable we can break up the solution process. We specifically choose the approach known as *team theory*, recently explored in [18]. (Note that the details given below are not due to us, but rather that we highlight their application to the problem of distributed timing control as it pertains to the robotic marionette application.)

4.1 Distributed Timing Coordination

Puppets j and k ($j \neq k$) are temporally constrained via the d^{th} switch as $\tau_d^j - \tau_d^k \leq 0$. The constrained problem becomes

$$L(\tau_d^j, \tau_d^k, \nu) = J^j(\tau_d^j) + J^k(\tau_d^k) + \nu(\tau_d^j - \tau_d^k) \tag{5}$$

where we have assumed, without loss of generality, that the only control parameters are τ_d^j and τ_d^k, and the other switching times are considered to be fixed. It should be noted that the cost functionals are decoupled (i.e. cost J^j depends *only* on system j's dynamics). Therefore, taking the derivative of the Lagrangian with respect to ν results in the expression,

$$\frac{\partial L}{\partial \nu} = \tau_d^j - \tau_d^k,$$

in combination with the previously defined gradient expressions for the switching times.

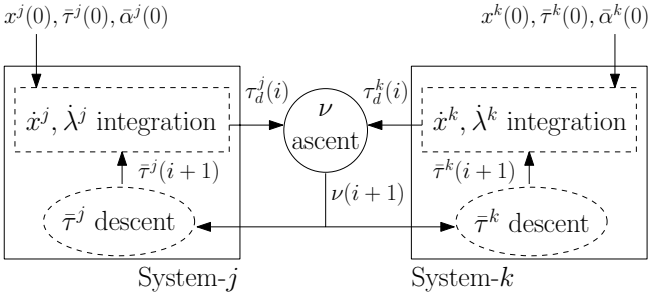


Fig. 5. This figure shows how information propagates between the two subsystems (puppets) in order to solve the networked timing problem. The initial values for system j are denoted $\bar{\tau}^j(0)$, $\bar{\alpha}^j(0)$ and similarly for system k.

Now, algorithmically, this formulation is interesting in that the dual problem becomes $g^\star = \max_\nu g(\nu),\ \nu \geqslant 0$, where

$$g(\nu) = \min_{\tau_d^j, \tau_d^k} \{J^j(\tau_d^j) + J^k(\tau_d^k) + \nu(\tau_d^j - \tau_d^k)\}. \tag{6}$$

Note that we are looking for a *local* minimum, since a global minimum may actually lead to behavior that is undesired by the original motion program specification.

An inefficient solution to this max-min problem would apply gradient descent on the minimization problem (6) and follow with a *single* gradient ascent step for $\max_\nu g(\nu)$, and repeat until a solution is found. However, since we already know that this max-min problem is separable, we can solve it using a saddle-point search algorithm, known as *Uzawa's algorithm* [2]. The Uzawa algorithm allows the descent *and* the ascent steps to be performed simultaneously. Consequently, we use a gradient descent for the switch times, and a gradient ascent for the Lagrange multiplier ν allowing us to decouple the numerical solution process and let the networking aspect be reflected only through the update of the multiplier, as was done in [18]. In fact, if we let

$$\begin{aligned} \dot{\tau}_d^j &= -\frac{\partial J^j}{\partial \tau_d^j} - \nu \\ \dot{\tau}_d^k &= -\frac{\partial J^k}{\partial \tau_d^k} + \nu \\ \dot{\nu} &= \tau_d^j - \tau_d^k \end{aligned}$$

all that needs to be propagated between the two systems is the value of the Lagrange multiplier ν. This observation in [18] leads us to a general architecture for solving networked, switched-time optimization problems, as shown in Figure 5.

This numerical architecture lets us optimize the switch times individually at each algorithm iteration, denoted by the index i. First, we initialize both systems with feasible switch times and scaling parameters based on a play of length p. These values we denote with the arrays, $\bar{\tau}^j(0) = [\tau_1^j(0)\ \cdots\ \tau_{N-1}^j(0)]$ and $\bar{\alpha} = [\alpha_1^j(0)\ \cdots\ \alpha_N^j(0)]$, respectively. Then we perform the forward-backward integration of the x^j and x^k systems and their associated co-states (λ^j, λ^k). In

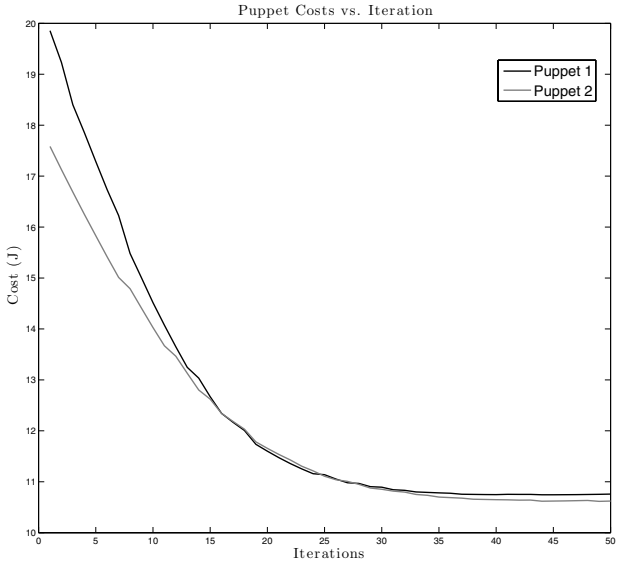


Fig. 6. The cost of both puppets using the distributed switch time constraint architecture

parallel to those integrations, the ν state is incremented using the current values for the switch times. These values are then passed to the individual systems so that they can take their gradient descent steps with the new ν values.

4.2 Example: Time-Switch Constraints for Puppetry

Using the same collection of control laws from Section 3 we define a multi-puppet play as follows,

$$(1, \kappa_1(1.2), r_1, 2.5)\ (1, \kappa_2(1.3), r_1, 3)\ (1, \kappa_3(1), r_1, 3)$$
$$(2, \kappa_1(1.2), r_3, 2.5)\ (2, \kappa_3(1.5), r_3, 3)\ (2, \kappa_2(1.3), r_3, 2.5).$$

This play uses two agents, both executing three modes with various timing requirements and scaling parameters.

Following the discussion of switch-time constraints in Section 4.1 we choose to constrain the *first* switch of each puppet, i.e. $d = 1$. If we denote $\bar{\tau}^i = [\tau_1^i\ \tau_2^i]$ as the switch times and $\bar{\alpha}^i = [\alpha_1^i\ \alpha_2^i\ \alpha_3^i]$ as the scaling parameters for puppet i, then the constrained minimization problem for these two puppets is stated as

$$\min_{\bar{\tau}^1, \bar{\tau}^2, \bar{\alpha}^1, \bar{\alpha}^2} J^1(\bar{\tau}^1, \bar{\alpha}^1) + J^2(\bar{\tau}^2, \bar{\alpha}^2)$$
$$\text{s.t. } \tau_1^2 \leqslant \tau_1^1.$$

We formulate a Lagrangian for this problem in a similar way as equation (5),

$$L(\bar{\tau}^1, \bar{\alpha}^1, \bar{\tau}^2, \bar{\alpha}^2, \nu) = J^1(\bar{\tau}^1, \bar{\alpha}^1) + J^2(\bar{\tau}^2, \bar{\alpha}^2) + \nu(\tau_1^2 - \tau_1^1),$$

and then apply the proposed algorithm approach visualized in Figure 5.

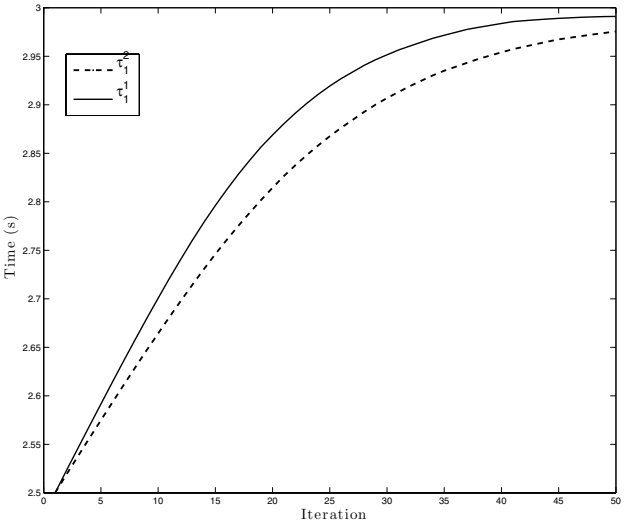


Fig. 7. A graph of the constrained switch time values τ_1^2 and τ_1^1

Figure 6 displays the cost graphs for the two puppets after the execution of the distributed algorithm. The cost is indeed reduced for both puppets and, furthermore, Figure 7 shows that the required inequality constraint is satisfied. The optimal switch times and scaling parameters for puppet 1 are $\bar{\tau}^1 = [2.9906\ 3.0463]$ and $\bar{\alpha}^1 = [1.1903\ 1.3249\ 1.0204]$, respectively. Additionally, the results for puppet 2's parameters are $\bar{\tau}^2 = [2.9683\ 2.9157]$ and $\bar{\alpha}^2 = [1.1901\ 1.5228\ 1.3124]$.

5 Conclusion

In this paper we discussed recent results for generating optimized control code for collections of interacting MDL-based systems, which are, in this case, robotic puppets. We formulated a special instantiation of the MDL framework that includes spatial costs and controller energy scaling. An optimal control-based compiler was developed for these types of MDLs, and applied to a collection of autonomous puppets. Finally, our work concludes with an examination and simulation of optimizing the MDL motion program for agents with timing constraints.

Acknowledgement

We acknowledge the U.S. National Science Foundation for its support through grant number 0757317. Additionally, we thank our collaborators Todd Murphey at Northwestern University and Jon Ludwig at the Atlanta Center for the Puppetry Arts.

References

1. Armijo, L.: Minimization of Functions Having Lipschitz Continuous First-Partial Derivatives. Pacific Journal of Mathematics 16, 1–3 (1966)
2. Arrow, K., Hurwicz, L., Uzawa, H.: Studies in Nonlinear Programming. Stanford University Press, Stanford (1958)
3. Attia, S.A., Alamir, M., Canudas de Wit, C.: Sub Optimal Control of Switched Nonlinear Systems Under Location and Switching Constraints. In: Proc. 16th IFAC World Congress, Prague, Czech Republic, July 3-8 (2005)
4. Baird, B.: The Art of the Puppet. Mcmillan Company, New York (1965)
5. Belta, C., Bicchi, A., Egerstedt, M., Frazzoli, E., Klavins, E., Pappas, G.: Symbolic Planning and Control of Robot Motion. IEEE Robotics and Automation Magazine (March 2007)
6. Brockett, R.W.: On the Computer Control of Movement. In: Proceedings of the 1988 IEEE Conference on Robotics and Automation, New York, April 1988, pp. 534–540 (1988)
7. Egerstedt, M., Brockett, R.W.: Feedback Can Reduce the Specification Complexity of Motor Programs. IEEE Transactions on Automatic Control 48(2), 213–223 (2003)
8. Egerstedt, M., Murphey, T., Ludwig, J.: Motion Programs for Puppet Choreography and Control. In: Bemporad, A., Bicchi, A., Buttazzo, G. (eds.) HSCC 2007. LNCS, vol. 4416, pp. 190–202. Springer, Heidelberg (2007)
9. Egerstedt, M., Wardi, Y., Axelsson, H.: Transition-Time Optimization for Switched-Mode Dynamical Systems. IEEE Trans. on Automatic Control AC-51, 110–115 (2006)
10. Frazzoli, E.: Explicit Solutions for Optimal Maneuver-based Motion Planning. In: Proceedings of 42nd IEEE Conference on Decision and Control, vol. 4, pp. 3372–3377 (2003)
11. Frazzoli, E., Dahleh, M.A., Feron, E.: Maneuver-Based Motion Planning for Nonlinear Systems with Symmetries. IEEE Transactions on Robotics 21(6), 1077–1091 (2005)
12. Johnson, E., Murphey, T.: Dynamic Modeling and Motion Planning for Marionettes: Rigid Bodies Articulated by Massless Strings. In: International Conference on Robotics and Automation, April 2007, pp. 330–335 (2007)
13. Kloetzer, M., Belta, C.: A Fully Automated Framework for Control of Linear Systems From Temporal Logic Specifications. IEEE Transactions on Automatic Control 53(1), 287–297 (2008)
14. Manikonda, V., Krishnaprasad, P.S., Hendler, J.: Languages, behaviors, hybrid architectures and motion control. In: Willems, J.C., Baillieul, J. (eds.) Mathematical Control Theory. Springer, Heidelberg (1998)
15. Martin, P., Egerstedt, M.: Optimal Timing Control of Interconnected, Switched Systems with Applications to Robotic Marionettes. In: Workshop on Discrete Event Systems, Gothenburg, Sweden (May 2008)
16. Martin, P., de la Croix, J.P., Egerstedt, M.: MDLn: A Motion Description Language for Networked Systems. In: Proceedings of 47th IEEE Conference on Decision and Control (December 2008)
17. Moeseke, P.V., de Ghellinck, G.: Decentralization in Separable Programming. Econometrica 37(1), 73–78 (1969)
18. Rantzer, A.: On Price Mechnanisms in Linear Quadratic Team Theory. In: IEEE Conference on Decision and Control, New Orleans, LA (December 2007)

19. Shaikh, M.S., Caines, P.: On Trajectory Optimization for Hybrid Systems: Theory and Algorithms for Fixed Schedules. In: IEEE Conference on Decision and Control (2002)
20. Tabuada, P., Pappas, G.J.: Linear Time Logic Control of Discrete-time Linear Systems. IEEE Transactions on Automatic Control 51(12), 1862–1877 (2006)
21. Xu, X., Antsaklis, P.J.: Optimal Control of Switched Systems via Nonlinear Optimization Based on Direct Differentiations of Value Functions. Int. J. of Control 75, 1406–1426 (2002)
22. Yamane, K., Hodgins, J.K., Brown, H.B.: Controlling a Marionette with Human Motion Capture Data. In: International Conference on Robotics and Automation, vol. 3, pp. 3834–3841 (2003)

Decompositional Construction of Lyapunov Functions for Hybrid Systems*

Jens Oehlerking and Oliver Theel

Department of Computer Science
Carl von Ossietzky University of Oldenburg
26111 Oldenburg, Germany
{jens.oehlerking,oliver.theel}@informatik.uni-oldenburg.de

Abstract. In this paper, we present an automatable decompositional method for the computation of Lyapunov functions for hybrid systems with complex discrete state spaces. We use graph-based reasoning to decompose hybrid automata into subgraphs, for which we then solve semidefinite optimization problems to obtain local Lyapunov functions. These local computations are made in a way that ensures that the family of local Lyapunov functions forms a global Lyapunov function, proving asymptotic stability of the system. The main advantages over standard LMI methods are 1) improved numerical stability due to smaller optimization problems, 2) the possibility of incremental construction of stable hybrid automata and 3) easier diagnosis of unstable parts of the automaton in case no Lyapunov function can be found.

1 Introduction

Proofs for progress properties of dynamic systems are usually conducted with the help of functions measuring the desired progress. For control systems and the property of asymptotic stability, such functions are called *Lyapunov functions*. A Lyapunov function maps each continuous state onto a non-negative real number. The function is required to decrease along every solution of the system and to have its only local minimum at the equilibrium point it is supposed to converge to. If one succeeds in identifying a function with this property, then this completes the proof of asymptotic stability. Naturally, there has been a strong desire to come up with algorithmic methods for the construction of such functions for hybrid systems. An important step in this direction was the development of linear matrix inequality (LMI) based methods for Lyapunov function computation by Johansson and Rantzer [1] and Pettersson and Lennartson [2]. Through the use of LMIs it is directly possible to employ numerical optimization software for the computation of piecewise quadratic Lyapunov functions. With the addition of methods based on the sums-of-squares decomposition [3], LMI methods can also be used

* This work was partly supported by the German Research Foundation (DFG) as part of the Transregional Research Center "Automatic Verification and Analysis of Complex Systems" (SFB/TR 14 AVACS), www.avacs.org.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 276–290, 2009.

to identify higher degree piecewise polynomial [4] and piecewise non-polynomial Lyapunov functions [5] for systems with complex dynamics. However, for hybrid systems with complex discrete structure, the computation of such piecewise continuous Lyapunov functions, while theoretically possible, often fails in practice for numerical reasons, or even worse, solvers report false Lyapunov functions due to inherent inaccuracies in the numerical algorithms. Generally, the more complex the discrete structures are, the more likely these problems are to occur. In fact, numerical examples in the literature are usually limited to just very few discrete modes. A different approach of analyzing systems with many discrete states is switched system analysis [6,7]. Switched systems only make few assumptions on the switching logic, viewing the discrete state of the hybrid system as an input signal with relatively mild restrictions (e.g., dwell time). Because of this, stability analysis for switched systems tends to scale better to systems containing many discrete modes, but is in itself more conservative and therefore often not sufficient for proving stability of systems with complex switching logic.

In this paper we propose a decompositional theory of stability proofs for hybrid systems with possibly complex discrete structure. We decompose hybrid automata into sub-automata for which small-scale optimization problems are solved. The results from these computations are then merged in such a way that a stability proof through a piecewise continuous Lyapunov function is obtained for the entire system. Not only does this approach reduce the numerical load on the solvers, it also allows the incremental design of stable hybrid automata by subsequent addition of new modes and transitions, by examining the Lyapunov function space for the subautomaton to be added. The compositional property is independent of the actual parameterization used for the Lyapunov functions and therefore compatible with the sums-of-squares decomposition or alternative (non-LMI) methods for Lyapunov function computation. In the scope of this paper, we will, however, employ LMI-based methods for the computation of local Lyapunov functions.

The decomposition is split into two major steps. The first step consists of the decomposition of the graph given by the hybrid automaton into *strongly connected components*. As it turns out, these components can be considered completely independently of one another for Lyapunov function computation (i.e., the Lyapunov functions of two components need not be interrelated in any manner). The second step then proceeds to decompose these components into *cyclic substructures*, for which, one by one, smaller optimization problems have to be solved. Whenever the computation for one such cycle is completed, the cycle is removed from the hybrid automaton and replaced by a constraint on its intersection nodes with other cycles.

The paper is structured as follows. After defining the stability property and the Lyapunov function concept used in this paper in Section 2, we introduce the two steps of decomposition in Section 3 and subsequently apply it to an example hybrid automaton representing a cruise controller with various saturations. Furthermore, we discuss issues related to the implementation of the proposed method. Finally, Section 4 concludes the paper and outlines future work.

2 Preliminaries

This section defines the notions of hybrid systems, stability, and Lyapunov functions that are used in the remainder of the paper.

Definition 1 (Hybrid Automaton). *Define $\mathcal{D}_n$ as the set of all Lipschitz-continuous differential equations on $\mathbb{R}^n$ and $\mathcal{P}(X)$ as the power set of X. A* hybrid automaton *H is a tuple $(\mathcal{M}, \mathcal{S}, \mathcal{T}, Flow, Inv)$, where*

- *$\mathcal{M}$ is a finite set of* modes
- *$\mathcal{S} = \mathbb{R}^n, n \in \mathbb{N}$ is the* continuous state space
- *$\mathcal{T}$ is a set of* mode transitions *given as tuples $(m_1, m_2, G, Update)$, where*
  - *$m_1 \in \mathcal{M}$ is the* source mode
  - *$m_2 \in \mathcal{M}$ is the* target mode
  - *$G \subseteq \mathcal{S}$ is the* guard set
  - *$Update : \mathcal{T} \times \mathcal{S} \to \mathcal{S}$ is the* update function *for the continuous state*
- *$Flow : \mathcal{M} \to \mathcal{D}_n$ is the* flow function*, mapping each mode onto a continuous evolution given as a differential equation*
- *$Inv : \mathcal{M} \to \mathcal{P}(\mathcal{S})$ is the* invariant function*, mapping each mode onto a subset of the continuous state space.*

A trajectory *of the hybrid automaton H is an infinite solution (in the classical sense) and is denoted $x(t)$. Each trajectory is associated with a* mode sequence *m_i, containing, in order, all modes visited by the trajectory.*

We will henceforth use the terms "hybrid system" and "hybrid automaton" interchangeably.

Definition 2 (Asymptotic Stability). *A hybrid system is called* globally stable *if $\forall \epsilon > 0 \, \exists \delta > 0 \, \forall t > 0 : ||x(0)|| < \delta \Rightarrow \ ||x(t)|| < \epsilon$, and* globally attractive *if for all trajectories $x(t)$ we have $x(t) \to 0$ for $t \to \infty$, where 0 is the origin of $\mathbb{R}^n$. A hybrid system that is both globally stable and globally attractive is called* globally asymptotically stable *(GAS).*

Since only infinite solutions are considered, we generally allow the existence of finite nonextendable trajectory segments (e.g., through Zeno behavior). However, these segments have no bearing with respect to stability analysis. This notion of stability is also sometimes called *preasymptotic stability* [8]. The global attractivity property can therefore be viewed as the absence of (infinite) trajectories that do not converge.

To prove asymptotic stability in a decompositional manner, we will employ a variant of the well-known Lyapunov theorem for hybrid systems and discontinuous Lyapunov functions.

Definition 3 (Definiteness). *A function $f : X \to \mathbb{R}, X \subseteq \mathbb{R}^n$ is called* positive definite *on X, if $f(x) > 0$ for $x \neq 0$, and $f(0) = 0$, in case $0 \in X$. A function f is called* negative definite *on X, if $-f$ is positive definite on X.*

Theorem 1 (Discontinuous Lyapunov Functions [2]). *Let H be a hybrid automaton according to Def. 1. If for each $m \in \mathcal{M}$ there exists a function $V_m : \mathcal{S} \to \mathbb{R}$ such that*

(1) V_m is positive definite on $Inv(m)$,
(2) $\dot{V}_m := \frac{dV_m}{dx} f_m$ is negative definite on $Inv(m)$, where f_m is the right hand side of the differential equation $Flow(m)$
(3) for each mode transition $(m_1, m_2, G, Update) \in \mathcal{T}$:
$x \in G \Rightarrow V_{m_2}(Update(x)) \leq V_{m_1}(x)$,

then H is globally asymptotically stable. The function V_m is called the local Lyapunov function (LLF) *of H for mode m. The family of the $V_m, m \in \mathcal{M}$ is called the* global (discontinuous) Lyapunov function (GLF) *of H.*

It is well-known that parameterized Lyapunov functions can be computed via convex optimization with the help of linear matrix inequalities [9]. One begins by selecting a parameterized Lyapunov function template for each function V_m and then proceeds to identify suitable parameters through convex optimization. These templates can come in various forms: quadratic functions can be dealt with directly [1], higher degree polynomials can be employed with help of the sums-of-squares decomposition [3], and even some nonlinear functions can be handled [5]. Each condition (1)-(3) of Theorem 1 is then mapped onto a constraint of the optimization problem, and all resulting constraints are then solved simultaneously. The solution of the optimization problem (if one can be found), is a valuation of the free parameters in the V_m, such that all conditions of Theorem 1 are fulfilled. This concluded the stability proof. While this approach is suitable for hybrid systems with small discrete state space (i.e., few modes), it becomes increasingly hard to use for larger hybrid automata. With each additional node, a new LLF with additional free parameters must be added to the optimization problem, together with additional Lyapunov constraints. This can cause numerical instability, resulting in finding no solution at all, or leading to false positives (i.e., the solver returns a solution that, upon closer inspection, marginally violates the conditions of Theorem 1). Furthermore, a negative answer is not constructive – there is no indication of the part of the system causing the problem. Therefore, we only use convex optimization locally on the automaton and string these local results together with the help of graph theory. This decompositional method is discussed next.

3 Graph-Based Decomposition

This section describes a decompositional method that can be used to verify the existence of a GLF (and thereby show asymptotic stability) for a hybrid system in a decompositional manner. The decomposition consists of two basic steps. First, we will give the necessary theorems for the decomposition into *strongly connected components* of the underlying graph structure of a hybrid automaton and discuss their implications. In a nutshell, it is sufficient to prove GAS for

each strongly connected component in isolation, with no interrelation between their Lyapunov functions whatsoever. Then, in the second step, stability of the individual strongly connected components is shown decompositionally. This is achieved by solving *local* LMI problems for cyclic sub-automata. For a cyclic subgraph, a constraint P on its intersection nodes with the remainder of the graph is computed, such that P implies the existence of a GLF for the entire cycle. This allow us to remove the cycle, apart from the intersection nodes, from the graph and replace it by attaching the additional constraint P to the intersection node.

These local computations can also be combined with the sums-of-squares decomposition (to deal with non-affine dynamics or non-quadratic Lyapunov functions) and the S-procedure [10, p.94] (to take into account the guards and invariants of the hybrid automaton). In general, one is not even restricted to using LMI based methods, as long as such a constraint P is obtained. The core results are independent of the actual computation method used.

By repeating such reduction steps, the graph defined by the hybrid automaton can eventually be reduced to the empty graph, proving stability of the strongly connected component. As we only consider local LMI problems for cycles (and not the entire strongly connected component at once), the optimization problems to be solved are kept relatively small and less prone to numerical problems than monolithical approaches.

3.1 Decomposition into Strongly Connected Components

A first, computationally simple, step in decomposing a hybrid automaton into sub-automata for Lyapunov function generation is the identification of strongly connected components of the graph representing the automaton. Strongly connected components can be computed in linear time through well-known algorithms [11]. Apart from computing the components, this type of decomposition comes at no additional cost compared to a monolithic approach. As it turns out, a hybrid automaton is GAS if all of its strongly connected components are GAS. For this reason, the large semidefinite optimization problem associated with the hybrid automaton can be broken down into a family of smaller problems associated with the strongly connected components. These smaller problems can then be solved completely independently of one another, effectively splitting the hybrid automaton into several sub-automata. The guards and updates of transitions connecting several strongly connected components can be ignored completely for the Lyapunov function computation.

Definition 4 (Strongly Connected Component). *A* strongly connected component *(SCC) of a directed graph G is a maximal subgraph G', such that for each pair of nodes $n_1 \neq n_2$ in G', there exists a forward path from n_1 to n_2.*

Each node of a graph is part of exactly one SCC. Each edge is either part of exactly one SCC, or it connects two SCCs. Two nodes belonging to different SCCs cannot lie on a common circular path. This implies that there is a relative

order on the SCCs of a directed graph, defined by the edges connecting them: for two SCCs C_1 and C_2, we write $C_1 \prec C_2$ if there exists an edge in the graph pointing from a node in C_1 to a node in C_2.

For a hybrid system H and fixed trajectory $x(t)$ of the system, consider the sequence $C_1 \prec C_2 \prec \ldots$ containing the SCCs the associated mode sequence m_i passes through, in the same order. Each SCC can only occur once in this sequence, and therefore the sequence must have finite length. To prove convergence, it is permissible to ignore finite prefixes of trajectories, so it suffices to just examine the system behavior in the last SCC in the list – the one that is entered, but never left again. Consequently, it is sufficient to prove convergence for each SCC individually. If a SCC is GAS, then this implies for a run of the system that either (a) the SCC is left eventually, or (b) the trajectory will converge toward 0 within this particular SCC. If the continuous state 0 does not satisfy any invariants of this SCC, then only option (a) is possible and the LLF can be viewed a kind of termination function for the SCC. This kind of decomposition also maintains global stability. For these reasons, GLFs for different SCCs can be computed completely independently, and the transitions connecting the SCCs, along with their guards and update functions, can be discarded.

Theorem 2 (Decomposition into Strongly Connected Components). *Let H be a hybrid automaton. If all sub-automata pertaining to the SCCs of H are GAS then so is H.*

Proof. Global Attractivity: *Let $x(t)$ be a fixed trajectory of H, and m_i the associated mode sequence. Let C_k be the sequence of SCCs m_i passes through, in corresponding order. Since no SCC can occur twice in C_k, the total number of SCCs is finite, therefore C_k must be finite. Let δ be the point in time when $x(t)$ enters the final SCC of C_k and let $\tilde{x}(t) = x(t-\delta)$. Since C_i is GAS, we have $\tilde{x}(t) \to 0$ for $t \to \infty$, which implies $x(t) \to 0$.*

Global Stability: *Let P be the set of all possible sequences $C_1 \prec \ldots \prec C_n$ consisting of SCCs of H and let $\epsilon > 0$. For a fixed $p = (\tilde{C}_1 \prec \ldots \prec \tilde{C}_n) \in P$, by successively applying the stability properties of the C_i in reverse order, we have for all trajectories entering the SCCs in the order given by p:*

$$\exists\, \delta_n^p > 0 \,\forall\, t_n > 0, t > t_n : x(t_n) < \delta_n^p \Rightarrow x(t) < \epsilon,$$

$$\exists\, \delta_{n-1}^p > 0 \,\forall\, t_{n-1} > 0, t_n > t_{n-1} : x(t_{n-1}) < \delta_{n-1}^p \Rightarrow x(t_n) < \delta_n^p,$$

and finally, for C_1,

$$\exists\, \delta_1^p > 0 \,\forall\, t_1 > 0, t_2 > t_1 : x(0) < \delta_1^p \Rightarrow x(t_1) < \delta_2^p$$

By setting $\delta = \min\{\delta_1^p \mid p \in P\}$, we obtain $\forall\, t > 0 : x_i(0) < \delta \Rightarrow x_i(t) < \epsilon$.

Remark 1. Theorem 2 allows us to consider all SCCs of a complex hybrid automaton separately, by computing separate Lyapunov functions, and still obtaining a proof of GAS with respect to one equilibrium for the entire system. It is also possible to adapt Theorem 2 to deal with different equilibria for different

SCCs. In this case, it is possible to guarantee convergence to one of the different equilibria in the system. This can, for instance, be used for hybrid automata that are supposed to converge to failsafe states, once certain safety conditions are violated.

The following lemma implies that this decomposition is not conservative with respect to Lyapunov functions.

Lemma 1. *Let H be a hybrid automaton and let $C_1, \ldots, C_n$ be the SCCs of H. If there exists a family of LLFs $V_m, m \in \mathcal{M}$, forming a GLF for H, then there exists a GLF for each C_i.*

This follows directly from the fact that a GLF can be split up into several Lyapunov functions on the subgraphs. If one is interested in proving only global attractivity, then it suffices to only consider SCCs where trajectories can potentially stay infinitely long, i.e., SCCs that can be the final SCC a trajectory moves into. This is a consequence of the proof of global attractivity for Theorem 2, which only requires the last SCC in sequence to be GAS.

Lemma 2. *Let H be a hybrid automaton and let $C_1, \ldots, C_n$ be exactly the strongly connected components of H for which there exists a trajectory entering but never leaving the component. If all C_i are globally attractive, then so is H.*

Remark 2. Note that there might be different representations of the same system as a hybrid automata that consist of smaller SCCs and therefore result in easier Lyapunov function computation. To discover such representations, for instance replacing one large SCC by two smaller ones exhibiting equivalent continuous behavior, reachable set computation can be employed.

In the next section, we are further going to decompose the SCCs into cyclic subgraphs, for which LMI problems can be solved in a way that guarantees asymptotic stability for the entire SCC.

3.2 Decomposition into Cycles

Since a single SCC can potentially still be a large part of the automaton (or even the entire automaton), we now proceed by decomposing these SCCs further. However, since the subgraphs of an SCC generally exhibit interdependencies in both directions (i.e., as opposed to SCCs there is no partial ordering of subgraphs that can be exploited), lossless decomposition is not possible. Instead, we will decompose an SCC into a cover of overlapping cycles. This is possible, since, inside each SCC, each node can be reached from every other node. Therefore, each node lies on at least one (simple) cycle of the graph. This decomposition is always correct. To limit the conservativeness of the approach, iterative approximation refinement techniques can be employed. We will now describe an algorithmic approach to deal with Lyapunov function computation on a per-cycle basis.

Definition 5 (Cyclic Decomposition). *A* simple cycle *of a directed graph $G = (N, E)$ with node set N and edge set E is a closed path that visits each node at most once. A* cyclic decomposition *of a graph is a family of simple cycles $C_i = (N_i, E_i)$ such that $N = \bigcup_i N_i$ and $E = \bigcup_i E_i$. A node $n \in N$ is a* border node *of G, if there exist two different cycles in a cyclic decomposition of G, both containing n.*

To achieve decomposition, we exploit the locality of the Lyapunov constraints in the hybrid automaton. A useful tool in this context are *constraint graphs*, which map Lyapunov constraints of the underlying optimization problem to the parts of the hybrid automaton's graph model they are based on.

Definition 6 (Constraint Graphs). *The* constraint graph $C(G)$ *of a hybrid automaton H is a graph that:*

1. *contains one node per mode in H, labelled with constraints (1) and (2) from Theorem 1 for that mode*
2. *contains one edge per mode transition in H, labelled with constraint (3) from Theorem 1 for the two Lyapunov functions corresponding to the incident modes*

For a subgraph C of $C(G)$, constr(C) is the conjunction of all constraints in C.

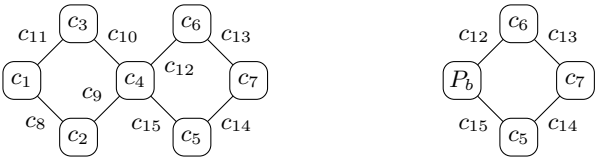
Constraint graphs will be used to visualize the locality of the Lyapunov and non-increasingness constraints. Whenever we talk about solving an LMI problems for some sub-automaton, we mean finding a solution that simultaneously fulfills all constraints on the nodes and edges of the constraint graph corresponding to the sub-automaton.

The basic idea of the algorithm is as follows. If, instead of solving the convex optimization problem for the whole SCC in one step, one wants to deal with one cycle at a time, the border nodes are exactly the nodes where the constraints associated with different cycles interfere with one another. In other words, each cycle that a border node b is part of implicitly induces different constraints on the LLF V_b of b. All these constraints need to be satisfied simultaneously for a LLF to exist. In contrast, all non-border nodes of a cycle are only relevant within the cycle and not subject to additional constraints from outside. First, we focus on cycles with only a single border node. For such a cycle, we compute a predicate P representing the constraints the cycle imposes on its border node. If we know that, whenever V_b fulfills P, there exists a GLF for the cycle, then we can discard all non-border nodes of the cycle and replace the cycle by attaching P to the border node (see Fig. 1). By successively removing cycles, one finally ends up with the empty graph, which implies the existence of a GLF for the entire SCC, and therefore its asymptotic stability. If we can only find cycles with more than one border node, we restructure the graph as described later in this section. The computational method can be summed up as follows:

1. check, whether there are cycles with at most one border node
2. if this is not the case, restructure the graph to produce one

3. select a cycle C with at most one border node
4. if cycle C has no border nodes, compute a Lyapunov function for cycle C
5. if cycle C has one border node b, compute a constraint P_b on V_b that implies the existence of LLFs for all nodes of the cycle and replace the constraint for b in the constraint graph by b
6. remove all non-border nodes of the cycle from the graph, together with all incident edges
7. if the resulting graph is empty, the SCC is GAS, otherwise return to step 1

If, during steps 4 or 5, no suitable Lyapunov function can be found, the previously computed border node constraints P_b can be refined iteratively. This is discussed in Section 3.4.



(a) before reduction (b) after reduction

Fig. 1. Constraint graphs before and after cycle reduction

Computing Predicates on Border Nodes. Since the set of all Lyapunov functions in the sense of Theorem 1 of any dynamical system forms a convex cone, it is possible to represent certain infinite sets of possible solutions by finitely many points in the parameter space, keeping track of only the corner points of a solution polytope. Consider a cycle C with only one border node b within a SCC. By computing an under-approximation of the set of all possible V_b that are feasible, according to the constraints of the cycle, we can abstract away all the non-border nodes of C. Such a computation is possible by employing a technique from multiobjective optimization, *normal-boundary intersection* [12]. A set of extremal points of the feasible set of V_b's parameters can be computed by repeatedly solving the optimization problem corresponding to the cycle with different objective functions (see Fig. 2). Generally, the number of optimization directions needed to achieve a sufficiently tight approximation depends on the system itself. Therefore, it is desirable to refine approximations on-the-fly, if needed. Techniques for iterative refinement are discussed in Section 3.4.

Theorem 3 (Cyclic Decomposition and Stability). *Let H be a hybrid automaton consisting of two subgraphs C_1 and C_2 with a single border node b. Let $b, n_1, \ldots, n_j$ be the nodes of C_1 and $b, m_1, \ldots, m_k$ be the nodes of C_2. Let $V_{b_1}, \ldots V_{b_m}$ be LLFs for b such that for each V_{b_i} there exist GLFs $V_{b_i}, V^i_{n_1}, \ldots, V^i_{n_j}$ for the entire subgraph C_1. If there exists a GLF $V_b, V_{m_1}, \ldots, V_{m_k}$ for subgraph C_2 with $\exists \lambda_1, \ldots, \lambda_m > 0 : V_b = \sum_i \lambda_i V_{b_i}$, then H is globally asymptotically stable.*

Proof. We need to prove that there exists a GLF for H, i.e., a family of V_m for all modes $m \in \mathcal{M}$, such that constr(H) is fulfilled. Per assumption, there

is a family of $V_b, V_{m_1}, \ldots, V_{m_k}$ *for subgraph* C_2 *fulfilling constr*(C_2)*, and additionally there exist* $\lambda_1, \ldots, \lambda_m > 0$ *such that* $V_b = \sum_i \lambda_i V_{b_i}$*. For all* i*, there exist* $V_{b_i}, V^i_{n_1}, \ldots, V^i_{n_j}$ *that fulfill constr*(C_1)*. Since the set of Lyapunov functions forms a convex cone, this implies that* $\sum_i \lambda_i V_{b_i}, \sum_i \lambda_i V^i_{n_1}, \ldots, \sum_i \lambda_i V^i_{n_j}$ *also fulfill constr*(C_1)*. This implies that* $\sum_i \lambda_i V_{b_i}, V_{m_1}, \ldots, V_{m_k}, \sum_i \lambda_i V^i_{n_1}, \ldots, \sum_i \lambda_i V^i_{n_j}$ *is the desired GLF for* H*.*

Remark 3. Since $V_b = \sum_i \lambda_i V_{b_i}$ already implies that V_b satisfies the Lyapunov constraints for node b, these constraints can be dropped for cycle C_2.

Representing the under-approximation of the solution set by a predicate corresponding to a conic polytope has one additional advantage: it is straightforward to add this constraint to an LMI problem for a neighboring cycle. Since a function for the border node b whose paremeters fulfill the conic predicate must already satisfy the Lyapunov properties for b, the Lyapunov constraints can be replaced by the polytopic constraint. The λ_i become the new free parameters for V_b, with the additional restriction that they must be non-negative. This can be interpreted as using a different parameterization for V_b, using the functions V_{b_i} instead of simple monomials. Therefore, the addition of a conic, polytopic constraint to a cycle does only incur local, simple changes to the LMI.

Splitting Border Nodes. If a cycle has two or more nodes that are also part of other cycles and have not been eliminated yet, then Theorem 3 is not directly applicable. In theory, one could produce analogous theorems that allow the under-approximation of the solution set for several border nodes. Since also interdependencies between the different solution sets need to be taken into account to achieve correctness, this would imply additional constraints on at least some edges of the constraint graph, greatly complicating the reduction.

A more straightforward approach is the application of simple transformations to eliminate border nodes from the graph: pick a border node with i incoming and j outgoing edges, where either $i > 1$ or $j > 1$. Split this node into $i \cdot j$ new nodes, each with exactly one incoming and one outgoing edge, such that

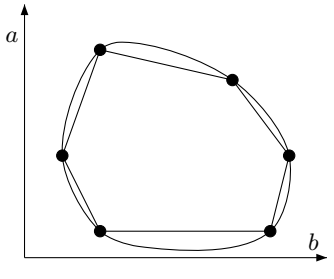


Fig. 2. Normal-boundary intersection for free Lyapunov function parameters a and b

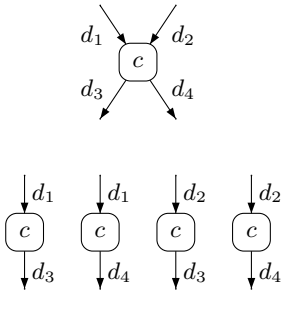
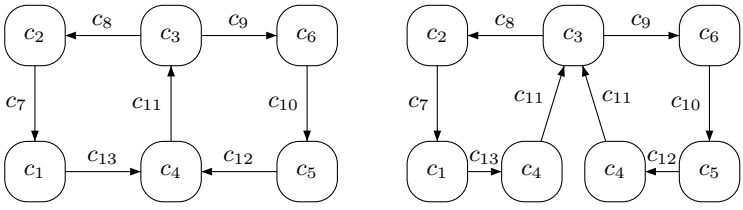


Fig. 3. Splitting a node



(a) before splitting (b) after splitting

Fig. 4. Eliminating border nodes through node splitting

all combinations of edges are represented. The new nodes and edges inherit the same constraints that were imposed on the old ones in the original constraint graph. This transformation will preserve the GAS property (see Fig. 3).

To see this, apply the transformation not only to the constraint graph, but also to the original hybrid automaton, preserving the differential equation and invariant of the old node on the new nodes and preserving the guards and update functions of all edges. The two systems are equivalent with respect to continuous behavior, i.e., every infinite trajectory of the old hybrid automaton is also a possible solution for the new automaton, because all combinations of incoming and outgoing edges are represented. Conversely, all infinite trajectories passing through one of the new modes can be mapped back on the single old mode. The transformation can produce additional spurious finite trajectory segments, since there is a non-determinstic choice between different newly introduced nodes with the same guard/update on their incoming edges. The outgoing edge that was taken in the original system might not exist as an outgoing edge for the non-deterministically chosen new node, resulting in a finite solution segment that cannot be further extended. However, since we only take infinite trajectories into account, these sprurious trajectory segments are of no consequence as far as stability per Definition 2 is concerned.

By repeated application of this transformation, it is possible to reduce the number of border nodes until obtaining a cycle that only contains a single border node (see Fig. 4). Clearly, Theorem 3 can then be applied to that cycle.

3.3 A Simple Example

We applied the proposed method to the example system given in Fig. 5(a), which represents a simple cruise control system. Here, GAS implies convergence of both the velocity differential v and the acceleration to 0. Engine and brake are both modelled having a saturation level representing a maximal acceleration and deceleration. Furthermore, the braking deceleration grows gradually instead of reaching its full effect immediately. Figure 5(b) shows the order of reduction and node splitting of the underlying graph, eventually resulting in a single cycle that is then reduced to the empty graph in a last step. We succeeded in computing a polytopic set of Lyapunov functions for the mode on the top left, such that existence

of a GLF for the whole system is guaranteed. This polytopic set can, for instance, be used to attach further braking modes to the top left mode, still maintaining stability. To achieve this, the newly added subgraphs must allow for a Lyapunov function for the top left mode that is also part of the polytopic set. The LMI for the bottom loop that is removed during the first step looks as follows:

$$\begin{aligned} V_j - \sum \lambda_i^j Q_i^j &\geq 0, j \in \{1,2\} \\ -A_j^T V_j - V_j A_j - \sum \mu_i^j Q_i^j &\geq 0, j \in \{1,2\} \\ V_1 - V_2 - \sum \nu_i^1 R_i^1 &\geq 0 \\ V_2 - V_1 - \sum \nu_i^2 R_i^2 &\geq 0 \\ \lambda_i^j, \mu_i^j, \nu_i^j &\geq 0 \end{aligned}$$

Here, V_1 and V_2 are the quadratic Lyapunov functions for the two modes of the loop, and the Q_i^j and R_i^j are S-Procedure terms representing the mode invariants and guards, respectively. Optimization in the directions of the free parameters (six directions in total) for V_2, the Lyapunov function for the border node, give us a polytopic set of possible Lyapunov functions with the following corner points (one function occured as the solution of two such optimization problems):

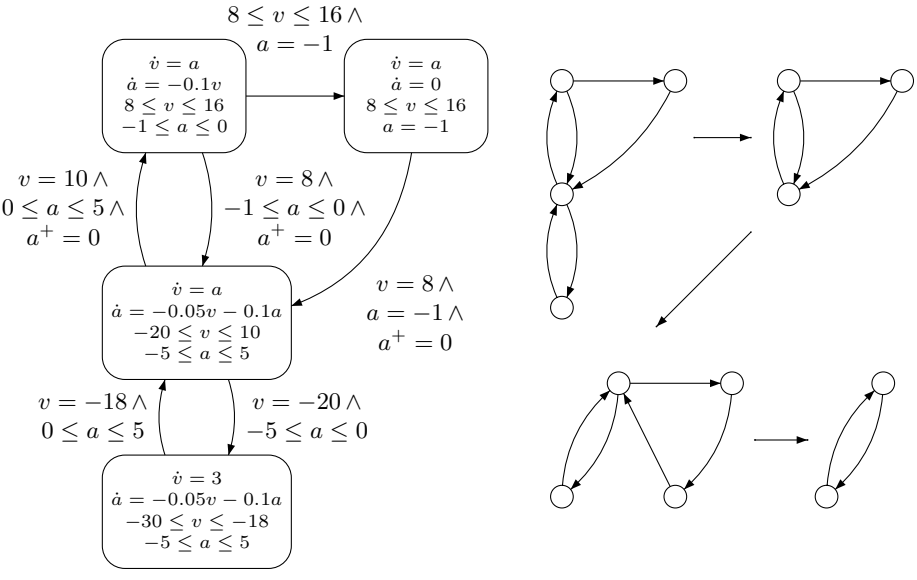
$$\begin{aligned} V_2^1 &= 7.5982v^2 + 12.903vt + 100t^2 \\ V_2^2 &= .625v^2 + 2vt + 10.5t^2 \\ V_2^3 &= 5.995v^2 + 19.9vt + 100t^2 \\ V_2^4 &= 2.532v^2 + 2vt + 48.641t^2 \\ V_2^5 &= 7.13v^2 + 13.936vt + 100t^2 \end{aligned}$$

These candidate functions are then used as the basis for the parameterized Lyapunov function for V_2 for the next loop to be reduced, i.e., the parameterized form is $\sum a_i V_2^i, a_i \geq 0$, where the a_i are the new unknown parameters.

3.4 Computational Issues

We now discuss various issues related to the implementation of the method proposed in this paper and additional decisions that need to be made.

Optimization Directions for the Normal-Boundary Intersection. Clearly, the choice of optimization direction for the normal-boundary intersection in the border nodes is important for the success of the procedure. Since one cannot assume knowledge about the shape of the feasible set at the beginning, it is reasonable to start with evenly spaced optimization directions. In our experiments, we found that choosing two optimization directions per parameter, one maximizing the parameter and one minimizing it, is a decent starting point, only requiring linearly many LMI problems to be solved. For systems with few parameters, more accurate approximations can be used as well, for instance optimizing in diagonal directions.



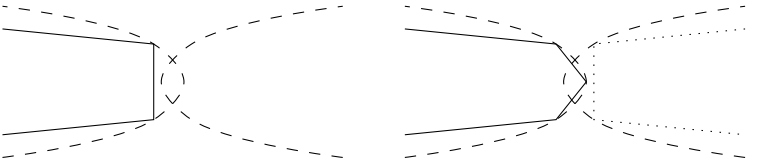
(a) Example automaton (b) Reduction order

Fig. 5. Cruise control example

Dealing with Unbounded Solution Sets. Since non-empty sets of Lyapunov functions for a system are always unbounded, one needs to bound the original problem for an extremal point to exist in all directions. This can be done by imposing an additional equality constraint on the Lyapunov functions or by adding bounds for each parameter. While the latter is somewhat conservative (some solutions might be excluded), we found it to be more stable numerically.

Numerical Inaccuracies. Convex optimization methods sometimes produce slightly infeasible "solutions" to a problem. In conjunction with normal-boundary intersection this is less of a problem, since one always computes multiple solution points. If a point is found to be slightly infeasible, it can be made feasible by shifting it slightly toward the polytope's center point via line search. The polytope's center point can be computed as the normed sum of all vertices.

Dealing with Conservativeness by Approximation Refinement. Whenever the polytopic under-approximation turns out to be too coarse, it is desirable to iteratively refine the predicate such that an existing solution is actually found. While it is possible to eventually find all solutions that do not lie on the boundary of some feasible set, by simply adding uniformly distributed new search directions, this will take a long time, especially if there are many free parameters for the Lyapunov function. We believe that heuristics for choosing a suitable optimization direction will be faster in most cases. For instance, consider two intersecting feasible sets induced on a single border node by two cycles (see Fig. 6). The feasible



(a) before refinement (b) after refinement

Fig. 6. Approximation refinement

sets (dashed) overlap, but the underapproximation for the left-hand constraint (solid) is too coarse, so that there is no intersection with the right-hand side feasible set. One first computes an under-approximation for the right hand cycle (dotted) without taking into account the left-hand side underapproximation. Then, for the left-hand cycle, one vertex is added to the under-approximation by optimizing in the direction given by the difference of the two center points of the polytopes. While there exist degenerate cases where this will not lead to improvement, this works well if the vertices are evenly spaced to start with. If such heuristics fail then one can still use a more general refinement scheme.

4 Conclusion

In this paper, we presented a decompositional approach for Lyapunov function computation of hybrid systems with complex state space. Contrary to the usual monolithical piecewise continuous Lyapunov function approach, we only solve local LMI problems. As only a limited number of modes are taken into account for each LMI problem, the chance of encountering numerical problems is reduced and the scalability of the method improved. Through a safe reduction scheme based on simple graph theoretic properties, we arrive at a sound method for proving the existence of a Lyapunov function for the entire system, thereby showing the system is GAS. Furthermore, in case of a failure, i.e., nonexistence of a Lyapunov function of the selected parametric form, it is easy to identify the part of hybrid automaton that causes the problem.

Apart from an implementation of a tool based on the proposed method, future work will encompass the exploitation of reachable set computations that allow further simplification of the resulting LMI problems. Furthermore, an interesting extension is the addition of probabilistic mode transitions, together with quantitative probabilistic stability properties.

References

1. Johansson, M., Rantzer, A.: On the computation of piecewise quadratic Lyapunov functions. In: 36th IEEE Conference on Decision and Control (CDC), pp. 3515–3520 (1997)
2. Pettersson, S., Lennartson, B.: Stability and robustness for hybrid systems. In: 35th IEEE Conference on Decision and Control (CDC), pp. 1202–1207 (1996)

3. Parrilo, P.A.: Semidefinite programming relaxations for semialgebraic problems. Mathematical Programming, Series B 96, 293–320 (2003)
4. Prajna, S., Papachristodoulou, A.: Analysis of switched and hybrid systems – beyond piecewise quadratic models. In: 22nd American Control Conference (ACC) (2003)
5. Papachristodoulou, A., Prajna, S.: Analysis of non-polynomial systems using the sums of squares decomposition. Positive Polynomials in Control, 23–43 (2005)
6. Chatterjee, D., Liberzon, D.: Stability analysis of deterministic and stochastic switched systems via a comparison principle and multiple Lyapunov functions. SIAM Journal on Control and Optimization 45(1), 174–206 (2006)
7. Vu, L., Liberzon, D.: Common Lyapunov functions for families of commuting nonlinear systems. Systems and Control Letters 54(5), 405–416 (2005)
8. Cai, C., Teel, A.R., Goebel, R.: Smooth Lyapunov functions for hybrid systems part II: (pre)asymptotically stable compact sets. IEEE Transactions on Automatic Control 53(3), 734–748 (2008)
9. Boyd, S., El Ghaoui, L., Feron, E., Balakrishnan, V.: Linear Matrix Inequalities in System and Control Theory. SIAM, Philadelphia (1994)
10. Pettersson, S.: Analysis and Design of Hybrid Systems. PhD thesis, Chalmers University of Technology, Gothenburg (1999)
11. Tarjan, R.: Depth-first search and linear graph algorithms. SIAM Journal on Computing 1(2), 146–160 (1972)
12. Das, I., Dennis, J.: Normal-boundary intersection: a new method for generating the Pareto surface in multicriteria optimization problems. SIAM J. Optimization 8, 631–657 (1998)

Existence of Periodic Orbits with Zeno Behavior in Completed Lagrangian Hybrid Systems

Yizhar Or[1] and Aaron D. Ames[2]

[1] Control and Dynamical Systems
California Institute of Technology, Pasadena, CA 91125
izi@cds.caltech.edu
[2] Department of Mechanical Engineering
Texas A&M University, College Station, TX
aames@tamu.edu

Abstract. In this paper, we consider hybrid models of mechanical systems undergoing impacts, *Lagrangian hybrid systems*, and study their periodic orbits in the presence of *Zeno behavior*—an infinite number of impacts occurring in finite time. The main result of this paper is explicit conditions under which the existence of stable periodic orbits for a Lagrangian hybrid system with *perfectly plastic impacts* implies the existence of periodic orbits in the same system with *non-plastic impacts*. Such periodic orbits contain phases of constrained and unconstrained motion, and the transition between them necessarily involves Zeno behavior. The result is practically useful for a wide range of unilaterally constrained mechanical systems under cyclic motion, as demonstrated through the example of a double pendulum with a mechanical stop.

1 Introduction

Periodic orbits play a fundamental role in the design and analysis of hybrid systems modeling a myriad of applications ranging from biological systems to chemical processes to robotics [25]. To provide a concrete example, bipedal robots are naturally modeled by hybrid systems [8,13]. The entire process of obtaining walking gaits for bipedal robots can be viewed simply as designing control laws that create stable periodic orbits in a specific hybrid system. This is a theme that is repeated throughout the various applications of hybrid systems [12].

In order to better understand the role that periodic orbits play in hybrid systems, we must first restrict our attention to hybrid systems that model a wide range of physical systems but are simple enough to be amenable to analysis. In this light, we consider *Lagrangian hybrid systems* modeling mechanical systems undergoing impacts; systems of this form have a rich history and are useful in a wide-variety of applications [5,20,26]. In particular, a *hybrid Lagrangian* consists of a configuration space, a Lagrangian modeling a mechanical systems, and a *unilateral constraint function* that gives the set of admissible configurations for this system. When the system's configuration reaches the boundary of its admissible region, the system undergoes an *impact event*, resulting in discontinuous velocity jump. The benefit of studying systems of this form is that they

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 291–305, 2009.

often display Zeno behavior (when an infinite number of impacts occur in a finite amount of time), so they give an ideal class of systems in which to gain an intuitive understanding of Zeno behavior and its relationship to periodic orbits in hybrid systems, which is the main focus of this paper.

Before discussing the type of periodic orbits that will be studied in this paper, we must first explain how one deals with Zeno behavior in Lagrangian hybrid systems by *completing* the hybrid model of these systems. Using the special structure of Lagrangian hybrid systems, the main observation is that points to which Zeno executions converge—*Zeno points*—must satisfy constraints imposed by the unilateral constraint function. These constraints are *holonomic* in nature, which implies that after the Zeno point, the hybrid system should switch to a holonomically constrained dynamical system evolving on the surface of zero level set of the constraint function. Moreover, if the force constraining the dynamical system to that surface becomes zero, there should be a switch back to the original hybrid system. These observations allow one to formally complete a Lagrangian hybrid system by adding an additional *post-Zeno* domain of constrained motion to the system [2,18].

In this paper, we study periodic orbits for completed Lagrangian hybrid systems, that pass through both the original and the *post-Zeno* domains of the hybrid system. Such periodic orbits are of paramount importance to a wide variety of applications, e.g., this is the type of orbits one obtains in bipedal robots. In particular, we begin by considering a *simple* periodic orbit which is an orbit that contains a single event of *perfectly plastic impact*. That is, after the impact, the system instantly switches to the post-Zeno domain. The key question is: *what happens to a simple periodic orbit when the impacts are not perfectly plastic*? The main result of this paper guarantees existence of a periodic orbit for completed Lagrangian hybrid system with non-plastic impacts given a stable periodic for the same system with plastic impacts; moreover, we give explicit bounds on the degree of plasticity that ensures the existence of such orbit.

The importance of the main result of this paper lies in the fact that impacts in mechanical systems are *never perfectly plastic*, so it is important to understand what happens to periodic orbits for perfectly plastic impacts in the case of non-plasticity. Using the example of a bipedal robot with knees [8,13,22], the knee locking (leg straightening) is modeled as a perfectly plastic impact. If one were to find a walking gait for this biped under this assumption, the main result of this paper would ensure that there would also be a walking gait in the case when the knee locking is not perfectly plastic, as would be true in reality. In light of this example, we conclude the paper by applying the main result of this paper to a double pendulum with a mechanical stop, which models a single leg of a bipedal robot with knees.

Both periodic orbits and Zeno behavior have been well-studied in the literature although they have yet to be studied simultaneously. With regard to Zeno behavior, it has been studied in the context of mechanical systems in [14,17] with results that complement the results of this paper, and studied for other hybrid models in [6,10,21,23,27]. Periodic orbits have primarily been studied in

hybrid systems in the context of bipedal locomotion for dynamic walking [8,9,15] and running [7], assuming perfectly plastic impacts. The pioneering work in [4] focuses on design of stable tracking control for cyclic tasks with Zeno behavior in Lagrangian hybrid systems, assuming that the system is *fully actuated*, i.e., all degrees-of-freedom are controlled. Note, however, that this assumption generally does not hold for locomotion systems, which are essentially *underactuated*.

2 Lagrangian Hybrid Systems

In this section, we introduce the notion of a hybrid Lagrangian and the associated Lagrangian hybrid system. Hybrid Lagrangians of this form have been studied in the context of Zeno behavior and reduction; see [1] and [17]. We begin this section by reviewing the notion of a simple hybrid system.

Definition 1. *A simple hybrid system is a tuple* $\mathscr{H} = (D, G, R, f)$*, where*

- D *is a smooth manifold called the domain,*
- G *is an embedded submanifold of* D *called the guard,*
- R *is a smooth map* $R : G \to D$ *called the reset map,*
- f *is a vector field on the manifold* D*.*

Hybrid executions. A *hybrid execution* of a simple hybrid system $\mathscr{H}$ is a tuple $\chi = (\Lambda, \mathcal{I}, \mathcal{C})$, where

- $\Lambda = \{0, 1, 2, \ldots\} \subseteq \mathbb{N}$ is an indexing set.
- $\mathcal{I} = \{I_i\}_{i\in\Lambda}$ is a *hybrid interval* where $I_i = [t_i, t_{i+1}]$ if $i, i+1 \in \Lambda$ and $I_{N-1} = [t_{N-1}, t_N]$ or $[t_{N-1}, t_N)$ or $[t_{N-1}, \infty)$ if $|\Lambda| = N$, N finite. Here, $t_i, t_{i+1}, t_N \in \mathbb{R}$ and $t_i \leq t_{i+1}$.
- $\mathcal{C} = \{c_i\}_{i\in\Lambda}$ is a collection of integral curves of f, i.e., $\dot{c}_i(t) = f(c_i(t))$ for $t \in I_i$, $i \in \Lambda$,

And the following conditions hold for every $i, i+1 \in \Lambda$:

$$\begin{array}{ll} \text{(i)} & c_i(t_{i+1}) \in G, \\ \text{(ii)} & R(c_i(t_{i+1})) = c_{i+1}(t_{i+1}), \\ \text{(iii)} & t_{i+1} = \min\{t \in I_i : c_i(t) \in G\}. \end{array}$$

The *initial condition* for the hybrid execution is $c_0(t_0)$.

Lagrangians. Let $q \in \mathbb{R}^n$ be the *configuration* of a mechanical system[1]. In this paper, we will consider Lagrangians, $L : \mathbb{R}^{2n} \to \mathbb{R}$, describing mechanical, or robotic, systems, which are Lagrangians of the form $L(q, \dot{q}) = \frac{1}{2}\dot{q}^T M(q)\dot{q} - V(q)$, where $M(q)$ is the (positive definite) inertial matrix, $\frac{1}{2}\dot{q}^T M(q)\dot{q}$ is the kinetic energy and $V(q)$ is the potential energy. We will also consider a *control law* $u(q, \dot{q})$, which is a given smooth function $u : \mathbb{R}^{2n} \to \mathbb{R}^n$. In this case, the Euler-Lagrange equations yield the (unconstrained, controlled) equations of motion for the system:

[1] For simplicity, we assume that the configuration space is identical to $\mathbb{R}^n$

$$M(q)\ddot{q} + C(q,\dot{q}) + N(q) = u(q,\dot{q}), \tag{1}$$

where $C(q,\dot{q})$ is the vector of centripetal and Coriolis terms (cf. [16]) and $N(q) = \frac{\partial V}{\partial q}(q)$. Defining the *state* of the system as $x = (q,\dot{q})$, the Lagrangian vector field, f_L, associated to L takes the familiar form:

$$\dot{x} = f_L(x) = \begin{pmatrix} \dot{q} \\ M(q)^{-1}(-C(q,\dot{q}) - N(q) + u(q,\dot{q})) \end{pmatrix}. \tag{2}$$

This process of associating a dynamical system to a Lagrangian will be mirrored in the setting of hybrid systems. First, we introduce the notion of a hybrid Lagrangian.

Definition 2. *A simple hybrid Lagrangian is defined to be a tuple* $\mathbf{L} = (Q, L, h)$, *where*

- Q *is the configuration space (assumed to be identical to* $\mathbb{R}^n$*),*
- $L : TQ \to \mathbb{R}$ *is a hyperregular Lagrangian,*
- $h : Q \to \mathbb{R}$ *provides a unilateral constraint on the configuration space; we assume that the zero level set* $h^{-1}(0)$ *is a smooth manifold.*

Simple Lagrangian hybrid systems. For a given Lagrangian, there is an associated dynamical system. Similarly, given a hybrid Lagrangian $\mathbf{L} = (Q, L, h)$ the *simple Lagrangian hybrid system* associated to $\mathbf{L}$ is the simple hybrid system $\mathscr{H}_{\mathbf{L}} = (D_{\mathbf{L}}, G_{\mathbf{L}}, R_{\mathbf{L}}, f_{\mathbf{L}})$. First, we define

$$\begin{aligned} D_{\mathbf{L}} &= \{(q,\dot{q}) \in TQ : h(q) \geq 0\}, \\ G_{\mathbf{L}} &= \{(q,\dot{q}) \in TQ : h(q) = 0 \text{ and } dh(q)\dot{q} \leq 0\}, \end{aligned}$$

where $dh(q) = [\frac{\partial h}{\partial q}(q)]^T = [\,\frac{\partial h}{\partial q_1}(q) \cdots \frac{\partial h}{\partial q_n}(q)\,]$. In this paper, we adopt the reset map ([5]) $R_{\mathbf{L}}(q,\dot{q}) = (q, P_{\mathbf{L}}(q,\dot{q}))$, which is based on the *impact equation*

$$P_{\mathbf{L}}(q,\dot{q}) = \dot{q} - (1+e)\frac{dh(q)\dot{q}}{dh(q)M(q)^{-1}dh(q)^T}M(q)^{-1}dh(q)^T, \tag{3}$$

where $0 \leq e \leq 1$ is the *coefficient of restitution*, which is a measure of the energy dissipated through impact. This reset map corresponds to rigid-body collision under the assumption of *frictionless impact*. Examples of more complicated collision laws that account for friction can be found in [5] and [24]. Finally, $f_{\mathbf{L}} = f_L$ is the Lagrangian vector field associated to L in (2).

3 Zeno Behavior and Completed Hybrid Systems

In this section we define Zeno behavior in Lagrangian hybrid systems, introduce the notion of a completed hybrid system ([2,18]), and define the notions of simple periodic orbit and Zeno periodic orbit, corresponding to periodic completed executions under plastic and non-plastic impacts. Then we define the stability of periodic orbits.

Zeno behavior. A hybrid execution χ is *Zeno* if $\Lambda = \mathbb{N}$ and $\lim_{i\to\infty} t_i = t_\infty < \infty$. Here t_∞ is called the *Zeno time*. If χ is a Zeno execution of a Lagrangian hybrid system $\mathscr{H}_{\mathbf{L}}$, then its *Zeno point* is defined to be

$$x_\infty = (q_\infty, \dot{q}_\infty) = \lim_{i\to\infty} c_i(t_i) = \lim_{i\to\infty} (q_i(t_i), \dot{q}_i(t_i)).$$

These limit points essentially lie on the *constraint surface* in state space, which is defined by $\mathcal{S} = \{(q, \dot{q}) \in \mathbb{R}^{2n} : h(q) = 0 \text{ and } dh(q)\dot{q} = 0\}$.

Constrained dynamical systems. We now define the holonomically constrained dynamical system $\mathscr{D}_{\mathbf{L}}$ associated with the hybrid Lagrangian $\mathbf{L}$. For such systems, the constrained equations of motion can be obtained from the equations of motion for the unconstrained system (1), and are given by (cf. [16])

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + N(q) = dh(q)^T\lambda + u(q, \dot{q}), \tag{4}$$

where λ is the Lagrange multiplier which represents the contact force. Differentiating the constraint equation $h(q) = 0$ twice with respect to time and substituting the solution for $\ddot{q}$ in (4), the solution for the constraint force λ is obtained as follows:

$$\begin{aligned}\lambda(q, \dot{q}) = &\left(dh(q)M(q)^{-1}dh(q)^T\right)^{-1} \\ &\left(dh(q)M(q)^{-1}(C(q, \dot{q})\dot{q} + N(q) - u(q, \dot{q})) - \dot{q}^T H(q)\dot{q}\right).\end{aligned} \tag{5}$$

From the constrained equations of motion, for $x = (q, \dot{q})$, we get the vector field

$$\dot{x} = \tilde{f}_L(x) = \begin{pmatrix} \dot{q} \\ M(q)^{-1}(-C(q, \dot{q})\dot{q} - N(q) + u(q, \dot{q}) + dh(q)^T\lambda(q, \dot{q})) \end{pmatrix}$$

Note that $\tilde{f}_L$ defines a vector field on the manifold $TQ|_{h^{-1}(0)}$, from which we obtain the dynamical system $\mathscr{D}_{\mathbf{L}} = (T\mathcal{S}, \tilde{f}_L)$. For this dynamical system, $q(t)$ slides along the constraint surface $\mathcal{S}$ as long as the constraint force λ is positive.

A *constrained execution* $\tilde{\chi}$ of $\mathscr{D}_{\mathbf{L}}$ is a pair $(\tilde{I}, \tilde{c})$ where $\tilde{I} = [\tilde{t}_0, \tilde{t}_f] \subset \mathbb{R}$ if $\tilde{t}_f$ is finite and $\tilde{I} = [\tilde{t}_0, \tilde{t}_f) \subset \mathbb{R}$ if $\tilde{t}_f = \infty$, and $\tilde{c} : \tilde{I} \to TQ$, with $\tilde{c}(t) = (q(t), \dot{q}(t))$ a solution to the dynamical system $\mathscr{D}_{\mathbf{L}}$ satisfying the following properties:

$$\begin{aligned} &\text{(i)} && h(q_0(\tilde{t}_0)) = 0, \\ &\text{(ii)} && dh(q_0(\tilde{t}_0))\dot{q}_0(\tilde{t}_0) = 0, \\ &\text{(iii)} && \lambda(q(\tilde{t}_0), \dot{q}(\tilde{t}_0)) > 0, \\ &\text{(iv)} && \tilde{t}_f = \min\{t \in \tilde{I} : \lambda(q(t), \dot{q}(t)) = 0\}. \end{aligned} \tag{6}$$

Using the notation and concepts introduced thus far, we introduce the notion of a completed hybrid system.

Definition 3. *If* $\mathbf{L}$ *is a simple hybrid Lagrangian and* $\mathscr{H}_{\mathbf{L}}$ *the corresponding Lagrangian hybrid system, the corresponding completed Lagrangian hybrid system*[2] *is defined to be:*

[2] As was orginally pointed out in [2], this terminology (and notation) is borrowed from topology, where a metric space can be completed to ensure that "limits exist."

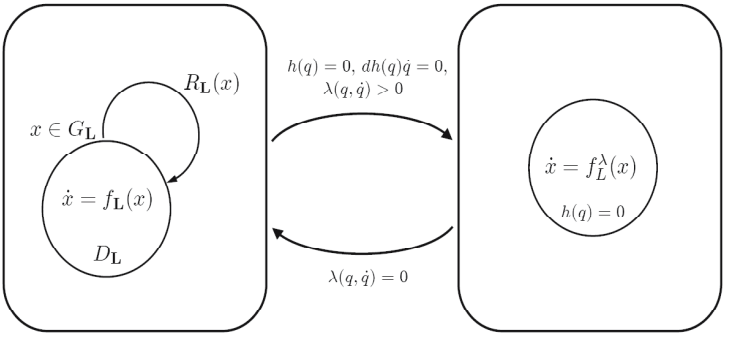


Fig. 1. A graphical representation of a completed hybrid system

$$\overline{\mathscr{H}}_{\mathbf{L}} := \begin{cases} \mathscr{D}_{\mathbf{L}} & \text{if} \quad h(q) = 0\ ,\ dh(q)\dot{q} = 0, \text{ and } \lambda(q, \dot{q}) > 0 \\ \mathscr{H}_{\mathbf{L}} & \text{otherwise.} \end{cases}$$

Remarks. The system $\overline{\mathscr{H}}_{\mathbf{L}}$ can be viewed simply as a hybrid system with two domains; in this case, the reset maps are the identity, and the guards are given as in Fig. 1. Also note that the only way for the transition to be made from the hybrid system $\mathscr{H}_{\mathbf{L}}$ to the constrained system $\mathscr{D}_{\mathbf{L}}$ is if a specific Zeno execution reaches its Zeno point. Second, a transition for $\mathscr{D}_{\mathbf{L}}$ to $\mathscr{H}_{\mathbf{L}}$ happens when the constraint force λ crosses zero. Finally, it is shown in [18] that the constraint acceleration $\ddot{h}(q, \dot{q})$ and the constraint force $\lambda(q, \dot{q})$ in (5) satisfy *complementarity relation*. That is, while sliding along the constraint surface $\mathcal{S}$, either $\ddot{h} = 0$ and $\lambda > 0$, corresponding to maintaining constrained motion, or $\ddot{h} > 0$ and $\lambda = 0$, corresponding to leaving the constraint surface and switching back to the hybrid system. Thus, the definition of the completed hybrid system is consistent.

The completed execution. Having introduced the notion of a completed hybrid system, we must introduce the semantics of solutions of systems of this form. That is, we must introduce the notion of a *completed execution*.

Definition 4. *Given a simple hybrid Lagrangian* $\mathbf{L}$ *and the associated completed system* $\overline{\mathscr{H}}_{\mathbf{L}}$*, a* completed execution $\overline{\chi}$ *is a (possibly infinite) ordered sequence of alternating constrained and hybrid executions* $\overline{\chi} = \{\tilde{\chi}^{(1)},\ \chi^{(2)},\ \tilde{\chi}^{(3)},\ \chi^{(4)},\ ...\}$*, with* $\tilde{\chi}^{(i)}$ *and* $\chi^{(j)}$ *executions of* $\mathscr{D}_{\mathbf{L}}$ *and* $\mathscr{H}_{\mathbf{L}}$*, respectively, that satisfy the following conditions:*

(i) For each pair $\tilde{\chi}^{(i)}$ *and* $\chi^{(i+1)}$*,*

$$\tilde{t}_f^{(i)} = t_0^{(i+1)} \text{ and } \tilde{c}^{(i)}(\tilde{t}_f^{(i)}) = c_0^{(i+1)}(t_0^{(i+1)}).$$

(ii) For each pair $\chi^{(i)}$ *and* $\tilde{\chi}^{(i+1)}$*,*

$$t_\infty^{(i)} = \tilde{t}_0^{(i+1)} \text{ and } c_\infty^{(i)} = \tilde{c}^{(i+1)}(\tilde{t}_0^{(i+1)}).$$

where the superscript (i) *denotes values corresponding to the* i^{th} *execution in* $\overline{\chi}$*, and* $t_\infty^{(i)}, c_\infty^{(i)}$ *denote the Zeno time and Zeno point associated with the* i^{th} *hybrid execution* $\chi^{(i)}$*.*

Periodic orbits of completed hybrid systems. In the special case of plastic impacts $e = 0$, a *simple periodic orbit* is a completed execution $\overline{\chi}$ with initial condition $\tilde{c}^{(1)}(0) = x^*$ that satisfies $\tilde{c}^{(3)}(\tilde{t}_0^{(3)}) = x^*$. The period of $\overline{\chi}$ is $T = \tilde{t}_0^{(3)}$. In other words, this orbit consists of a constrained execution starting at x^*, followed by a hybrid (unconstrained) execution which is ended by a single plastic collision at $t = T$, that resets the state back to x^*.

For non-plastic impacts $e > 0$, a *Zeno periodic orbit* is a completed execution $\overline{\chi}$ with initial condition $\tilde{c}^{(1)}(0) = x^*$ that satisfies $c_\infty^{(2)} = \tilde{c}^{(3)}(\tilde{t}_0^{(3)}) = x^*$. The period of $\overline{\chi}$ is $T = t_\infty^{(2)} = \tilde{t}_0^{(3)}$. In other words, this orbit consists of a constrained execution starting at x^*, followed by a Zeno execution with infinite number of non-plastic impacts, which converges in finite time back to x^*.

Stability of hybrid periodic orbits. We now define the stability of hybrid periodic orbits.

Definition 5. *A Zeno (or simple) periodic orbit* $\overline{\chi} = \{\tilde{\chi}^{(1)}, \chi^{(2)}, \tilde{\chi}^{(3)}, \chi^{(4)}, ...\}$ *with initial condition* $x^* \in \mathcal{S}$ *is* locally exponentially stable *if there exist a neighborhood* $U \subset \mathcal{S}$ *of* x^* *and a scalar* $\gamma \in (0, 1)$ *such that for any initial condition* $x_0 = \tilde{c}^{(1)}(0) \in U$*, the resulting completed execution satisfies* $\|\tilde{c}^{(2k+1)}(\tilde{t}_0^{(2k+1)}) - x^*\| \leq \|x_0 - x^*\|\gamma^k$ *for* $k = 1, 2, \ldots$.

Choice of coordinates. In the rest of this paper, we assume that the generalized coordinates contain the constraint function h as a coordinate, i.e. $q = (z, h)$. This assumption is quite general, since a transformation to such coordinate set must exist, at least locally, due to the regularity of $h(q)$. The state of the system thus takes the form $x = (z, h, \dot{z}, \dot{h}) \in I\!R^{2n}$. When the coordinates take this special form, the reset map (3) simplifies to

$$P_{\mathbf{L}}(q, \dot{q}) = \begin{pmatrix} \dot{z} - (1+e)\dot{h}\eta(z) \\ -e\dot{h} \end{pmatrix}, \text{ where } \eta(z) = \left. \frac{[M^{-1}(q)]_{1\ldots n-1,n}}{[M^{-1}(q)]_{n,n}} \right|_{h=0}. \quad (7)$$

The instantaneous solution for the accelerations $\ddot{q}$ in (1) is given by

$$\ddot{q}(q, \dot{q}) = (\ddot{z}(q, \dot{q}), \ddot{h}(q, \dot{q})) = M(q)^{-1} \left(u(q, \dot{q}) - C(q, \dot{q}) - N(q)\right). \quad (8)$$

4 Main Result

In this section we present the main result of this paper, namely, conditions under which the existence and stability of a simple periodic orbit imply existence of a Zeno periodic orbit.

4.1 Statement of Main Result

Before stating this result, some preliminary setup is needed. We can write $x^* = (z^*, 0, \dot{z}^*, 0)$, and define three types of neighborhoods of x^* in three different

subspaces of $\mathbb{R}^{2n}$. For $\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4 > 0$, the neighborhoods $\Omega_1(\epsilon_1)$, $\Omega_2(\epsilon_1, \epsilon_2)$ and $\Omega_4(\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4)$ are defined as follows.

$$\begin{aligned}
\Omega_1(\epsilon_1) &= \{(q,\dot{q}) : \ h = 0,\ \dot{h} = 0,\ \text{and } \|z - z^*\| < \epsilon_1\} \\
\Omega_2(\epsilon_1, \epsilon_2) &= \{(q,\dot{q}) : \ h = 0,\ \dot{h} = 0,\ \|z - z^*\| < \epsilon_1, \text{ and } \|\dot{z} - \dot{z}^*\| < \epsilon_2\} \\
\Omega_4(\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4) &= \{(q,\dot{q}) : \ \|z - z^*\| < \epsilon_1,\ \|\dot{z} - \dot{z}^*\| < \epsilon_2,\ 0 < h < \epsilon_3, \text{ and } |\dot{h}| < \epsilon_4\}
\end{aligned}$$

Assume we are given a control law $u(q,\dot{q})$ and a starting point $x^* \in \mathcal{S}$ for which there exists of a simple periodic periodic orbit $\overline{\chi}^*$ starting at x^* which is locally exponentially stable. Define $v^* = \left|\dot{h}_0^{(2)}(t_1^{(2)})\right|$, which is the pre-collision velocity at the single (plastic) collision in the periodic orbit. The following assumption is a direct implication of the stability of $\overline{\chi}^*$:

Assumption 1. *Assume that there exist $\epsilon_1, \epsilon_2 > 0, \kappa \geq 1$ and $\gamma \in (0,1)$, such that for any initial condition $x_0 \in \Omega_2(\epsilon_1, \epsilon_2)$, the corresponding completed execution with $e = 0$ satisfies the two following requirements:*

$$\begin{aligned}
&(a) \qquad \tilde{c}^{(3)}(\tilde{t}_0^{(3)}) \in \Omega_2(\gamma\epsilon_1, \gamma\epsilon_2) \\
&(b) \qquad \left|\dot{h}_0(t_1^{(2)})\right| < \kappa v^*.
\end{aligned} \tag{9}$$

Setup. To provide the conditions needed for the main result, for the given ϵ_1, ϵ_2 and κ, let the neighborhood Ω be defined as $\Omega = \Omega_4(\epsilon_1, \epsilon_2, \epsilon_3, \kappa v^*)$ for some $\epsilon_3 > 0$, and define the following scalars:

$$\begin{aligned}
a_{min} &= -\max_{(q,\dot{q}) \in \Omega} \ddot{h}(q,\dot{q}) \\
a_{max} &= -\min_{(q,\dot{q}) \in \Omega} \ddot{h}(q,\dot{q}) \\
\delta &= \sqrt{\left|\frac{a_{max}}{a_{min}}\right|} \\
\dot{z}_{max} &= \|\dot{z}^*\| + \epsilon_2 \\
\ddot{z}_{max} &= \max_{(q,\dot{q}) \in \Omega} \|\ddot{z}(q,\dot{q})\| \\
\eta_{max} &= \max_{z \in \Omega_1(\epsilon_1)} \|\eta(z)\|.
\end{aligned} \tag{10}$$

The following theorem establishes sufficient conditions for existence of a Zeno periodic orbit given a simple periodic orbit.

Theorem 1. *Consider a simple periodic orbit $\overline{\chi}^*$ which is locally exponentially stable, and the given $\epsilon_1, \epsilon_2 > 0$, $\kappa \geq 1$, and $\gamma \in (0,1)$ that satisfy Assumption 1. Then for a given coefficient of restitution e, if the neighborhood Ω and its associated scalars defined in (10) satisfy the following conditions:*

$$a_{max} \geq a_{min} > 0 \tag{11}$$

$$e\delta < 1 \tag{12}$$

$$\frac{2e\kappa v^*}{a_{min}(1-\delta e)}\dot{z}_{max} \leq \epsilon_1(1-\gamma) \tag{13}$$

$$\left(\frac{1+\delta}{1-\delta e}\eta_{max} + \frac{2}{a_{min}(1-\delta e)}\ddot{z}_{max}\right) e\kappa v^* \leq \epsilon_2(1-\gamma), \tag{14}$$

$$\frac{(e\kappa v^*)^2}{2a_{min}} \leq \epsilon_3 \tag{15}$$

then there exists a Zeno periodic orbit with initial condition within $\Omega_2(\epsilon_1, \epsilon_2)$.

4.2 Proof of the Main Result

Before proving Theorem 1, we must define some preliminary notation. Consider the completed execution $\overline{\chi}$ with $e = 0$, and the execution $\overline{\chi}'$ with $e > 0$ under the same given initial condition $x_0 \in \Omega_2(\epsilon_1, \epsilon_2)$. Since we are only interested in the first hybrid and constrained elements of $\overline{\chi}$ and $\overline{\chi}'$, we simplify the notation by defining $\overline{\chi} = \{\tilde{\chi},\ \chi, ...\}$ and $\overline{\chi}' = \{\tilde{\chi}',\ \chi', ...\}$. Since the constrained motion does not contain any collisions, it is clear that $\tilde{\chi} = \tilde{\chi}'$. Moreover, the hybrid executions χ and χ' are also identical until the first collision time, that is $c_0(t) = c_0'(t)$ for $t \in [t_0, t_1]$ and $t_1 = t_1'$. Therefore, we will compare the solutions $c_i'(t)$ and $c_i(t)$ for $i > 1$, i.e. after the time t_1.

We now give the outline of the proof, which is divided into three steps. The first step proves that if the hybrid execution χ' stays within the neighborhood Ω, then conditions (11) and (12) imply that it is a Zeno execution. Step 2 verifies that under conditions (13) and (14), the execution χ' actually stays within Ω. The results of these two steps are stated as two lemmas, whose detailed proofs are relegated to [19] due to space constraints. Finally, the third step utilizes the two previous steps to complete the proof of Theorem 1.

Step 1. Consider a neighborhood Ω that satisfies conditions (11) and (12), and assume that the trajectory of the hybrid execution χ' satisfies $c_i'(t) \in \Omega$ for all $t \in I_i'$, $i \in \Lambda' \setminus \{0\}$. This assumption implies that the h-component of $c_i'(t) = (z_i'(t), h_i'(t), \dot{z}_i'(t), \dot{h}_i'(t))$ satisfies the second-order differential inclusion

$$\ddot{h}_i'(t) \in [-a_{max}, -a_{min}], \tag{16}$$

for all $t \in I_i'$, $i \in \Lambda' \setminus \{0\}$. At each collision time t_i', $i \in \Lambda' \setminus \{0\}$, (16) is re-initialized according to the collision law (7) as

$$\dot{h}_{i+1}'(t_i') = -e\dot{h}_i'(t_i'), \text{ and } h_{i+1}'(t_i') = h_i'(t_i') = 0. \tag{17}$$

Let $\tau_i = t_{i+1}' - t_i'$, which is the time difference between consecutive collisions, and let $v_i = -\dot{h}_{i-1}'(t_i')$, which is the pre-collision velocity at time t_i'. The following lemma summarizes results on the hybrid execution χ' under the differential inclusion (16).

Lemma 1 ([19]). *Assume that the hybrid execution χ' satisfies the differential inclusion (16) for all $t \in I'_i$, $i \in \Lambda' \setminus \{0\}$, and that a_{min}, a_{max}, and δ satisfy conditions (11) and (12). Then χ' is a Zeno execution with a Zeno time t_∞. Moreover, the solution $c'_i(t)$ satisfies the following for all $i \geq 1$*

$$v_i \leq v_1(e\delta)^{i-1} \tag{18}$$

$$\left|\dot{h}'_i(t)\right| \leq v_1 \text{ for all } t \in I'_i \tag{19}$$

$$\tau_i \leq \frac{2ev_1}{a_{min}}(e\delta)^{i-1} \tag{20}$$

$$t'_\infty - t'_1 \leq \frac{2ev_1}{a_{min}(1-e\delta)} \tag{21}$$

$$h'_i(t) \leq \frac{e^2 v_1^2}{2a_{min}} \text{ for all } t \in I'_i. \tag{22}$$

The key idea in the proof is utilization of optimal control theory to find the "most unstable" execution under the differential inclusion (16) and the impact law (17). It is shown in [19] that all possible executions satisfy the bound $v_{i+1} \leq e\delta v_i$. Therefore, condition (12) implies that the v_i-s are bounded by the decaying geometric series (18). All other bounds in (19)-(22) are then implied by (18).

Step 2: We now verify that for any initial condition in $\Omega_2(\epsilon_1, \epsilon_2)$, the solution actually stays within Ω, as summarized in the following lemma.

Lemma 2 ([19]). *Consider a neighborhood Ω that satisfies conditions (11)-(14). Then for any initial condition $x_0 \in \Omega_2(\epsilon_1, \epsilon_2)$, the hybrid execution χ' is a Zeno execution that satisfies $c'_i(t) \in \Omega$ for all $t \in I'_i$, $i \in \Lambda' \setminus \{0\}$.*

The main idea of the proof in [19], is to assume that the execution initially stays within the neighborhood Ω, and use (18)-(22) to find bounds on $q(t), \dot{q}(t)$ during the execution. Then, conditions (11)-(15) guarantee that the execution does not leave Ω at all times.

Step 3: We now utilize Lemma 1 and Lemma 2 to prove the main result.

Proof (of Theorem 1). Consider the completed execution $\overline{\chi}' = \{\tilde{\chi}', \chi', ...\}$ with $e > 0$, under initial condition $x_0 \in \Omega_2(\epsilon_1, \epsilon_2)$. Lemma 1 and Lemma 2 imply that χ' is a Zeno execution which reaches $\mathcal{S}$ in time t_∞, and that $c'_i(t) \in \Omega$ for all $i \geq 1$. Define the function $\Phi : \Omega_2(\epsilon_1, \epsilon_2) \to \mathcal{S}$ as $\Phi(x_0) = c'_\infty$, under initial condition $c'_0(0) = x_0$. Note that Φ is well-defined, since for any initial condition within $\Omega_2(\epsilon_1, \epsilon_2)$, a Zeno execution is guaranteed. Moreover, since the limit point satisfies $c'_\infty \in \Omega \cap \mathcal{S} = \Omega_2(\epsilon_1, \epsilon_2)$, Φ maps $\Omega_2(\epsilon_1, \epsilon_2)$ onto itself. The continuity of the hybrid flow with respect to its initial condition, which is a fundamental property of a completed hybrid system with a single constraint (cf. [5]) implies that Φ is continuous. Invoking the *fixed point theorem* (cf. [11]), we conclude that there exists a fixed point $\bar{x} \in \Omega_2(\epsilon_1, \epsilon_2)$ such that $\Phi(\bar{x}) = \bar{x}$. Finally, the definition of Φ then implies that $\bar{x}$ corresponds to the starting point of a Zeno periodic orbit with period $T' = t'_\infty$.

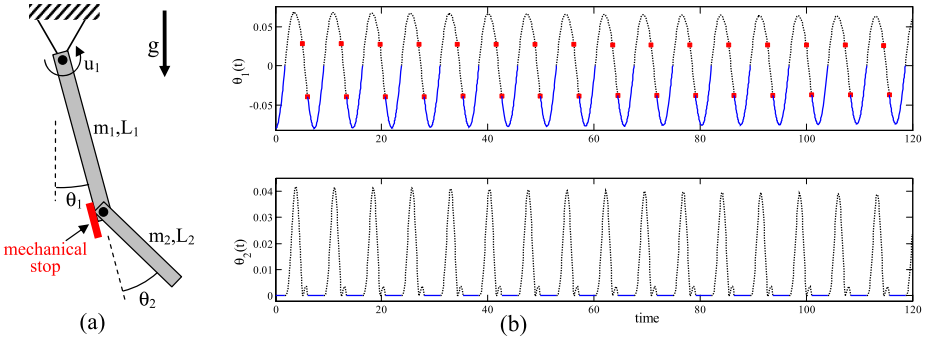


Fig. 2. (a) The constrained double pendulum system (b) Time plots of the solution $\theta_1(t)$ and $\theta_2(t)$ of the double pendulum with no actuation under plastic collisions

5 Simulation Example

This section demonstrates the theoretical results on a constrained double pendulum, which is depicted in Figure 2(a). The double pendulum consists of two rigid links of masses m_1, m_2, lengths L_1, L_2, and uniform mass distribution, which are attached by revolute joints, while a mechanical stop dictates the range of motion of the lower link. The upper joint is actuated by a torque u_1, while the lower joint is passive. This example serves as a simplified model of a leg with a passive knee and a mechanical stop.

The configuration of the double pendulum is $q = (\theta_1, \theta_2)$, and the constraint that represents the mechanical stop is given by $h(q) = \theta_2 \geq 0$. Note that in that case the coordinates are already in the form $q = (z, h)$, where $z = \theta_1$ and $h = \theta_2$. The Lagrangian of the system is given by $L(q, \dot{q}) = \frac{1}{2}\dot{q}^T M(q)\dot{q} + (\frac{1}{2}m_1 L_1 + m_2 L_1)g\cos\theta_1 + \frac{1}{2}m_2 L_2 g\cos(\theta_1 + \theta_2)$, with the elements of the 2×2 inertia matrix $M(q)$ given by $M_{11} = m_1 L_1^2/3 + m_2(L_1^2 + L_2^2/3 + L_1 L_2\cos\theta_2)$, $M_{12} = M_{21} = m_2(3L_1 L_2\cos\theta_2 + 2L_2^2)/6$, $M_{22} = m_2 L_2^2/3$. The values of parameters for the simulations were chosen as $m_1 = m_2 = L_1 = L_2 = g = 1$.

The first running simulation shows the motion of the *uncontrolled system* i.e. $u_1 = 0$, under *plastic collisions*, i.e. $e = 0$. Fig. 2(b) shows the time plots of $\theta_1(t)$ and $\theta_2(t)$ under initial condition $q(0) = (-0.08, 0)$ and $\dot{q}(0) = (0, 0)$. The parts of unconstrained motion appear as dashed curves, and the parts of constrained motion appear as solid curves. The points of collision events are marked with squares ('■') on the curve of $\theta_1(t)$. The double pendulum exhibits a slightly decaying periodic-like motion with *two* plastic collisions per cycle. At each cycle, after the first plastic collision, the constraint force λ required to maintain the constraint $\theta_2 = 0$ is negative. Thus, the lower link instantaneously detaches to another phase of unconstrained motion, until a second plastic collision occurs. After the second collision, the lower link locks at $\theta_2 = 0$, and the pendulum switches to a constrained motion with positive constraint force $\lambda > 0$ for some finite time, until λ crosses zero, and the lower link detaches again.

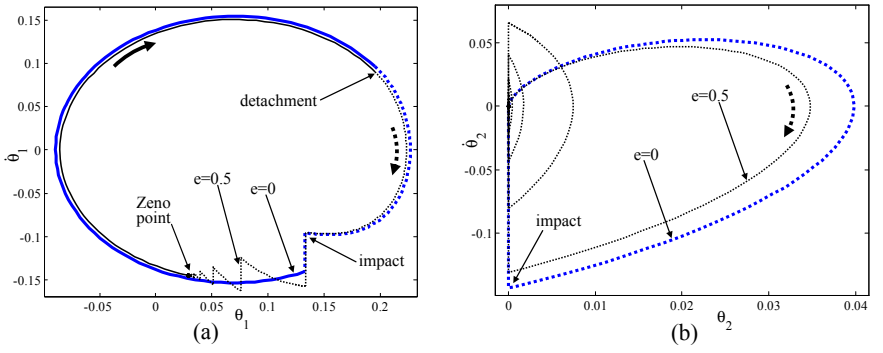


Fig. 3. Phase portraits of the periodic orbit in (a)$(\theta_1, \dot{\theta}_1)$-plane and (b)$(\theta_2, \dot{\theta}_2)$- plane for $e = 0$ (thin black) and $e = 0.5$ (thick blue)

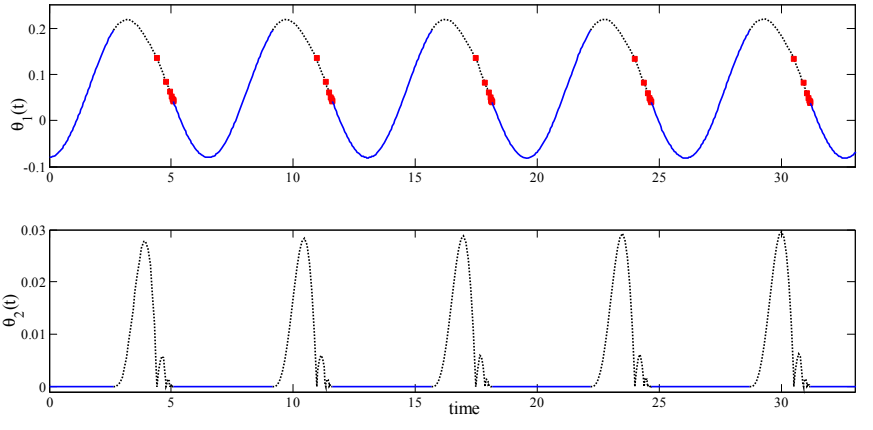


Fig. 4. Time plots of $\theta_1(t)$ and $\theta_2(t)$ for the controlled double pendulum with $e = 0.5$

In order to obtain a non-decaying periodic solution with a *single* plastic collision per cycle, i.e., a simple periodic orbit, we add a PD control law for the torque u_1 as $u_1(\theta_1, \dot{\theta}_1) = -k_1(\theta_1 - \theta_{1e}) - c_1\dot{\theta}_1$. The control parameters are chosen as $k_1 = 0.5$, $\theta_{1e} = \pi/9$ and $c_1 = -0.01$. The proportional term associated with k_1 was chosen as to increase the positive acceleration $\ddot{\theta}_1$ and decrease the negative acceleration $\ddot{\theta}_2$ for $\theta_1 < 0$, and thus increase the constraint force λ that ensures that after the first collision, the lower link does not detach. The negative dissipation term associated with c_1 injects a small amount of energy to the system, that compensates for the losses due to collisions. In simulation under the control law with the same initial condition as above, we obtained convergence to a simple periodic orbit with a single plastic collision per cycle. Figures 3(a) and 3(b) show the phase portraits of the periodic orbit in $(\theta_1, \dot{\theta}_1)$- and $(\theta_2, \dot{\theta}_2)$- planes, respectively. (Time plots of θ_1 and θ_2 appear in [19]).

Next, we apply Theorem 1 to check for existence of a Zeno periodic orbit with $e > 0$. One can verify numerically that the assumptions of the theorem

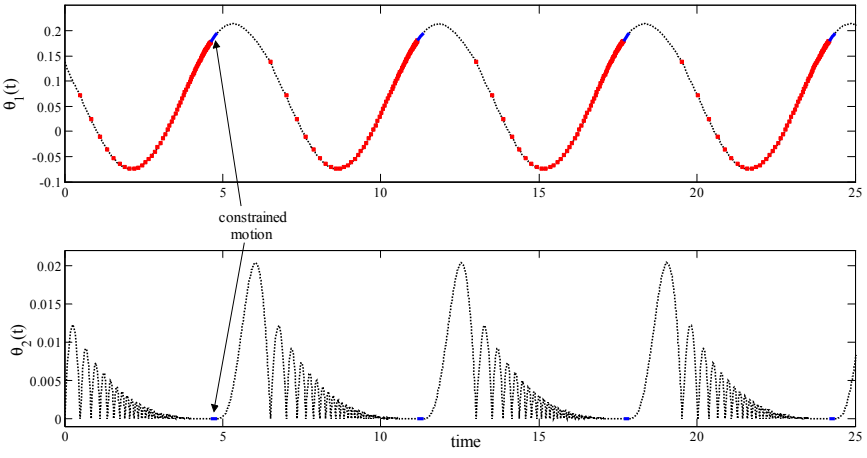


Fig. 5. Time plots of $\theta_1(t)$ and $\theta_2(t)$ for the controlled double pendulum with $e = 0.9225$

are satisfied and that, in particular, the simple periodic orbit obtained through the control law is locally exponentially stable with $\gamma = 0.9404$. Choosing $\epsilon_1 = 0.017$, $\epsilon_2 = 0.06$ and $\epsilon_3 = 0.005$, Theorem 1 implies that the existence of a Zeno periodic orbit with initial condition within $\Omega_2(\epsilon_1, \epsilon_2)$ is guaranteed for any $e \leq 0.0015$. Simulation of the double-pendulum system with $e = 0.0015$ verifies the existence of a Zeno periodic orbit. The simulation results were not shown, since they are not visually distinguishable from the results with $e = 0$.

In order to illustrate the strong conservatism of Theorem 1, we conducted another simulation under the same initial condition, with a coefficient of restitution $e = 0.5$. The infinite Zeno executions were truncated after a finite number of collisions at which the collision velocity $\dot{h}$ is below a threshold of 10^{-10}. The simulation results, which are shown in the time plots of Figure 4, clearly indicate the existence of a Zeno periodic orbit, which was verified numerically to be also locally stable. Figures 3(a) and 3(b) show the phase portraits of the periodic orbits in $(\theta_1, \dot{\theta}_1)$- and $(\theta_2, \dot{\theta}_2)$- planes, respectively, for coefficients of restitution $e = 0$ (plastic impacts) and $e = 0.5$. The thick (blue) curves correspond to the case $e = 0$, and the thin (black) curves correspond to the case $e = 0.5$. The parts of unconstrained motion appear as dashed curves, and the parts of constrained motion appear as solid curves. Note that in Figure 3(b), the constrained motion collapses to the single point $(\theta_2, \dot{\theta}_2) = 0$. From the figures, one can clearly see how the simple periodic orbit is perturbed under non-plastic impacts.

Finally, we gradually increased the coefficient of restitution e and numerically checked for existence of Zeno periodic orbits. The largest value of e for which we obtained such an orbit was $e = 0.9225$. For this value of e, the duration of the constrained motion in the Zeno periodic orbit is very short, as shown in the simulation results of Figure 5. For larger values of e, *the phase of constrained motion vanishes*, and the execution is no longer Zeno. This transition can be viewed as a new type of bifurcation in Lagrangian hybrid systems, in which a

Zeno periodic orbit ceases to be Zeno. To our knowledge, this type of bifurcation was never studied before in the recently emerging literature on bifurcations in non-smooth mechanical systems, (cf. [3,12]).

6 Conclusion

This paper considered two types of periodic orbits in completed Lagrangian hybrid systems: *simple* and *Zeno*. The main result presented is sufficient conditions on when a simple periodic orbit in a Lagrangian hybrid system implies the existence of a Zeno periodic orbit in the same Lagrangian hybrid system with a different coefficient of restitution. Moreover, these conditions give an explicit upper bound the change in the coefficient of restitution that guarantees existence of the Zeno periodic orbit.

The results indicate two major future research directions: better bounds on the allowable change in the coefficient of restitution and conditions on the preservation of stability. For the first direction, as was illustrated by the example, the obtained bounds are strongly conservative; computing tighter bounds in a rigorous fashion will be practically useful and theoretically satisfying. The second future research direction—studying stability—is even more interesting. The authors have been able to show that under certain simplifying assumptions, stability of the simple periodic orbit directly implies the stability of the Zeno periodic orbit. However, this preliminary result was not included in the paper due to space constraints. In the future, understanding how stability extends from one type of orbit to the other with the fewest possible assumptions will provide new and interesting challenges. Finally, extending the results to Lagrangian hybrid system *with multiple constraints* will enable the analysis of full models of bipeds with knees for designing stable walking and running under non-plastic impacts.

References

1. Ames, A.D., Sastry, S.: Hybrid Routhian reduction of Lagrangian hybrid systems. In: Proc. American Control Conference, pp. 2640–2645 (2006)
2. Ames, A.D., Zheng, H., Gregg, R.D., Sastry, S.: Is there life after Zeno? Taking executions past the breaking (Zeno) point. In: Proc. American Control Conference, pp. 2652–2657 (2006)
3. Di Bernardo, M., Garofalo, F., Iannelli, L., Vasca, F.: Bifurcations in piecewise-smooth feedback systems. International Journal of Control 75, 1243–1259 (2002)
4. Bourgeot, J.M., Brogliato, B.: Tracking control of complementarity Lagrangian systems. International Journal of Bifurcation and Chaos 15(6), 1839–1866 (2005)
5. Brogliato, B.: Nonsmooth Mechanics. Springer, Heidelberg (1999)
6. Camlibel, M.K., Schumacher, J.M.: On the Zeno behavior of linear complementarity systems. In: Proc. IEEE Conf. on Decision and Control, pp. 346–351 (2001)
7. Chevallereau, C., Westervelt, E.R., Grizzle, J.W.: Asymptotically stable running for a five-link, four-actuator, planar bipedal robot. International Journal of Robotics Research 24(6), 431–464 (2005)

8. Collins, S.H., Wisse, M., Ruina, A.: A 3-D passive dynamic walking robot with two legs and knees. International Journal of Robotics Research 20, 607–615 (2001)
9. Grizzle, J.W., Abba, G., Plestan, F.: Asymptotically stable walking for biped robots: Analysis via systems with impulse effects. IEEE Trans. on Automatic Control 46(1), 51–64 (2001)
10. Heymann, M., Lin, F., Meyer, G., Resmerita, S.: Analysis of Zeno behaviors in a class of hybrid systems. IEEE Trans. on Automatic Control 50(3), 376–384 (2005)
11. Hirsch, M.W.: Differential Topology. Springer, Heidelberg (1980)
12. Leine, R.I., Nijmeijer, H.: Dynamics and Bifurcations of Non-smooth Mechanical Systems. Springer, Heidelberg (2004)
13. McGeer, T.: Passive walking with knees. In: Proc. IEEE Int. Conf. on Robotics and Automation, vol. 3, pp. 1640–1645 (1990)
14. Miller, B.M., Bentsman, J.: Generalized solutions in systems with active unilateral constraints. Nonlinear Analysis: Hybrid Systems 1, 510–526 (2007)
15. Morris, B., Grizzle, J.W.: A restricted Poincaré map for determining exponentially stable periodic orbits in systems with impulse effects: Application to bipedal robots. In: Proc. IEEE Conf. on Decision and Control and European Control Conf., pp. 4199–4206 (2005)
16. Murray, R.M., Li, Z., Sastry, S.: A Mathematical Introduction to Robotic Manipulation. CRC Press, Boca Raton (1993)
17. Or, Y., Ames, A.D.: Stability of Zeno equlibria in Lagrangian hybrid systems. In: Proc. IEEE Conf. on Decision and Control, pp. 2770–2775 (2008)
18. Or, Y., Ames, A.D.: A formal approach to completing Lagrangian hybrid system models. Submitted to ACC 2009 (2009), www.cds.caltech.edu/~izi/publications.htm
19. Or, Y., Ames, A.D.: Existence of periodic orbits with Zeno behavior in completed Lagrangian hybrid systems. Technical Report (2009), www.cds.caltech.edu/~izi/publications.htm
20. Pfeiffer, F., Glocker, C.: Multibody Dynamics with Unilateral Contacts. John Wiley and Sons, New York (1996)
21. Pogromsky, A.Y., Heemels, W.P.M.H., Nijmeijer, H.: On solution concepts and well-posedness of linear relay systems. Automatica 39(12), 2139–2147 (2003)
22. Pratt, J., Pratt, G.A.: Exploiting natural dynamics in the control of a planar bipedal walking robot. In: Proc. 36th Annual Allerton Conf. on Communications, Control and Computing, pp. 739–748 (1998)
23. Shen, J., Pang, J.-S.: Linear complementarity systems: Zeno states. SIAM Journal on Control and Optimization 44(3), 1040–1066 (2005)
24. Stronge, W.J.: Impact Mechanics. Cambridge University Press, Cambridge (2004)
25. van der Schaft, A., Schumacher, H.: An Introduction to Hybrid Dynamical Systems. Lecture Notes in Control and Information Sci. Springer, Heidelberg (2000)
26. van der Schaft, A.J., Schumacher, J.M.: The complementary-slackness class of hybrid systems. Math. of Control, Signals, and Systems 9(3), 266–301 (1996)
27. Zhang, J., Johansson, K.H., Lygeros, J., Sastry, S.: Zeno hybrid systems. Int. J. Robust and Nonlinear Control 11(2), 435–451 (2001)

Computation of Discrete Abstractions of Arbitrary Memory Span for Nonlinear Sampled Systems

Gunther Reißig

Technische Universität Berlin, Fakultät Elektrotechnik und Informatik, Heinrich-Hertz-Lehrstuhl für Mobilkommunikation HFT 6, Einsteinufer 25, D-10587 Berlin, Germany
http://www.reiszig.de/gunther/

Abstract. In this paper, we present a new method for computing discrete abstractions of arbitrary memory span for nonlinear sampled systems with quantized output. In our method, abstractions are represented by collections of conservative approximations of reachable sets by polyhedra, which in turn are represented by collections of half-spaces. Important features of our approach are that half-spaces are shared among polyhedra, and that the determination of each half-space requires the solution of a single initial value problem in an ordinary differential equation over a single sampling interval only. Apart from these numerical integrations, the only nontrivial operation to be performed repeatedly is to decide whether a given polyhedron is empty. In particular, in contrast to previous approaches, there are no intermediate bloating steps, and convex hulls are never computed. Our method heavily relies on convexity of reachable sets and applies to any sufficiently smooth system if either the sampling period, or the system of level sets of the quantizer can be chosen freely. In particular, it is not required that the system to be abstracted have any stability properties.

1 Introduction

A well-known method for the solution of analysis and synthesis problems for continuous, discrete and hybrid systems consists in first computing a *discrete abstraction* of the system's behavior in the sense of Willems [1,2], and then solving a corresponding (auxiliary) problem for the abstraction, e.g. [3,4,5,6,7,8]. Here, the term *abstraction* refers to a conservative approximation, i.e., a superset, of the system's behavior, which is called *discrete* if it can be realized by a finite (in general non-deterministic) automaton. Auxiliary problems arising in this way are solvable controller synthesis problems if the original problem is and the abstraction is sufficiently accurate. Solvability may be verified, and solutions may be obtained using well-known algorithms from discrete mathematics [9,10,11,8]. In addition, under mild assumptions, it follows from the conservativeness of the approximation that any solution of the auxiliary problem will also be a solution to the problem for the original system, e.g. [5,6,7,8]

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 306–320, 2009.

One of the most complex steps in the above approach is the computation of sufficiently accurate discrete abstractions, which is equivalent to conservative approximation of a large number of reachable sets [6]. Known methods are restricted to rather limited classes of systems or to abstractions of memory span 1, lead to overly conservative abstractions, suffer from their prohibitive computational complexity, or require the solution of non-convex optimization or optimal control problems, see [12,13,14,15,16,17,18,19,20,21] and the references given there. In this paper, we present a new method for computing discrete abstractions of arbitrary memory span for nonlinear sampled systems with quantized output.

In our method, abstractions are represented by collections of conservative approximations of reachable sets by polyhedra, which in turn are represented by collections of half-spaces: We start from a collection of conservative polyhedral approximations of the level sets of the quantizer, which represents a discrete abstraction of memory span 0, and then iteratively determine conservative polyhedral approximations of the reachable sets that define abstractions of greater memory span. Important features of our approach are that half-spaces are shared among polyhedra, and that the determination of each half-space requires the solution of a single initial value problem in an ordinary differential equation over a single sampling interval only. Apart from these numerical integrations, the only nontrivial operation to be performed repeatedly is to decide whether a given polyhedron is empty. In particular, in contrast to previous approaches, there are no intermediate bloating steps, and convex hulls are never computed. Our method heavily relies on convexity of reachable sets and applies to any sufficiently smooth system if either the sampling period, or the system of level sets of the quantizer can be chosen freely. In particular, it is not required that the system to be abstracted have any stability properties.

The remaining of this paper is structured as follows: In section 2 we define the class of sampled systems with quantized output, for which we shall develop an efficient algorithm for computing discrete abstractions. In section 3 we characterize the smallest of those abstractions in terms of reachable sets. In section 4, we present an efficient algorithm for computing discrete abstractions for the class of systems introduced in section 2, under the assumption that all relevant reachable sets are convex. We also discuss two recent results from [22,23] from which convexity of reachable sets can be deduced under mild conditions. Finally, we apply our method to the problem of swinging up the mathematical pendulum in section 5. Proofs of our results will be published with an extended (journal) version of this manuscript.

2 Sampled Systems with Quantized Output

Let the control system

$$\dot{x} = f(x, u(t)) \tag{1}$$

with $f\colon \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$, a sampling period $T > 0$, and a finite set $U \subseteq (\mathbb{R}^m)^{[0,T]}$ of admissible controls on sampling intervals be given. Hence, elements of U are

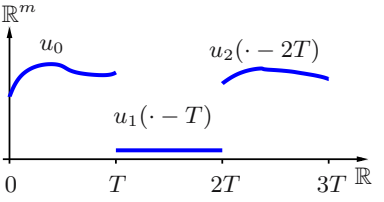


Fig. 1. An admissible control signal for (1)

signals $[0, T] \to \mathbb{R}^m$, and we identify each sequence $(u_k)_{k\in\mathbb{Z}_+}$ of such signals with a control signal defined on $\mathbb{R}_+$,

$$(u_0, u_1, \dots)(t) := u_{\lfloor t/T \rfloor}(t - \lfloor t/T \rfloor) \quad \text{for all } t \geq 0,$$

see Fig. 1. Here, $\lfloor x \rfloor$ denotes the largest integer not greater than x, $\mathbb{R}_+$ and $\mathbb{Z}_+$, the set of nonnegative reals and integers, respectively, and A^B, the set of maps $B \to A$.

The set $\mathcal{U}$ of controls $u\colon \mathbb{R}_+ \to \mathbb{R}^m$ admissible for (1) is now defined as the set of all sequences in U,

$$\mathcal{U} = \left\{ u\colon \mathbb{R}_+ \to \mathbb{R}^m \,\middle|\, \forall_{k\in\mathbb{Z}_+}\ \exists_{u_k\in U}\ \forall_{t\in[kT,(k+1)T[}\ u(t) = u_k(t - kT) \right\}. \tag{2}$$

Here, $[a, b]$, $]a, b[$, and $[a, b[$, $]a, b]$ denote closed, open and half-open intervals, respectively.

We assume throughout this paper that for any admissible control $u \in \mathcal{U}$, initial value problems composed of (1) and an initial condition

$$x(0) = x_0 \tag{3}$$

are uniquely solvable for any $x_0 \in \mathbb{R}^n$, with all solutions extendable to the entire time axis $\mathbb{R}_+$.

As an extension of the well-known concepts of flow and general solution for ordinary differential equations [24,25], we define the *general solution* φ of (1) by

$$\varphi(t, x_0, u) := \text{value of the solution of initial value problem (1), (3) at time } t,$$

where $x_0 \in \mathbb{R}^n$ and $u \in \mathcal{U}$. Note that it is not necessary to specify all the components u_k of $u = (u_0, u_1, \dots)$, and that the values of u at sampling instants are irrelevant, so we may write

$$\varphi(t, x_0, u_0, \dots, u_k) := \varphi(t, x_0, u) \quad \text{if } t \leq (k+1)T,\ u = (u_0, \dots, u_k, \dots).$$

We now consider the sampled version

$$x(k+1) = \varphi(T, x(k), u_k), \quad k \in \mathbb{Z}_+ \tag{4}$$

of (1), where φ is the general solution of (1). By our assumptions on (1), the right hand side of the difference equation (4) is defined for all $(x(k), u_k) \in \mathbb{R}^n \times U$.

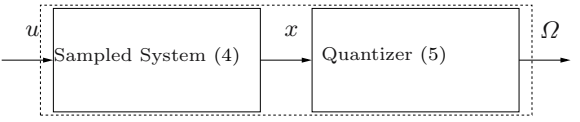


Fig. 2. Sampled system with quantized output

We define the quantizer by specifying its level sets: Let $C \subseteq \mathcal{P}(\mathbb{R}^n)$ be a finite covering of the state space $\mathbb{R}^n$ of (4) that does not contain the empty set, where $\mathcal{P}(M)$ denotes the power set of M. The quantizer Q is defined as the map

$$Q\colon \mathbb{R}^n \to \mathcal{P}(C)\colon x \mapsto \{\, \Omega \in C \,|\, x \in \Omega \,\}. \tag{5}$$

Note that $Q(x) \neq \emptyset$ for any $x \in \mathbb{R}^n$ as C is a covering of $\mathbb{R}^n$, and that, in general, the quantizer is non-deterministic. See [13] for the equivalent concept of a "measurement map".

The system composed of the sampled system (4) and the quantizer (5) shown in Fig. 2 may be described by the following difference equation with set valued output:

$$x(k+1) = \varphi(T, x(k), u_k), \quad k \in \mathbb{Z}_+, \tag{6a}$$
$$\Omega_k \in Q(x(k)). \tag{6b}$$

The *(external) behavior* [1,2] $B_{(6)}$ of the system (6) is the set of all (external) signals $(u, \Omega) \in U^{\mathbb{Z}_+} \times C^{\mathbb{Z}_+}$ that are compatible with (6), i.e.,

$$B_{(6)} = \left\{\, (u,\Omega) \,\middle|\, \exists_{x\colon \mathbb{Z}_+ \to \mathbb{R}^n}\ \forall_{k\in\mathbb{Z}_+}\ \ (x(k) \in \Omega_k \text{ and } x(k+1) = \varphi(T, x(k), u_k)) \,\right\}. \tag{7}$$

3 Smallest Discrete Abstractions and Reachable Sets

In this section we characterize the smallest N-complete discrete abstraction of the behavior $B_{(6)}$ given by (7) in terms of reachable sets of the time-continuous control system (1). We begin with some terminology from behavioral theory [2]:

Let a set $I \subseteq \mathbb{Z}_+$, an arbitrary set X and a behavior $B \subseteq X^{\mathbb{Z}_+}$ be given. The *restriction of B to I*, $B|_I$, is defined by $B|_I := \{\, x|_I \,|\, x \in B \,\}$, where $x|_I$ denotes the restriction of the map x to I.

B is called *time-invariant* if $\sigma B \subseteq B$, where σ denotes the shift operator defined by $\sigma := \sigma^1$ and $(\sigma^k x)(t) := x(t+k)$ for all $x\colon \mathbb{Z}_+ \to X$ and all $k, t \in \mathbb{Z}_+$.

If B is time-invariant, it is called *complete* if

$$B = \left\{\, x \in X^{\mathbb{Z}_+} \,\middle|\, \forall_{k_1,k_2\in\mathbb{Z}_+, k_1\le k_2}\ x|_{[k_1,k_2]} \in B|_{[k_1,k_2]} \,\right\},$$

and it is called *complete with memory span N* (or *N-complete*, for short) for some $N \in \mathbb{Z}_+$ if

$$B = \left\{\, x \in X^{\mathbb{Z}_+} \,\middle|\, \forall_{k\in\mathbb{Z}_+}\ (\sigma^k x)|_{[0,N]} \in B|_{[0,N]} \,\right\}.$$

If B is time-invariant, we call the set

$$\bigcap_{\substack{B \subseteq B' \subseteq X^{\mathbb{Z}_+}, \\ B' \text{ is } N\text{-complete}}} B' \tag{8}$$

the *N-complete hull* of B. (The map that assigns to B its N-complete hull (8) is a closure operator [26], which is why we call (8) a *hull*; N-complete hulls are called "strongest N-complete approximations" in [6].)

The behavior $B_{(6)}$ of the sampled system with quantized output is time-invariant, but in general not complete, which is why we are looking for a discrete abstraction of it. For any $N \in \mathbb{Z}_+$, the N-complete hull of $B_{(6)}$ is the smallest abstraction of the kind we seek to obtain. Unfortunately, that abstraction may be computed exactly for special classes of systems (1) and quantizers (5) only. Nevertheless, the following characterizations of that smallest abstraction will be useful in the next section when we derive an algorithm for effectively computing another abstraction that conservatively approximates the smallest one.

Theorem 1. *Let $N \in \mathbb{Z}_+$ and B_N be the N-complete hull of the behavior $B_{(6)}$ given by (7), φ the general solution of (1), and $(u, \Omega) \in U^{\mathbb{Z}_+} \times C^{\mathbb{Z}_+}$. Then the following are equivalent:*

(i) $(u, \Omega) \in B_N$.

(ii) *For all $\tau \in \mathbb{Z}_+$ there exists $x_0 \in \mathbb{R}^n$ such that $\varphi(kT, x_0, u_\tau, \dots, u_{\tau+k-1}) \in \Omega_{\tau+k}$ holds for all $k \in \{0, \dots, N\}$.*

(iii) *For all $\tau \in \mathbb{Z}_+$ the following holds:*

$$\Omega_{\tau+N} \cap \bigcap_{k=1}^{N} \varphi(kT, \Omega_{\tau+N-k}, u_{\tau+N-k}, \dots, u_{\tau+N-1}) \neq \emptyset. \tag{9}$$

(iv) *$M_N^\tau \neq \emptyset$ for all $\tau \in \mathbb{Z}_+$, where M_N^τ is defined by*

$$\begin{aligned} M_0^\tau &= \Omega_\tau, \\ M_k^\tau &= \Omega_{\tau+k} \cap \varphi(T, M_{k-1}^\tau, u_{\tau+k-1}) \quad (k \in \{1, \dots, N\}). \end{aligned}$$

Characterization (iv) has been given in [6], and a characterization similar to (iii) has been proposed in [23].

A set of the form $\varphi(t, \Omega, u)$ arising in Theorem 1 is called *reachable set from Ω at time t under control u*.

4 Computation of Discrete Abstractions and Convexity of Reachable Sets

We have seen in Section 3 that computing the smallest discrete abstraction of a particular memory span for the behavior $B_{(6)}$ given by (7) requires the solution of numerous difficult reachability problems that may be solved exactly for special

classes of systems (1) and quantizers (5) only. In the present section, we aim at computing another discrete abstraction which approximates the smallest one. Our starting point is the following basic idea:

If $\Omega_{\tau+N}$ and all the reachable sets on the left hand side of condition (9) in Theorem 1 were convex, these sets could be substituted with approximations by means of supporting half-spaces. The resulting approximate condition would then characterize a superset of the N-complete hull of $B_{(6)}$ that is N-complete, and hence, a discrete abstraction of memory span N of $B_{(6)}$.

In the remaining of this section, we shall derive an algorithm for the computation of discrete abstractions of arbitrary memory span N for the behavior $B_{(6)}$ given by (7) which approximates the N-complete hull of $B_{(6)}$. To this end, we start with the question of how to obtain conservative polyhedral approximations of reachable sets by means of supporting half-spaces.

Definition 1. *Let $\Omega \subseteq \mathbb{R}^n$ be convex and $p \in \Omega$. A vector $v \in \mathbb{R}^n$ is normal to Ω at p [27] if $\langle v | x - p \rangle \leq 0$ for all $x \in \Omega$, where $\langle \cdot | \cdot \rangle$ denotes the standard Euclidean inner product.*

Proposition 1. *Let the right hand side f of (1) be of class C^1 w.r.t. its first argument and continuous, and let φ denote the general solution of (1). Let further $u \in \mathcal{U}$ be a piecewise continuous control admissible for (1), $\Omega \subseteq \mathbb{R}^n$ be convex, $p \in \Omega$, $v \in \mathbb{R}^n$, and $\tau \in \mathbb{R}_+$. Finally, let v' be the value at time τ of the solution of the following initial value problem:*

$$\begin{aligned} \dot{x} &= -D_1 f(\varphi(t, p, u), u(t))^* x, \\ x(0) &= v, \end{aligned}$$

where $(\cdot)^*$ denotes the transpose, and D_1, the partial derivative w.r.t. the first argument.

If the reachable set $\varphi(\tau, \Omega, u)$ is convex, then v is normal to Ω at p if and only if v' is normal to $\varphi(\tau, \Omega, u)$ at $\varphi(\tau, p, u)$.

The above result tells us that conservative polyhedral approximations of all the reachable sets on the left hand side of condition (9) may be obtained from analogous approximations of the level sets of the quantizer (5) by solving initial value problems in the $2n$-dimensional ordinary differential equation

$$\dot{x} = f(x, u(t)), \tag{10a}$$

$$\dot{y} = -D_1 f(x, u(t))^* y. \tag{10b}$$

For further reference, we define a map φ^* that realizes the determination of a supporting half-space of the reachable set $\varphi((k+1)T, \Omega, u)$ from a supporting half-space of Ω ($k \in \mathbb{Z}_+$):

$$\varphi^* \colon \mathbb{R}^n \times \mathbb{R}^n \times \mathcal{U} \to \mathbb{R}^n \times \mathbb{R}^n \colon (p, v, u_0, \ldots, u_k) \mapsto \psi((k+1)T, (p, v), u_0, \ldots, u_k), \tag{11}$$

where ψ is the general solution of (10) and T, the sampling period.

We now formalize the substitution of reachable sets on the left hand side of condition (9) in Theorem 1 with approximations by means of supporting halfspaces:

Definition 2. *Let P be the map defined by*

$$P\colon \mathbb{R}^n \times \mathbb{R}^n \to \mathcal{P}(\mathbb{R}^n)\colon (p,v) \mapsto \{\, x \in \mathbb{R}^n \mid \langle v | x - p\rangle \leq 0 \,\}$$

and set

$$P(\Sigma) = \bigcap_{(p,v)\in\Sigma} P(p,v)$$

for $\Sigma \subseteq \mathbb{R}^n \times \mathbb{R}^n$.
Let $\Sigma \subseteq \mathbb{R}^n \times \mathbb{R}^n$ and $\Omega \subseteq \mathbb{R}^n$.
We call Σ a conservative polyhedral approximation of Ω if $\Omega \subseteq P(\Sigma)$, and a supporting polyhedral approximation of Ω, if $p \in \Omega$ and v is normal to Ω at p, for all $(p,v) \in \Sigma$.

If Σ is a conservative polyhedral approximation of Ω, $(p,v) \in \Sigma$ is called redundant in Σ if $P(\Sigma) = P(\Sigma \setminus \{(p,v)\})$.

Let $N \in \mathbb{Z}_+$, $\Omega_0, \ldots, \Omega_N \subseteq \mathbb{R}^n$, and $u_0, \ldots, u_{N-1} \in U$ and define

$$M(\Omega_0, \ldots, \Omega_N, u_0, \ldots, u_{N-1}) := \Omega_N \cap \bigcap_{k=1}^{N} \varphi(kT, \Omega_{N-k}, u_{N-k}, \ldots, u_{N-1}).$$

Hence, $M(\Omega_\tau, \ldots, \Omega_{\tau+N}, u_\tau, \ldots, u_{\tau+N-1})$ is just the set on the left hand side of condition (9). For any convex $\Omega \subseteq \mathbb{R}^n$, let $\Sigma(\Omega)$ be a supporting polyhedral approximation of Ω and define

$$\widehat{M}(\Omega_0, \ldots, \Omega_N, u_0, \ldots, u_{N-1}) := P(\Sigma(\Omega_N)) \cap \bigcap_{k=1}^{N} P\left(\varphi^*(\Sigma(\Omega_{N-k}), u_{N-k}, \ldots, u_{N-1})\right).$$

Hence, $\widehat{M}(\Omega_\tau, \ldots, \Omega_{\tau+N}, u_\tau, \ldots, u_{\tau+N-1})$ is the left hand side of condition (9) with all the reachable sets substituted with supporting polyhedral approximations obtained from application of the map φ^* to supporting polyhedral approximations of level sets of the quantizer.

We now define

$$S(\Omega_0) := \Sigma(\Omega_0), \tag{12}$$

$$S(\Omega_0, \ldots, \Omega_k, u_0, \ldots, u_{k-1}) := \Sigma(\Omega_k) \cup \bigcup_{q=1}^{k} \varphi^*(\Sigma(\Omega_{k-q}), u_{k-q}, \ldots, u_{k-1}) \tag{13}$$

for all $k \in \{1, \ldots, N\}$ to obtain

$$\widehat{M}(\Omega_0, \ldots, \Omega_N, u_0, \ldots, u_{N-1}) = P(S(\Omega_0, \ldots, \Omega_N, u_0, \ldots, u_{N-1})).$$

Hence, $S(\Omega_0, \ldots, \Omega_N, u_0, \ldots, u_{N-1})$ is a conservative polyhedral approximation of $\widehat{M}(\Omega_0, \ldots, \Omega_N, u_0, \ldots, u_{N-1})$, though not necessarily a supporting one.

The following result shows that the sets $S(\dots)$ just defined enjoy a recursive description analogous to the one for the sets M_k^τ. (See condition (iv) in Theorem 1.)

Theorem 2. *Let $N \in \mathbb{Z}_+$, $\Omega_0, \dots, \Omega_N \subseteq \mathbb{R}^n$ be convex, and $u_0, \dots, u_{N-1} \in U$. Let φ be the general solution of (1) and assume that the reachable sets $\varphi(kT, \Omega_{N-k}, u_{N-k}, \dots, u_{N-1})$ are convex for all $k \in \{1, \dots, N\}$. For each $k \in \{0, \dots, N\}$, let $\Sigma(\Omega_k)$ be a supporting polyhedral approximation of Ω_k, and let $S(\Omega_0)$, $\dots$, $S(\Omega_0, \dots, \Omega_N, u_0, \dots, u_{N-1})$ be defined by (12)–(13). Then*

$$S(\Omega_0, \dots, \Omega_k, u_0, \dots, u_{k-1}) = \Sigma(\Omega_k) \cup \varphi^*(S(\Omega_0, \dots, \Omega_{k-1}, u_0, \dots, u_{k-2}), u_{k-1})$$

for all $k \in \{1, \dots, N\}$.
Furthermore, if (p, v) is redundant in $S(\Omega_0, \dots, \Omega_{k-1}, u_0, \dots, u_{k-2})$, then it is so in $S(\Omega_0, \dots, \Omega_k, u_0, \dots, u_{k-1})$.

It follows from the above result that half-spaces are shared among many of the conservative polyhedral approximations $\widetilde{M}(\dots)$ of reachable sets and that the computational cost per half-space is just a single solution of an initial value problem in the $2n$-dimensional ordinary differential equation (10) over a single sampling interval.

We now propose an algorithm that, under the assumption that reachable sets are convex, determines discrete abstractions of the behavior $B_{(6)}$ given by (7).

Input:

(i) $N \in \mathbb{Z}_+$ (memory span of abstraction to be computed);
(ii) T, $\mathcal{U}$, U (see section 2);
(iii) C: finite covering of $\mathbb{R}^n$ by convex polyhedra (level sets of the quantizer (5));
(iv) $C' := \{\, \Omega \in C \mid \Omega \text{ bounded} \,\}$;
(v) a set $\widehat{\Omega}$ for each $\Omega \in C'$ with $\Omega \subseteq \widehat{\Omega}$ and reachable sets $\varphi(kT, \widehat{\Omega}, u)$ convex for all $k \in \{0, \dots, N\}$, $\Omega \in C'$ and $u \in \mathcal{U}$;
(vi) a supporting polyhedral approximation $\Sigma(\widehat{\Omega})$ of $\widehat{\Omega}$ for all $\Omega \in C'$.

for all $\Omega \in C'$ **do**
  $\widetilde{S}(\Omega) = \Sigma(\widehat{\Omega})$
end for
for all $\Omega \in C \setminus C'$ **do**
  $\widetilde{S}(\Omega) = \emptyset$
end for
for $k = 1, \dots, N$ **do**
  for all $(\Omega_0, \dots, \Omega_k, u_0, \dots, u_{k-1}) \in C^{k+1} \times U^k$ **do**
    if $\widetilde{S}(\Omega_0, \dots, \Omega_{k-1}, u_0, \dots, u_{k-2}) = \emptyset$ **then**
      $Z := \emptyset$
    else if $\Omega_k \cap P(\varphi^*(\widetilde{S}(\Omega_0, \dots, \Omega_{k-1}, u_0, \dots, u_{k-2}), u_{k-1})) = \emptyset$ **then**
      $Z := \mathbb{R}^n \times \mathbb{R}^n$
    else if $\Omega_k \notin C'$ **then**
      $Z := \emptyset$

15: **else**
16: $Z := \Sigma(\widehat{\Omega}_k) \cup \varphi^*(\widetilde{S}(\Omega_0, \ldots, \Omega_{k-1}, u_0, \ldots, u_{k-2}), u_{k-1})$
17: **end if**
18: $\widetilde{S}(\Omega_0, \ldots, \Omega_k, u_0, \ldots, u_{k-1}) := Z$
19: **end for**
20: **end for**

Output: $\widetilde{S}(\ldots)$.

The following result shows that the above algorithm determines a discrete abstraction of the behavior $B_{(6)}$ of the sampled and quantized system (6).

Theorem 3. *Denote by $\widetilde{S}(\ldots)$ the sets determined by the above algorithm. Under the assumptions made in the list of inputs, the set*

$$\left\{ (u, \Omega) \in U^{\mathbb{Z}_+} \times C^{\mathbb{Z}_+} \,\middle|\, \forall_{\tau \in \mathbb{Z}_+} \; P(\widetilde{S}(\Omega_\tau, \ldots, \Omega_{\tau+N}, u_\tau, \ldots, u_{\tau+N-1})) \neq \emptyset \right\}$$

is an N-complete conservative approximation of the behavior $B_{(6)}$ given by (7).

Some remarks are in order. First, note that the algorithm contains just two nontrivial operations which need to be performed repeatedly, namely, the determination of the set

$$\varphi^*(\widetilde{S}(\Omega_0, \ldots, \Omega_{k-1}, u_0, \ldots, u_{k-2}), u_{k-1}), \tag{14}$$

which appears at lines 11 and 16, and the test for emptiness at line 11. Regarding the former operation, it follows from the definition (11) of the map φ^* that for each $s \in \widetilde{S}(\Omega_0, \ldots, \Omega_{k-1}, u_0, \ldots, u_{k-2})$, determination of (14) requires the solution of an initial value problem in the $2n$-dimensional ordinary differential equation (10) over a single sampling interval. Hence, the set (14) may be determined from at most $|\widetilde{S}(\Omega_0, \ldots, \Omega_{k-1}, u_0, \ldots, u_{k-2})|$ such solutions, where $|\cdot|$ denotes cardinality. As $P(\varphi^*(\widetilde{S}(\Omega_0, \ldots, \Omega_{k-1}, u_0, \ldots, u_{k-2}), u_{k-1}))$ is a convex polyhedron, the test for emptiness at line 11 may also be effectively performed since Ω_k is also a convex polyhedron by hypothesis (iii) in the list of inputs of the algorithm.

Second, it should be obvious that the sets $\emptyset$ and $\mathbb{R}^n \times \mathbb{R}^n$ play a role similar to zeros in sparse matrices [28,29]. In particular, if the sets $\widetilde{S}(\ldots)$ are stored in a tree, sets $\widetilde{S}(\ldots) = \emptyset$ and $\widetilde{S}(\ldots) = \mathbb{R}^n \times \mathbb{R}^n$ do not need to be stored and computations on them do not need to actually be performed.

Finally, note that all our arguments so far were based on the *assumption* that reachable sets arising in characterizations of N-complete hulls are convex. It follows from recent results of the author [22,23] that convexity of reachable sets can be guaranteed under mild smoothness assumptions on the right hand side f of the continuous control system (1). Let us briefly look at special cases of two such results from [22].

Theorem 4. *Let the right hand side f of (1) be of class $C^{1,1}$ (C^1 with Lipschitz derivative) with respect to its first argument and continuous, and let $u \in \mathcal{U}$ be*

piecewise continuous. Let further $x_0 \in \mathbb{R}^n$, $r > 0$ *and* $t \geq 0$. *Finally, assume that*

$$M_1 \geq 2\mu_+ \left(D_1 f(x, u(\tau))\right) - \mu_- \left(D_1 f(x, u(\tau))\right) \tag{15}$$

holds for all $(\tau, x) \in \mathbb{R}_+ \times \mathbb{R}^n$, *and let* M_2 *be a Lipschitz constant for the map* $(\tau, x) \mapsto D_1 f(x, u(\tau))$ *w.r.t. its second argument on* $\mathbb{R}_+ \times \mathbb{R}^n$. *Then the reachable set* $\varphi(t, \bar{B}(x_0, r), u)$ *is convex if*

$$rM_2 \int_0^t \mathrm{e}^{M_1 \tau} d\tau \leq 1. \tag{16}$$

Here, $\mu_+(M)$ *and* $\mu_-(M)$ *denote the maximum and minimum, respectively, eigenvalues of the symmetric part* $(M + M^*)/2$ *of* M, *and* $\bar{B}(x_0, r)$, *the closed Euclidean ball of radius* r *centered at* x_0 *w.r.t. the Euclidean norm* $\|\cdot\|$.

Theorem 5. *Let* u, f, x_0, r, *and* t *as in Theorem 4 and assume in addition that* f *is of class* C^2 *with respect to its first argument. Then* $\varphi(t, \bar{B}(x_0, r), u)$ *is convex if and only if*

$$\int_0^t \left\langle x - x_0 \middle| D_2\varphi(\tau, x, u)^{-1} D_1^2 f(\varphi(\tau, x, u), u(\tau))(D_2\varphi(\tau, x, u)h)^2 \right\rangle d\tau \leq 1 \tag{17}$$

for all $x \in \partial\bar{B}(x_0, r)$ *and all* $h \perp (x - x_0)$ *with* $\|h\| = 1$. *Here,* ∂X *denotes the boundary of* X, D_1^2, *the second order partial derivative w.r.t. the first argument, and* $D_1^2 f(x, u)h^2 := D_1^2 f(x, u)(h, h)$.

The bounds M_1 and M_2 in Theorem 4 may be directly determined from the right hand side f of the time-continuous system (1) and the set $\mathcal{U}$ of admissible controls, and the bound (16) on the radius is sharp provided $n \geq 2$ [22]. Application of previous results from [30,31,32] would necessarily be based on estimates of $\|D_2\varphi(t, \cdot, u)^{-1}\|$ and $\|D_2^2\varphi(t, \cdot, u)\|$ and, in general, would yield a smaller bound.

Theorem 4 implies that the reachable set $\varphi(t, \bar{B}(x_0, r), u)$ is convex whenever t or r is sufficiently small. Hence, if either the sampling period T, or the system of level sets of the quantizer (5) can be chosen freely, convexity of reachable sets arising in the algorithm proposed in this section, and hence, applicability of the algorithm, can be guaranteed by either choosing T sufficiently small or choosing sufficiently small balls as the elements of C'.

While Theorem 4 gives a sufficient condition for the convexity of a reachable set, Theorem 5 appears to have the form of a criterion. However, condition (17) contains the general solution φ of (1) and therefore, may only rarely be directly verified. Instead, one usually has to resort to estimating the integrand on the left hand side of (17). In the twice continuously differentiable case, use of the estimate obtained from Ważewski's inequality [22] would yield precisely the bound (16) in Theorem 4. The advantage of Theorem 5 is that for specific examples of (1) one is often able to obtain better estimates for the integrand in (17), and hence, larger bounds on the radius than (16). This has been demonstrated in [22]. In view of the algorithm proposed in this section, note that larger bounds directly translate into lower computational complexity.

As Theorems 4 and 5 extend to right hand sides f defined in arbitrary Hilbert spaces, convexity of reachable sets from ellipsoids rather than from Euclidean balls may also be guaranteed [22]. In view of these results, each bounded element $\Omega \in C$ of the system C of level sets of the quantizer (5) should be contained in some ellipsoid $\widehat{\Omega}$ whose reachable sets are guaranteed to be convex by Theorems 5 and 4. See items (iv) and (v) of the list of inputs of the algorithm.

5 Example

Consider the pendulum equations

$$\dot{x}_1 = x_2, \tag{18a}$$
$$\dot{x}_2 = -\sin(x_1) - u\cos(x_1), \tag{18b}$$

which describe frictionless motion of a pendulum mounted on a cart whose acceleration is u. The motion of the cart is not modeled; u is considered an input. We seek to design a controller that steers a sampled version of (18) from some neighborhood of the origin within a finite number of steps into the ellipsoid E defined by

$$E = (\pi, 0) + \left\{ x \in \mathbb{R}^2 \,\middle|\, \langle x|Hx\rangle \le 1 \right\}, \quad H = \frac{1}{10\sqrt{2}} \begin{pmatrix} 13 & 3\sqrt{3} \\ 3\sqrt{3} & 7 \end{pmatrix} \tag{19}$$

and shown in Fig. 3(a), such that the closed loop satisfies the constraints

$$|x_2| \le \pi, \tag{20a}$$
$$u \in \{0, -2, 2\}, \tag{20b}$$

with controls being constant on sampling intervals, i.e.,

$$U = \{[0,T] \to \mathbb{R} \colon t \mapsto 0, [0,T] \to \mathbb{R} \colon t \mapsto -2, [0,T] \to \mathbb{R} \colon t \mapsto 2\}$$

in the notation of section 2.

To this end, let $\mathcal{U}$ be defined by (2), let φ denote the general solution of (18), and consider the sampled system (4) with sampling period $T = 0.35$. Define the quantizer Q of (5) by its system C of level sets ("cells"),

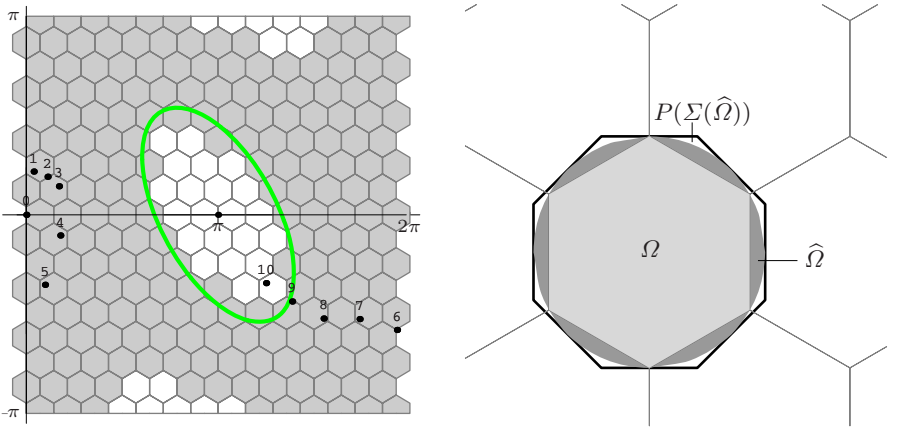
$$C = C' \cup \{\mathbb{R} \times]\pi, \infty[\,, \mathbb{R} \times]-\infty, -\pi[\},$$

where C' is a set of 238 translated and possibly truncated copies of the hexagon

$$\frac{\pi}{14\sqrt{3}} \operatorname{conv}\{(0,-2), (\sqrt{3},-1), (\sqrt{3},1), (0,2), (-\sqrt{3},1), (-\sqrt{3},-1)\}, \tag{21}$$

see Fig. 3(a). Since the right hand side of (18) is periodic in x with period $(2\pi, 0)$, we tacitly consider the system (18) on the cylinder, so that C really is a covering of the state space of (18) and of (4).

For each $\Omega \in C'$, let $\widehat{\Omega}$ be a translated copy of the circumcircle of the hexagon (21), and $\Sigma(\widehat{\Omega})$, a supporting polyhedral approximation of $\widehat{\Omega}$ consisting of 8 equally distributed hyperplanes. See Fig. 3(b). The next result shows that the reachable set $\varphi(t, \widehat{\Omega}, u)$ is convex for all $\Omega \in C'$, all $u \in \mathcal{U}$, and all $t \in \{T, 2T\}$.



(a) (b)

Fig. 3. (a) Covering of the state space by cells which defines the quantizer (5). A discrete controller obtained from a 2-complete abstraction of the behavior of the the sampled and quantized system (6) steers (6) from any shaded cell into some of the (unshaded) cells contained in the ellipsoid (19). The labeled points represent a particular closed-loop solution on the time interval $\{0, 1, \ldots, 10\}$. (b) A hexagon $\Omega \in C'$, its circumcircle $\widehat{\Omega}$, and a supporting polyhedral approximation $\Sigma(\widehat{\Omega})$ of $\widehat{\Omega}$.

Theorem 6. *Let $x_0 \in \mathbb{R}^2$, u piecewise continuous with $|u(\tau)| \leq \hat{u}$ for all $\tau \in \mathbb{R}_+$, $\omega = \left(1 + \hat{u}^2\right)^{1/4}$, $0 < t \leq \frac{\pi}{2\omega}$,*

$$R = \frac{6\omega^2}{(1+\omega^2)^{3/2} \sinh(\omega t) \left(5 + \cosh(2\omega t) - 10 \exp(-\omega)\right)}. \tag{22}$$

Then the reachable set $\varphi(t, \bar{B}(x_0, r), u)$ is convex whenever $0 < r \leq R$.

Indeed, the circumcircle of the hexagon (21) is of radius $\pi/(7\sqrt{3}) < 0.26$, while the bound (22) exceeds 0.26 for $\hat{u} = 2$ and $t \in \{T, 2T\}$, and $T < 2T < 1 < \frac{\pi}{2}\left(1 + \hat{u}^2\right)^{-1/4}$. Hence, for memory span $N \in \{0, 1, 2\}$, all relevant reachable sets are convex, and Theorem 3 guarantees that the algorithm proposed in section 4 yields a discrete abstraction B_N of the behavior of the sampled and quantized system (6).

We have implemented our algorithm from section 4 in *Mathematica 5.2* [33] and computed B_N for $N \in \{0, 1, 2\}$. Tab. 1 gives some statistics. Note that for $N = 2$, the number of half-spaces to be determined is less than the number of conservative polyhedral approximations of reachable sets to be stored, which demonstrates an important feature of our method. It took 0.8, 30.2 and 101.6 seconds to compute B_N for $N = 0, 1, 2$, respectively, on an IBM Thinkpad X60 with 1.83 GHz clock rate.

Based on the abstraction B_2 and using well-known methods from discrete mathematics [11,8], we have obtained a discrete controller which, by construction,

Table 1. Application of the algorithm proposed in section 4 to the present example (N: memory span of computed abstraction; s: number of half-spaces to be determined; q: number of polyhedra tested for emptiness; p: number of conservative polyhedral approximations of reachable sets to be stored)

N	s	q	p
0	1906	0	240
1	5184	30118	3060
2	21424	70496	22840

steers the sampled and quantized system (6) from any cell shaded in Fig. 3 into some cell inside the ellipsoid E within at most 16 steps, and in particular, within 10 steps if starting from the origin. See Fig. 3(a). By construction, solutions of the closed loop remain in C' before entering E, which guarantees control and state constraints (20) are satisfied.

Acknowledgments. The author thanks Lars Grüne (Bayreuth) and Jan Willems (Leuven) for stimulating discussions, on a preliminary version of the contributions made in the present paper in the first case, and on certain properties of behaviors, in the second. The author also thanks Marcus von Lossow (Bayreuth) for the opportunity to use his implementation of shortest path algorithms for hypergraphs [34], and four anonymous referees whose constructive criticism helped to improve this text.

References

1. Willems, J.C.: Models for dynamics. In: Dynam. Report. Ser. Dynam. Systems Appl., vol. 2, pp. 171–269. Wiley, Chichester (1989)
2. Willems, J.C.: Paradigms and puzzles in the theory of dynamical systems. IEEE Trans. Automat. Control 36(3), 259–294 (1991)
3. Blanke, M., Kinnaert, M., Lunze, J., Staroswiecki, M.: Diagnosis and fault-tolerant control. Springer, Berlin (2003)
4. Tomlin, C.J., Mitchell, I., Bayen, A.M., Oishi, M.: Computational techniques for the verification of hybrid systems. Proc. IEEE 91(7), 986–1001 (2003)
5. Koutsoukos, X.D., Antsaklis, P.J., Stiver, J.A., Lemmon, M.D.: Supervisory control of hybrid systems. Proc. IEEE 88(7), 1026–1049 (2000)
6. Moor, T., Raisch, J.: Supervisory control of hybrid systems within a behavioural framework. Systems Control Lett. 38(3), 157–166 (1999)
7. Moor, T., Davoren, J.M., Anderson, B.D.O.: Robust hybrid control from a behavioural perspective. In: Proc. 41th IEEE Conference on Decision and Control, Las Vegas, U.S.A., 2002, pp. 1169–1174. IEEE, New York (2002)
8. Grüne, L., Junge, O.: Approximately optimal nonlinear stabilization with preservation of the Lyapunov function property. In: Proc. 46th IEEE Conference on Decision and Control, New Orleans, Louisiana, U.S.A., 2007, pp. 702–707. IEEE, New York (2007)
9. Ramadge, P.J., Wonham, W.M.: Modular feedback logic for discrete event systems. SIAM J. Control Optim. 25(5), 1202–1218 (1987)

10. Ramadge, P.J.G., Wonham, W.M.: The control of discrete event systems. Proc. IEEE 77(1), 81–98 (1989)
11. Gallo, G., Longo, G., Pallottino, S., Nguyen, S.: Directed hypergraphs and applications. Discrete Appl. Math. 42(2-3), 177–201 (1993)
12. Kurzhanski, A.B., Varaiya, P.: Ellipsoidal techniques for reachability analysis. In: Lynch, N.A., Krogh, B.H. (eds.) HSCC 2000. LNCS, vol. 1790, pp. 202–214. Springer, Heidelberg (2000)
13. Moor, T., Raisch, J.: Abstraction based supervisory controller synthesis for high order monotone continuous systems. In: Engell, S., Frehse, G., Schnieder, E. (eds.) Modelling, Analysis, and Design of Hybrid Systems. Lect. Notes Control Inform. Sciences, vol. 279, pp. 247–265. Springer, Heidelberg (2002)
14. Junge, O.: Rigorous discretization of subdivision techniques. In: International Conference on Differential Equations, Berlin, 1999, vol. 1, 2, pp. 916–918. World Sci. Publ., River Edge (2000)
15. Puri, A., Varaiya, P., Borkar, V.: ϵ-approximation of differential inclusions. In: Proceedings of the 34th IEEE Conference on Decision and Control, New Orleans, LA, U.S.A., December 13-15, 1995, vol. 3, pp. 2892–2897. IEEE, Los Alamitos (1995)
16. Chutinan, A., Krogh, B.H.: Computational techniques for hybrid system verification. IEEE Trans. Automat. Control 48(1), 64–75 (2003)
17. Geist, S., Reißig, G., Raisch, J.: An approach to the computation of reachable sets of nonlinear dynamic systems – an important step in generating discrete abstractions of continuous systems. In: Domek, S., Kaszyński, R. (eds.) Proc. 11th IEEE Int. Conf. Methods and Models in Automation and Robotics (MMAR), Miedzyzdroje, Poland, August 29-September 1, pp. 101–106 (2005), `www.reiszig.de/gunther/pubs/i05MMAR.abs.html`
18. Grüne, L., Müller, F.: Set oriented optimal control using past information. In: Proc. 2008 Math. Th. of Networks and Systems (MTNS), Blacksburg, Virginia, U.S.A, July 28 - August 1 (2008)
19. Tiwari, A.: Abstractions for hybrid systems. Form. Methods Syst. Des. 32(1), 57–83 (2008); Proc. 7th Intl. Workshop Hybrid Systems: Computation and Control (HSCC), Philadelphia, U.S.A., March 25-27 (2004)
20. Kloetzer, M., Belta, C.: A fully automated framework for control of linear systems from temporal logic specifications. IEEE Trans. Automat. Control 53(1), 287–297 (2008)
21. Tabuada, P.: An approximate simulation approach to symbolic control. IEEE Trans. Automat. Control 53(6), 1406–1418 (2008)
22. Reißig, G.: Convexity of reachable sets of nonlinear ordinary differential equations. Automat. Remote Control 68(9), 1527–1543 (2007); (Russian transl. in Avtomat. i Telemekh. (9), 64–78 (2007), `www.reiszig.de/gunther/pubs/i07Convex.abs.html`
23. Reißig, G.: Convexity of reachable sets of nonlinear discrete-time systems. In: Kaszyński, R. (ed.) Proc. 13th IEEE Int. Conf. Methods and Models in Automation and Robotics (MMAR), Szczecin, Poland, August 27-30, pp. 199–204 (2007), `www.reiszig.de/gunther/pubs/i07MMAR.abs.html`
24. Hirsch, M.W., Smale, S.: Differential Equations, Dynamical Systems, and Linear Algebra. Pure and Appl. Math., vol. 60. Academic Press, London (1974)
25. Hartman, P.: Ordinary differential equations. Classics in Applied Mathematics, vol. 38. SIAM, Philadelphia (2002)
26. Halin, R.: Graphentheorie, 2nd edn. Wiss. Buchgesellschaft, Darmstadt (1989)

27. Rockafellar, R.T.: Convex analysis. Princeton Mathematical Series, vol. 28. Princeton University Press, Princeton (1970)
28. Reißig, G.: Local fill reduction techniques for sparse symmetric linear systems. Electr. Eng. 89(8), 639–652 (2007), www.reiszig.de/gunther/pubs/i06Fill.abs.html
29. Reißig, G.: Fill reduction techniques for circuit simulation. Electr. Eng. 90(2), 143–146 (2007), www.reiszig.de/gunther/pubs/i07Fill.abs.html
30. Zampieri, G., Gorni, G.: Local homeo- and diffeomorphisms: invertibility and convex image. Bull. Austral. Math. Soc. 49(3), 377–398 (1994)
31. Polyak, B.T.: Convexity of nonlinear image of a small ball with applications to optimization. Set-Valued Anal. 9(1-2), 159–168 (2001)
32. Bobylev, N.A., Emelýanov, S.V., Korovin, S.K.: Convexity of images of convex sets under smooth maps. Nelineinaya Dinamika i Upravlenie (2), 23–32 (2002); Russian. Engl. transl. in Comput. Math. Model. 15(3), 213–222
33. Wolfram, S.: The Mathematica® book, 5th edn. Wolfram Media, Inc., Champaign (2003)
34. von Lossow, M.: A min-max version of Dijkstra's algorithm with application to perturbed optimal control problems. In: Proceedings in Applied Mathematics and Mechanics ICIAM 2007/GAMM 2007, Zürich, Schweiz, vol. 7 (2007)

Hybrid Modeling, Identification, and Predictive Control: An Application to Hybrid Electric Vehicle Energy Management

G. Ripaccioli[1], A. Bemporad[1], F. Assadian[2], C. Dextreit[2], S. Di Cairano[3], and I.V. Kolmanovsky[3]

[1] Dept. of Information Engineering, University of Siena, Italy
{ripaccioli,bemporad}@dii.unisi.it
[2] Jaguar Land Rover Research
{fassadia,cdextrei}@jaguarlandrover.com
[3] Ford Motor Company, Dearborn, Michigan, USA
{sdicaira,ikolmano}@ford.com

Abstract. Rising fuel prices and tightening emission regulations have resulted in an increasing need for advanced powertrain systems and systematic model-based control approaches. Along these lines, this paper illustrates the use of hybrid modeling and model predictive control for a vehicle equipped with an advanced hybrid powertrain. Starting from an existing high fidelity nonlinear simulation model based on experimental data, the hybrid dynamical model is developed through the use of linear and piecewise affine identification methods. Based on the resulting hybrid dynamical model, a hybrid MPC controller is tuned and its effectiveness is demonstrated through closed-loop simulations with the high-fidelity nonlinear model.

Keywords: Hybrid systems, model predictive control, powertrain control, hybrid electric vehicles, piecewise affine systems, piecewise affine system identification.

1 Introduction

The complexity of powertrain systems is increasing in response to tightening fuel economy and emission requirements. In particular, the powertrains have now more subsystems, components, inputs, outputs, operating modes and constraints than in the past. Their effective treatment benefits from systematic modeling and model-based control approaches.

In the paper we demonstrate how a hybrid dynamical model of an advanced powertrain can be developed using linear and piecewise affine identification techniques. The resulting hybrid model can be used as a basis for the design of a hybrid Model Predictive Controller which uses mixed integer quadratic programming (MIQP) solvers for the on-line optimization to coordinate commands to powertrain subsystems and enforce pointwise-in-time state and control constraints.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 321–335, 2009.

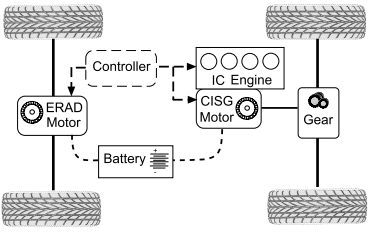


Fig. 1. Configuration of the 4x4 hybrid electric vehicle

A specific case study based on a parallel Hybrid Electric Vehicle (HEV) introduced in [1] has been chosen to demonstrate the proposed approach. This vehicle relies on two electric motors (one in the front and one in the rear of the vehicle), in addition to a turbocharged diesel engine and a high voltage battery. A realistic simulation model with detailed component representations will be used as a basis for deriving a hybrid model; the hybrid model will than be used for prediction in a model predictive control (MPC) strategy. Any upfront manual simplification of the simulation model is avoided to demonstrate how hybrid modeling and piecewise affine system identification techniques can be directly and systematically applied to the high fidelity industrial models.

2 Model

Our case study is an advanced 4x4 hybrid electric vehicle configuration discussed by Dextreit et al., in [1]. This vehicle is equipped with a turbocharged diesel engine, a high voltage electric battery and two electric motors one acting on the front axis and one acting on the rear axis. The front electric motor is the Crankshaft Integrated Starter Generator (CISG), which is directly mounted on the engine crankshaft and is used for starting and assisting the engine and for generating electric energy. An Electric Rear Axle Drive (ERAD) motor is located on the rear differential. The ERAD can operates as a traction motor to drive the rear wheels or as generator, either during regenerative braking or when the battery needs to be charged.

Our developments are based on a high fidelity simulation model of the overall vehicle. The simulation model is based on the nonlinear maps of the HEV components, including nonlinear models of battery and vehicle dynamics, and switching components such as gears. The simulation model can be subdivided into the following subsystems:

Electrical battery, describes the dynamics of the NiMH high voltage battery on board of the vehicle. The model equations are

$$\frac{dSoC(t)}{dt} = \frac{P_w(t)}{V_{batt}(t)} \cdot \frac{1}{C_{Ch}}, \quad V_{batt}(t) = OCV(t) - \frac{P_w(t)}{V_{batt}(t)} \cdot R(t), \qquad (1)$$

$$\frac{dOCV(t)}{dt} = \frac{P_w(t)}{V_{batt}(t)} \cdot f_1(SoC(t)) - OCV(t), \quad R(t) = f_2(SoC(t), T(t)),$$

where $0 \leq SoC \leq 1$ is the State of Charge of the battery, P_w (W) is the electrical power entering the battery, V_{batt} (V) is the output voltage, C_{Ch} (F) is the charge capacity, OCV (V) is the open circuit voltage, R (Ω) is the internal resistance, and T (°C) is the temperature. The input of (1) is the power requested ($P_w \leq 0$) or generated ($P_w \geq 0$) by the electrical motors and by the auxiliary devices. The outputs are the actual voltage V_{batt}, the delivered current $I_{batt} = \frac{P_w}{V}$, and the state of charge SoC.

Vehicle longitudinal dynamics model. The model which describes the longitudinal vehicle dynamics has the form

$$M_{tot}(t)\dot{v}_{veh}(t) = F(t), \tag{2}$$

where $M_{tot}(t)$ (kg) is the sum of the vehicle mass M and the inertial mass $M_i(t)$. Here the inertial mass is calculated as the ratio between the overall inertia at wheels $J(t)$ (kg m^2) and the square of the wheel radius r_w (m). In (2), $F(t)$ (N) is the sum of all the equivalent forces acting on the vehicle

$$F(t) = F_{drl}(t) + F_{ae}(t) + F_{rol}(t) + F_{brake}(t) + F_{gr}(t) \tag{3}$$

The forces involved in (3) are the driving force F_{drl}, which is a function of the total torque at wheels $\tau_{wheel} = \tau_{fdrl} + \tau_{\text{ERAD}}$ (Nm) applied by the motors, the aerodynamic force $F_{ae}(t)$, the rolling resistance forces, $F_{rol}(t)$ and the braking force $F_{brake}(t)$ which are functions of the vehicle speed, and the force due to the gravity on a non-zero road grade, $F_{gr}(t)$. The inputs in (2), (3) are the torques applied to the wheels $\tau_{tot} = r_w \cdot F(t)$ coming from the driveline subsystem. The output is the vehicle speed v_{veh} (m/s).

Powerplant model models the internal combustion (IC) engine through different maps which characterize its instantaneous efficiency, fuel consumption, and operating limits. The inputs are the torque requested $\tau_{\text{IC,req}}$ (Nm) to an existing torque controller, and the actual shaft speed ω_{shaft} (rad/s), which is also the speed of the CISG motor. We denote by J_{IC} (kg m^2) the inertia of the engine. The outputs are the torque τ_{IC} (Nm) actually delivered by the engine, and the fuel flowrate, f_{rate}.

Driveline model is composed by front and rear drivelines. The model of the front driveline includes maps representing the losses along the driveline. The rear driveline model includes maps for losses, efficiency, and limits in generating and motoring modes for the ERAD motor. The inputs are the battery states (1), the torque requested $\tau_{erad,req}$ (Nm), which is positive during the motoring and negative during the generating phase, and the torque τ_{gear} delivered to the front driveline. The outputs are the torque to each wheel and the power P_{ERAD} (W) requested or generated by the ERAD.

Transmission model includes the maps of the CISG and of the gearbox, which characterize the efficiency, the operating limits, and the transmission reductions. The main inputs are the selected gear, $gear_\sharp \in \{N, 1, 2, 3, 4, 5, 6\}$, the battery states, and the requested CISG torque $\tau_{cisg,req}$. The outputs are the transmission

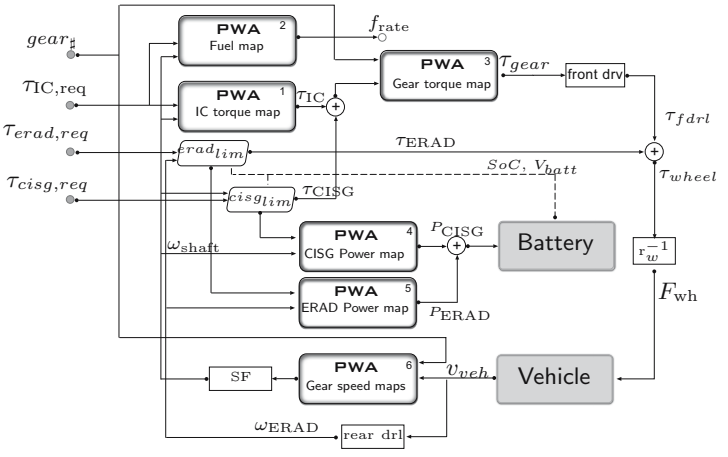


Fig. 2. Overall schematics of the HEV hybrid dynamical model

output torque τ_{gear}, the rotational speed of the gear shaft ω_{shaft}, and the power P_{CISG} (W) drained from or supplied to the battery by the CISG.

3 Hybrid Dynamical Model of the HEV

Hybrid dynamical models have been used in recent years to analyze and optimize a large variety of systems in which physical processes interact with embedded digital controllers and switching devices. Several modeling formalisms have been developed to represent hybrid systems [2,3,4], including Mixed Logical Dynamical (MLD) systems [5], which are discrete-time hybrid models useful to formulate optimization problems involving hybrid dynamics. The language HYSDEL (HYbrid Systems DEscription Language) was developed in [6] to obtain MLD models from a high level textual description of the hybrid dynamics. MLD models can be converted into piecewise affine (PWA) models [7] through automated procedures [8,9]. HYSDEL, MLD and PWA models are used in the Hybrid Toolbox for Matlab$^{\text{TM}}$ [10] for modeling, simulating, and verifying hybrid dynamical systems and for designing hybrid model predictive controllers.

The complex model described in Section 2 is approximated by a discrete-time hybrid model with sampling period $T_s = 1$s that is described in HYSDEL and automatically converted in MLD form. The procedure to obtain such a model involves the following operations:

1. *Linear identification* and time-discretization of the continuous dynamics. Sections 3.2 and 3.3 below describe the identification of discrete-time linear models of the battery (1) and of the vehicle longitudinal dynamics (2), respectively.
2. *Piecewise affine identification.* The nonlinear model is based on interconnected nonlinear maps in the form of lookup tables. These are identified

as static piecewise affine maps through the bounded-error approach [11] to hybrid system identification as detailed below in Section 3.1.

3. *Setup of the HYSDEL model.* Once each subsystem is modeled in discrete-time piecewise affine form, all the submodels are assembled and interconnected in a single HYSDEL model. This is used to generate the corresponding control-oriented MLD model and to synthesize MPC algorithms for the energy management of the HEV under consideration.

The overall hybrid dynamical system is constructed by looking at the energy distribution among the different components that constitute the HEV, rather than at the mechanical devices that compose the nonlinear simulation model, in accordance with the overall scheme depicted in Figure 2. The components of the hybrid dynamical model are described in the following section.

3.1 Piecewise Affine Identification

In order to apply linear hybrid modeling and optimization techniques, nonlinear relations between input/output variables of different subsystems must be approximated by static piecewise affine (PWA) functions. This identification task (or "hybridization" process of the model) is performed algorithmically from input/output data samples. Such samples can be either measured experimentally or obtained by evaluation of existing nonlinear models that have been previously calibrated on measured data. The identification algorithm automatically partitions the input data set into a finite number of polyhedral regions and defines a linear/affine map in each region.

In this paper we use the bounded-error approach of [11] to hybrid system identification. Consider a static PWA model in the form

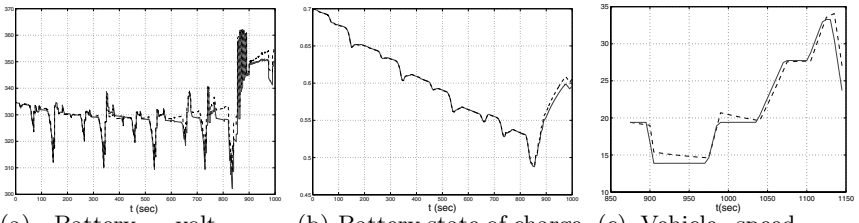
$$y_k = f(\mathbf{u}_k) + \varepsilon_k, \qquad \text{where} \qquad f(\mathbf{u}_k) = \begin{cases} \theta_1' \begin{bmatrix} \mathbf{u}_k \\ 1 \end{bmatrix} & \text{if } \mathbf{u}_k \in \chi_1 \\ \vdots & \vdots \\ \theta_s' \begin{bmatrix} \mathbf{u}_k \\ 1 \end{bmatrix} & \text{if } \mathbf{u}_k \in \chi_s, \end{cases} \tag{4}$$

where $\mathbf{u}_k \in \mathbb{R}^n$ are the input samples, $y_k \in \mathbb{R}$ are the corresponding output samples, $\varepsilon_k \in \mathbb{R}$ are the error terms, $k = 1, \ldots, N$. $\chi_i = \{\mathbf{x} : H_i \mathbf{u}_k \leq K_i\}$, are polyhedral sets defining a partition of the given set of interest $\chi \subseteq \mathbb{R}^n$, and $\theta_i \in \mathbb{R}^{n+1}$, $i = 1, ..., s$, are the parameter vectors defining the affine submodels.

Given the tolerated bound $\delta > 0$ on the fit error ε_k, the bounded-error approach determines a PWA model (4) satisfying the condition $|y_k - f(\mathbf{u}_k)| \leq \delta$. The bound δ is the tuning knob of the procedure. It determines the tradeoff between complexity and accuracy of the model to fit samples. In this paper we have modified the toolbox of [12] to approximate PWA functions based on a maximum "relative" error $\delta_{rel} > 0$

$$\frac{|y_k - f(\mathbf{u}_k)|}{1 + |y_k|} \leq \delta_{rel} \ , \quad \forall k = 1, \ldots, N. \tag{5}$$

Compared to the original absolute error proposed in [11], we have found that the criterion (5) leads to a reduced complexity in terms of number s of affine models.



(a) Battery voltage dynamics V_{batt} (dashed: linear model, solid: nonlinear model)

(b) Battery state of charge dynamics (dashed: linear model, solid: nonlinear model)

(c) Vehicle speed dynamics (dashed: linear model, solid: nonlinear model)

Fig. 3. Open-loop validation of the linear ARX model of the battery and of the vehicle chassis model

Given N data points $(y_k, \mathbf{u}_k)$, $k = 1, \ldots, N$ and a chosen $\delta_{rel} > 0$, the three-step procedure proposed in [11] is applied to look for the minimum positive integer s, for a partition $\chi_1, \ldots, \chi_s$, and for a set of parameter vectors $\theta_1, \ldots, \theta_s$ such that the corresponding PWA model (4) satisfies the bounded error condition (5). As detailed in the next sections, different parameters δ_{rel} were optimized for each identified map, depending on the relevance of the fit error on the dynamic behavior of the overall hybrid system. The N data points for each PWA model are chosen using the response of the nonlinear model controlled by a rule-based controller developed in [1] and a collection of points uniformly distributed on the input range of the nonlinear map. The toolbox of [12] has also been interfaced to the Hybrid Toolbox [10] by automatically generating the HYSDEL code that describes the identified PWA function.

3.2 Battery Model

In order to model the battery described by the nonlinear dynamics (1) in a hybrid form oriented toward the synthesis of MPC controller, the model was approximated as a piecewise affine autoregressive exogenous (PWARX) model via the parametric identification procedure [11].

By restricting the safe range of the State of Charge, $SoC \in [0.2, 0.8]$, neglecting the dependence on temperature (we assumed $T = 25°C$ constant) and assuming that the charging and discharging characteristics are equal, a satisfactory fit has been obtained by a multi-output PWA autoregressive model that consists of only one partition, that is, by the linear autoregressive model

$$\begin{bmatrix} SoC(k) \\ V_{batt}(k) \end{bmatrix} = b_0 P_w(k) + b_1 P_w(k-1) + a_{1s} SoC(k-1) + a_{2s} SoC(k-2) + a_{1v} V_{batt}(k-1) + a_{2v} V_{batt}(k-2) \quad (6)$$

where k denotes the sampling instant for sampling period $T_s = 1$s, a_{1s}, a_{2s}, b_0, b_1, a_{1v}, $a_{2v} \in \mathbb{R}^2$ are the coefficient matrices. The model was validated against the response of the nonlinear model using $N_v = 1000$ samples of a real use of the

battery during a driving cycle. The results of the comparison, obtained using open-loop simulation, are reported in Figure 3. The fit on validation data is $\simeq 96\%$ for the SoC and $\simeq 77\%$ for V_{batt}.

In order to account for modeling errors, the State of Charge constraints enforced by the controller are set tighter than real safety and realistic limits

$$0.3 \leq SoC(t) \leq 0.7, \ \forall t \geq 0. \tag{7}$$

Since in fact the real SoC safe range is is wider, this constraint is treated as soft, i.e., its violations will cause an increased value of the cost, which means that they are tolerable, but only during short transients.

3.3 Vehicle Model

For the purpose of power management only the force F_{drl} delivered by the controlled motors to the driveline, $F_{drl} \geq 0$, is considered as a manipulated input to the linear model (2) of the vehicle longitudinal dynamics. The remaining forces $F_{ae}(t)$, $F_{rol}(t)$, $F_{gr}(t)$ model resistance effects on the car. The braking force F_{brake} is considered as a disturbance, since it is actuated by the driver. The full nonlinear model of the vehicle longitudinal dynamics takes into account the fact that the equivalent inertia of the system is not constant and in particular it depends on the engaged gear. Nonetheless, the simple mass-damper model

$$M\dot{v}_{veh} + \beta v_{veh} = \frac{1}{r_w} \cdot \tau_{\text{wh}} \tag{8}$$

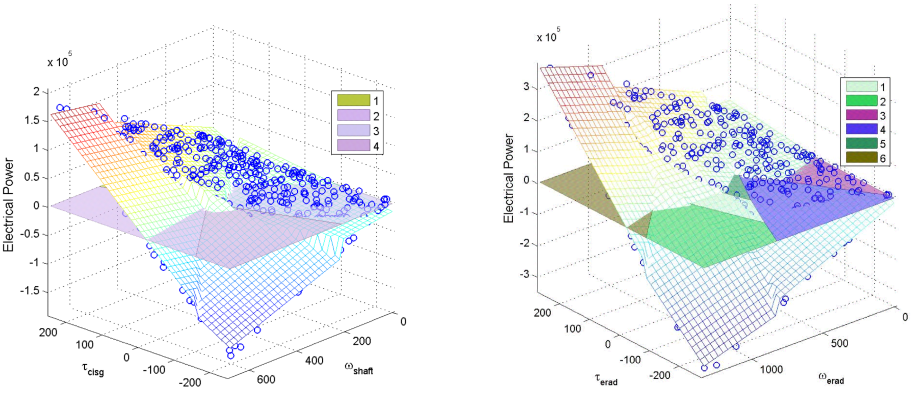
was fit to N simulation data of wheel torque τ_{wh}, speed v_{veh}, and acceleration $\dot{v}_{veh}$ obtaining a good approximation. The parameters M and β were simply estimated by solving the standard least square estimation problem

$$\left[\begin{smallmatrix} M \\ \beta \end{smallmatrix}\right] = (X^T X)^{-1} X^T Y, \ X = \begin{bmatrix} \dot{v}(0) & v(0) \\ \vdots & \vdots \\ \dot{v}(N-1) & v(N-1) \end{bmatrix}, \ Y = \frac{1}{r_w} \begin{bmatrix} \tau(0) \\ \vdots \\ \tau(N-1) \end{bmatrix}. \tag{9}$$

Figure 3(c) compares the vehicle speed signal generated by the open-loop simulation of the estimated linear model excited by τ_{wh} against the vehicle speed signal obtained by simulating the full nonlinear model. The open-loop simulation error over a period of 300 s is bounded and does not tend to diverge; it is smaller than 2 m/s for the most part of the simulation.

3.4 Internal Combustion Engine

Since the aim of the hybrid dynamical model is to synthesize a control algorithm for managing power flows within HEV, the engine and its low-level torque regulator are modeled as a subsystem whose inputs are the desired torque $\tau_{\text{IC,req}}$ to the crankshaft and engine speed ω_{shaft}, and whose outputs are the actual delivered torque τ_{IC} and the fuel flowrate f_{rate}, therefore assuming torque generation dynamics are fast enough to be negligible. This assumption is justified by the



(a) CISG electrical motor (b) ERAD electrical motor

Fig. 4. PWA maps of the electric motors

fact that energy management is performed at a much slower rate than torque control. Accordingly, two of the following PWA output maps were identified

$$\tau_{\mathrm{IC,req}} - \tau_{\mathrm{IC}} = f_{PWA,\tau}(\tau_{\mathrm{IC,req}}, \omega_{\mathrm{shaft}}), \tag{10}$$

which consists of 4 regions, with a fit error below 10%, and

$$f_{\mathrm{rate}} = f_{PWA,f}(\tau_{\mathrm{IC,req}}, \omega_{\mathrm{shaft}}), \tag{11}$$

which consists of 5 regions, with a fit below 5%.

3.5 Electric Motors

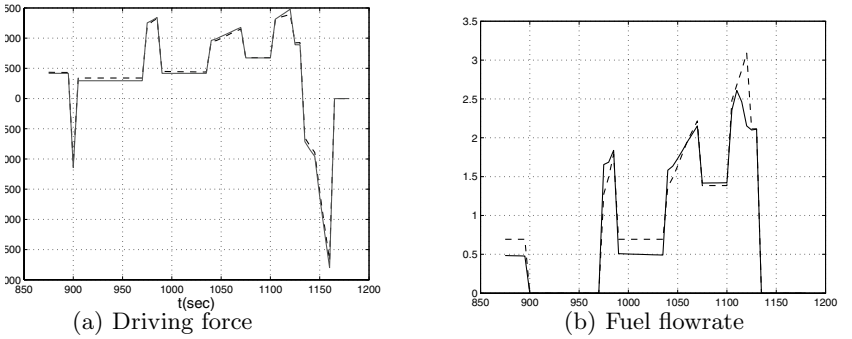
The electric motors are assumed to have fast dynamics and generate torque equal to the requested torque unless limits are exceeded by the requested torque, in which case the actual torque is saturated. The limits are not modeled in the hybrid model, but rather calculated and imposed by the MPC controller externally through a piecewise affine bound. The elimination of the limit maps from the model is justified by the fact that the saturation limits are never reached in simulation as long as the constraints on the State of Charge SoC of the battery are enforced, and this reduces the complexity of the hybrid dynamical model. The mechanical power delivered by the CISG and ERAD motors are $P_{\mathrm{CISG,mec}} = \tau_{\mathrm{CISG}} \cdot \omega_{\mathrm{shaft}}$ and $P_{\mathrm{ERAD,mec}} = \tau_{\mathrm{ERAD}} \cdot \omega_{\mathrm{ERAD}}$, respectively. The following PWA maps represent the actual delivered electrical power $P_{\mathrm{CISG,ele}}$ and $P_{\mathrm{ERAD,ele}}$

$$P_{\mathrm{CISG,ele}} = f_{PWA,c}(\tau_{\mathrm{CISG}}, \omega_{\mathrm{shaft}}), \;\; P_{\mathrm{ERAD,ele}} = f_{PWA,e}(\tau_{\mathrm{ERAD}}, \omega_{\mathrm{ERAD}}). \tag{12}$$

The functional relationships in (12) have been identified from the full nonlinear model and are reported in Figure 3.4. The maps (12) incorporate the effect of electro-mechanical losses.

Table 1. Elecrical Motor PWA maps limits

	max speed (rad/s)	torque range (Nm)	max power loss (kW)
CISG	700	-200 ÷ 200	13
ERAD	1300	-300 ÷ 300	21



(a) Driving force (b) Fuel flowrate

Fig. 5. Open-loop validation of the IC engine model: PWA model (dashed), nonlinear model (solid)

The ERAD motor operates in a wider range than the CISG motor (see Table 1). The latter mainly assists the IC engine. A reasonable tradeoff between accuracy of the maps and model complexity has been reached by setting a relative maximum fitting error of 15% in the PWA identification algorithm for both maps. The number of regions for the ERAD and CISG electrical power maps is 6 and 4, respectively.

3.6 Gear Model

The model of the gearbox is split in two different maps. As sketched in Figure 2 the PWA map of gear torque models the effects of the mechanical reduction on the torque entering the gearbox, $\tau_{\text{ingear}} = \tau_{\text{IC}} + \tau_{\text{CISG}}$ [Nm], as a function of the selected gear, $gear_\sharp$,

$$\tau_{gear} = f_{PWA}(\tau_{\text{ingear}}, gear_\sharp). \tag{13}$$

A second one-dimensional map defines the transmission ratio $TR_\sharp$ for each gear, where $\sharp$ denotes gear number. The transmission ratio relates the shaft speed ω_{shaft} [rad/s] and the actual vehicle speed v_{veh} [m/s]

$$\omega_{\text{shaft}} = v_{veh} \cdot TR_\sharp \cdot SF, \tag{14}$$

where SF is the scaling factor due to the front differential.

3.7 Overall Hybrid Dynamical Model

The overall hybrid dynamical model is constructed according to the structure depicted in Figure 2, where each component has been approximated through linear or piecewise affine identification as described in the previous sections. The model has been validated by running an open-loop simulation on the New European Driving Cycle (NEDC), which defines a vehicle speed reference profile, $v_{veh,ref}$, to be tracked for a duration of 20 minutes, along with the gear to engage. Figure 5(a) reports the traction force acting on the vehicle, Figure 5(b) the fuel flowrate consumed by the vehicle. The quality of the fit is considered adequate, as the mere role of the model is to predict the behavior of the HEV over a short time horizon as required for model predictive control.

In order to track the vehicle speed with zero steady-state offset the model is extended by introducing integral action. The sampled desired vehicle speed $v_{veh,ref}$ and the integral $I_{v,err}$ of the difference between $v_{veh,ref}$ and v_{veh} are included as additional states

$$\begin{aligned} v_{veh,ref}(k+1) &= v_{veh,ref}(k) \\ I_{v,err}(k+1) &= I_{v,err}(k) + T_s(v_{veh}(k) - v_{veh,ref}(k)) \end{aligned}$$

where $T_s = 1$s is the sampling period. With the aim of reducing the prediction horizon of the hybrid MPC controller based on the hybrid dynamical model developed above, rather than considering the tracking error of the state of charge we consider its one step ahead prediction, obtained by iterating (6) for one step under the assumption that the electrical power satisfies $P_w(k+1) = P_w(k)$.

The braking force F_{brake} from the driver is also modeled as a constant state $F_{brake}(k+1) = F_{brake}(k)$, although it will be assumed to be unknown in the following simulations by the controller, and hence set $F_{brake} = 0$.

The overall hybrid dynamical model has been described in HYSDEL and converted to MLD form using the Hybrid Toolbox for Matlab [10]. The resulting MLD model has 9 continuous states ($v_{veh}(k)$, $I_{v,err}(k)$, $v_{veh,ref}(k)$, $SoC(k)$, $SoC(k-1)$, $V_{batt}(k)$, $V_{batt}(k-1)$, $P_w(k-1)$, $F_{brake}(k)$), 7 binary states storing the current engaged gear (Neutral, 1st, ..., 6th) and subject to an exclusive-or constraint, 3 continuous inputs ($\tau_{\mathrm{IC,req}}$, $\tau_{ERAD,req}$, $\tau_{CISG,req}$), 32 binary inputs used to detect the active regions in the 6 PWA maps (for each map the corresponding group of binary inputs is subject to an exclusive-or constraint), 56 continuous auxiliary variables, used for representing the PWA maps, engine speed, engine torque, and other ancillary variables, 1 continuous output, fuel consumption f_{rate}, no binary auxiliary variables, and 490 mixed-integer inequalities.

4 Model Predictive Control Design

MPC was used in many industrial applications [13], and more recently model predictive control of hybrid dynamical systems has shown potential for applications in the automotive domain [14,15,16,17,18,19]. In this section we design an MPC controller for the HEV based on the overall hybrid dynamical model

described in Section 3. In the MPC approach, at each sampling instant a finite horizon open-loop optimization problem is solved, using the current state as the initial condition of the problem. The optimization provides a control sequence, only the first element of which is applied to the process. This process is iteratively repeated at each subsequent time instant, thereby providing a feedback mechanism for disturbance rejection and reference tracking. The optimal control problem is defined as:

$$\min_{\xi} J(\xi, x(t)) \triangleq Q_\rho \rho^2 + \sum_{k=1}^{N} (\Gamma_x x_k - x_{ref})^T S (\Gamma_x x_k - x_{ref}) + \tag{15a}$$
$$+ \sum_{k=0}^{N-1} (\Gamma_u u_k - u_{ref})^T R (\Gamma_u u_k - u_{ref}) + (y_k - y_{ref})^T Q (y_k - y_{ref}),$$

$$\text{subj. to} \begin{cases} x_0 & = x(t), \\ x_{k+1} & = A x_k + B_1 u_k + B_3 z_k, \\ y_k & = C x_k + D_1 u_k + D_3 z_k, \\ E_3 z_k & \leq E_1 u_k + E_4 x_k + E_5, \\ 0.3 - \rho \leq & SoC_k \leq 0.7 + \rho, \end{cases} \tag{15b}$$

where N is the control horizon, $x(t)$ is the state of the MLD system at sampling time t, $\xi \triangleq [u_0^T, z_0^T, \ldots, u_{N-1}^T, z_{N-1}^T, \rho]^T \in \mathbb{R}^{59N+1} \times \{0,1\}^{32N}$ is the optimization vector, Q, R and S are weight matrices, Q_ρ is a large weight used to enforce the softened version (15b) of constraint (7), and $\Gamma_u \in \mathbb{R}^{3\times 36}$, $\Gamma_x \in \mathbb{R}^{3\times 16}$ are matrices that select the subset of vector components to be weighted (Γ_u, Γ_x are formed by rows of identity matrices). In particular we define the reference signals used in (15) for the output and for the components selected by Γ_u, Γ_x as

$$y_{ref} \triangleq f_{\text{rate,ref}}, \tag{16a}$$
$$u_{ref} \triangleq [\tau_{\text{IC,req}} \ \tau_{\text{ERAD,req}} \ \tau_{\text{CISG,req}}]', \tag{16b}$$
$$x_{ref} \triangleq [v_{\text{veh,ref}} \ \ I_{\text{v,err}} \ \ SoC_{\text{ref}}]', \tag{16c}$$

and, accordingly, we set the cost weights in (15b) to be

$$Q = q_{\text{fuel}}, \quad R = \begin{bmatrix} r_{\tau,\text{IC}} & 0 & 0 \\ 0 & r_{\tau,\text{CISG}} & 0 \\ 0 & 0 & r_{\tau,\text{ERAD}} \end{bmatrix}, \quad S = \begin{bmatrix} s_{\text{v,veh}} & 0 & 0 \\ 0 & s_{\text{SoC}} & 0 \\ 0 & 0 & s_{\text{v,int}} \end{bmatrix}, \ Q_\rho = 10^5,$$

where the components of vector u_{ref} are all zero in order to minimize the control action.

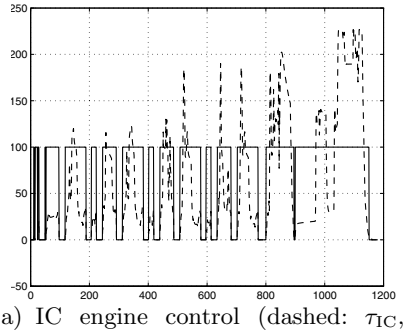
Problem (15) can be transformed into a mixed integer quadratic program (MIQP), i.e., into the minimization of a quadratic cost function subject to linear constraints, where some of the variables are binary. Even if this class of problems has exponential complexity, efficient numerical tools for its solution are available [20].

5 Simulation Results

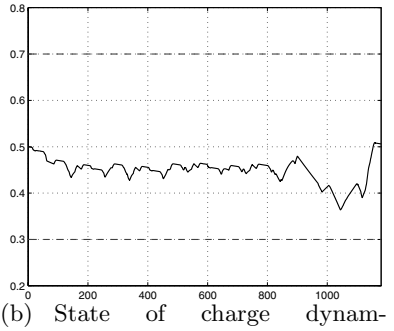
The closed-loop behavior of the HEV in closed loop with MPC controller has been evaluated in simulations by using the high-fidelity nonlinear model described in Section 2. The design parameters for the MPC (15) are the prediction

Table 2. MPC design parameters ($r_{\tau,\mathrm{IC}} = 6{\cdot}10^{-2}$, $r_{\tau,\mathrm{CISG}} = 3{\cdot}10^{-2}$, $r_{\tau,\mathrm{ERAD}} = 3{\cdot}10^{-2}$, $s_{\mathrm{v,veh}} = 5{\cdot}10^{3}$). The number in the first column represents the MPC design number (0 = conventional vehicle). The fuel consumption values are normalized to the conventional vehicle consumption.

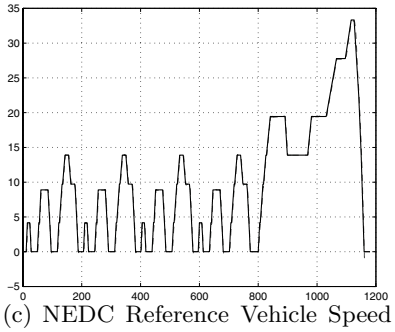
	q_{fuel}	s_{SoC}	$s_{\mathrm{v,int}}$	fuel cons (norm)	$\max \lvert v_{veh} - v_{veh,ref} \rvert$	$\max \lvert SoC - SoC_{ref} \rvert$
0	*	*	*	1	*	*
1	1e-2	2e6	10	0.79	2.105	0.1364
2	1e1	1e6	1	0.76	2.789	0.2484



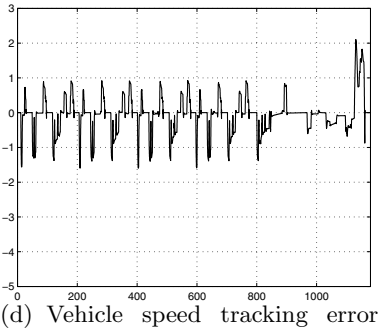
(a) IC engine control (dashed: τ_{IC}, solid: IC engine on/off signal)



(b) State of charge dynamics(dashed: SoC bounds, solid: SoC dynamics)



(c) NEDC Reference Vehicle Speed profile

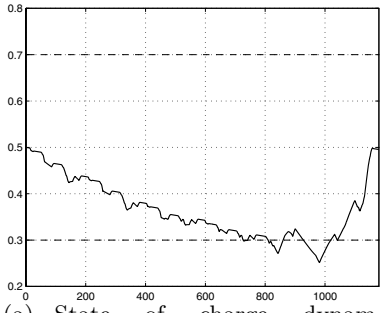


(d) Vehicle speed tracking error ($v_{\mathrm{veh}} - v_{\mathrm{veh,ref}}$)

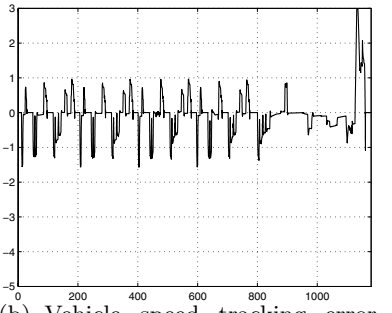
Fig. 6. MPC design #1: closed-loop response

horizon $N = 1$, and weights $r_{\tau,\mathrm{IC}} = 6{\cdot}10^{-2}$, $r_{\tau,\mathrm{CISG}} = 3{\cdot}10^{-2}$, $r_{\tau,\mathrm{ERAD}} = 3{\cdot}10^{-2}$, $s_{\mathrm{v,veh}} = 5 \cdot 10^{3}$. The weights q_{fuel}, s_{SoC} and $s_{\mathrm{v,int}}$ are reported in Table 2 for two different MPC designs. Note that the weight on $r_{\tau,\mathrm{IC}}$ is much greater than $r_{\tau,\mathrm{CISG}}$, $r_{\tau,\mathrm{ERAD}}$ to force the use of torque from electric motors rather than from the IC engine, and that $s_{\mathrm{v,int}}$ is used to maintain the speed tracking performance.

For both controllers it took approximately 175.5 s to simulate the closed-loop system on a PC Intel Centrino Duo 2.0 GHz with 2GB RAM running the Hybrid Toolbox for Matlab [10] and the MIQP solver of CPLEX 9 [20], of which 156.7 s



(a) State of charge dynamics(dashed: SoC bounds, solid: SoC dynamics)



(b) Vehicle speed tracking error ($v_{\text{veh}} - v_{\text{veh,ref}}$)

Fig. 7. MPC design #2: closed-loop response

are spent by CPLEX, that is an average of approximately 0.13 s per time step. The control action is computed in the worst case in approximately 0.29 s. The closed-loop dynamics obtained from the first MPC design are described in Figure 6.

In this simulation q_{fuel} has a small weight as the fuel consumption is less important than keeping the battery SoC close to the setpoint. To improve fuel consumption the internal combustion engine is turned off when the torque request is lower than a given threshold, see Figure (6(a)). The results of the second MPC design are shown in Figure 7, where a higher emphasis to fuel consumption is given, where more freedom to draw power from the battery (lower s_{SoC}) is allowed to the controller, which also has a lower weight on the speed tracking integral action ($s_{\text{v,int}}$). The weight on $s_{\text{v,veh}}$ allowed to maintain the maximum error in speed tracking smaller than 3.2 [m/s]. The SoC signal violates the soft constraint (15b) on minimum charge for a maximum time of 92s. However, it should be noted that the SoC always remains in the physical battery safety and reliability range $SoC \in [0.2, 0.8]$. For both MPC designs the fuel consumption is reduced with respect to a conventional vehicle. In the first simulation the fuel consumption improvement is 20.7%. In the second simulation the controller is allowed to use more electric power due to smaller weight on s_{SoC}, and this results a slight violation of the soft constraint. On the other hand the fuel consumption improvement is 23.8%. These improvements are similar to the values reported in [1], but it is interesting to observe that in this paper the MPC controller does not exploit any knowledge of the driving cycle but only of the vehicle model.

6 Conclusions

In the paper we have exemplified an effective control approach for advanced powertrain systems which combines hybrid modeling, identification and model predictive control. In this approach, piecewise affine system identification techniques serve as a bridge between detailed nonlinear simulation models (or experimental

powertrain hardware) and hybrid models in such a way that the on-line implementation of model predictive control becomes feasible using a mixed integer quadratic programming. In the paper, a realistic (industrial strength) simulation model with high fidelity components representation was used as a basis for deriving an approximate hybrid model: the latter was used to define the hybrid MPC optimization problem. This design approach could have been equally applied to experimental vehicle data or to a mixture of experimental data and simulation data.

References

1. Dextreit, C., Assadian, F., Kolmanovsky, I.V., Mahtani, J., Burnham, K.: Hybrid electric vehicle energy management using game theory. In: Proceedings of SAE World Congress, Detroit, MI (April 2008)
2. Branicky, M.S.: Studies in hybrid systems: modeling, analysis, and control. PhD thesis, LIDS-TH 2304, Massachusetts Institute of Technology, Cambridge, MA (1995)
3. Heemels, W.P.M.H., De Schutter, B., Bemporad, A.: Equivalence of hybrid dynamical models. Automatica 37(7), 1085–1091 (2001)
4. Lygeros, J., Johansson, K.H., Simic, S.N., Zhang, J., Sastry, S.S.: Dynamical properties of hybrid automata. IEEE Trans. Automatic Control 48, 2–17 (2003)
5. Bemporad, A., Morari, M.: Control of systems integrating logic, dynamics, and constraints. Automatica 35(3), 407–427 (1999)
6. Torrisi, F.D., Bemporad, A.: HYSDEL — A tool for generating computational hybrid models. IEEE Trans. Contr. Systems Technology 12(2), 235–249 (2004)
7. Sontag, E.D.: Nonlinear regulation: The piecewise linear approach. IEEE Trans. Automatic Control 26(2), 346–358 (1981)
8. Bemporad, A.: Efficient conversion of mixed logical dynamical systems into an equivalent piecewise affine form. IEEE Trans. Automatic Control 49(5), 832–838 (2004)
9. Geyer, T., Torrisi, F.D., Morari, M.: Efficient Mode Enumeration of Compositional Hybrid Models. In: Maler, O., Pnueli, A. (eds.) HSCC 2003. LNCS, vol. 2623, pp. 216–232. Springer, Heidelberg (2003)
10. Bemporad, A.: Hybrid Toolbox – User's Guide (January 2004), http://www.dii.unisi.it/hybrid/toolbox
11. Bemporad, A., Garulli, A., Paoletti, S., Vicino, A.: A bounded-error approach to piecewise affine system identification. IEEE Trans. Automatic Control 50(10), 1567–1580 (2005)
12. Paoletti, S., Roll, J.: PWAID: Piecewise affine system identification toolbox (2007)
13. Qin, S.J., Badgwell, T.A.: A survey of industrial model predictive control technology. Control Engineering Practice 11(7), 733–764 (2003)
14. Borrelli, F., Bemporad, A., Fodor, M., Hrovat, D.: An MPC/hybrid system approach to traction control. IEEE Trans. Contr. Systems Technology 14(3), 541–552 (2006)
15. Giorgetti, N., Ripaccioli, G., Bemporad, A., Kolmanovsky, I.V., Hrovat, D.: Hybrid Model Predictive Control of Direct Injection Stratified Charge Engines. IEEE/ASME Transactions on Mechatronics 11(5), 499–506 (2006)

16. Di Cairano, S., Bemporad, A., Kolmanovsky, I., Hrovat, D.: Model predictive control of magnetically actuated mass spring dampers for automotive applications. Int. J. Control 80(11), 1701–1716 (2007)
17. Vašak, M., Baotić, M., Morari, M., Petrović, I., Perić, N.: Constrained optimal control of an electronic throttle. Int. J. Control 79(5), 465–478 (2006)
18. Corona, D., De Schutter, B.: Adaptive cruise control for a smart car: A comparison benchmark for MPC-PWA control methods. IEEE Trans. Contr. Systems Technology 16(2), 365–372 (2008)
19. Ortner, P., del Re, L.: Predictive control of a diesel engine air path. IEEE Trans. Contr. Systems Technology 15(3), 449–456 (2007)
20. ILOG, Inc. CPLEX 9.0 User Manual. Gentilly Cedex, France (2003)

On Event Based State Estimation

Joris Sijs[1] and Mircea Lazar[2]

[1] TNO Science and Industry
2600 AD Delft, The Netherlands
joris.sijs@tno.nl
[2] Eindhoven University of Technology
5600 MB Eindhoven, The Netherlands
m.lazar@tue.nl

Abstract. To reduce the amount of data transfer in networked control systems and wireless sensor networks, measurements are usually taken only when an event occurs, rather than at each synchronous sampling instant. However, this complicates estimation and control problems considerably. The goal of this paper is to develop a state estimation algorithm that can successfully cope with event based measurements. Firstly, we propose a general methodology for defining event based sampling. Secondly, we develop a state estimator with a hybrid update, i.e. when an event occurs the estimated state is updated using measurements; otherwise the update makes use of the knowledge that the monitored variable is within a bounded set that defines the event. A sum of Gaussians approach is employed to obtain a computationally tractable algorithm.

1 Introduction

Different methods for state estimation have been introduced during the last decades. Each method is specialized in the type of process, the type of noise or the type of system architecture. In this paper we focus on the design of a state estimator that can efficiently cope with event based sampling. By event sampling we mean that measurements are generated only when an a priori defined event occurs in the data monitored by sensors. Such an estimator is very much needed in networked control systems and wireless sensor networks (WSNs) [1]. Especially in WSNs, where the limiting resource is energy, data transfer and processing power must be minimized. Existing estimators that could be used in this framework are discussed in Section 4. For related research on event based control, the interested reader is referred to the recent works [2,3,4,5,6].

The contribution of this paper is twofold. Firstly, using standard probability notions we set up a general mathematical description of event sampling depending on time and previous measurements. We assume that the estimator does not have information about when new measurements are available, which usually results in an unbounded error-covariance matrix. To prevent this from happening, we develop an estimation algorithm with hybrid update, which is the second main contribution. The developed event based estimator is updated both when an event occurs, with a received measurement sample, as well as at sampling instants synchronous in time, without receiving a measurement sample. In the latter case the update makes use of the knowledge that the monitored

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 336–350, 2009.

variable, i.e. the measurement, is within a bounded set that defines the event. In order to meet low processing power specifications, the proposed state estimator is based on the Gaussian sum filter [7,8], which is known to be computationally tractable.

2 Background Notions and Notation

$\mathbb{R}$ defines the set of real numbers whereas the set $\mathbb{R}_+$ defines the non-negative real numbers. The set $\mathbb{Z}$ defines the integer numbers and $\mathbb{Z}_+$ defines the set of non-negative integer numbers. The notation $\underline{0}$ is used to denote either the null-vector or the null-matrix. Its size will become clear from the context.

Suppose a vector $x(t) \in \mathbb{R}^n$ depends on time $t \in \mathbb{R}$ and is sampled using some sampling method. Two different sampling methods are discussed. The first one is time sampling in which samples are generated whenever time t equals some predefined value. This is either synchronous in time or asynchronous. In the synchronous case the time between two samples is constant and defined as $t_s \in \mathbb{R}_+$. If the time t at sampling instant $k_a \in \mathbb{Z}_+$ is defined as t_{k_a}, with $t_{0_a} := 0$, we define:

$$x_{k_a} := x(t_{k_a}) \quad \text{and} \quad x_{0_a:k_a} := (x(t_{0_a}), x(t_{1_a}), \cdots, x(t_{k_a})).$$

The second sampling method is event sampling, in which samples are taken only when an event occurs. If t at event instant $k_e \in \mathbb{Z}_+$ is defined as t_{k_e}, with $t_{0_e} := 0$, we define:

$$x_{k_e} := x(t_{k_e}) \quad \text{and} \quad x_{0_e:k_e} := (x(t_{0_e}), x(t_{1_e}), \cdots, x(t_{k_e})).$$

A transition-matrix $A_{t_2-t_1} \in \mathbb{R}^{a\times b}$ relates the vector $u(t_1) \in \mathbb{R}^b$ to a vector $x(t_2) \in \mathbb{R}^a$ as follows: $x(t_2) = A_{t_2-t_1}u(t_1)$.

The transpose, inverse and determinant of a matrix $A \in \mathbb{R}^{n\times n}$ are denoted as $A^\top, A^{-1}$ and $|A|$ respectively. The i^{th} and maximum eigenvalue of a square matrix A are denoted as $\lambda_i(A)$ and $\lambda_{max}(A)$ respectively. Given that $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times n}$ are positive definite, denoted with $A \succ 0$ and $B \succ 0$, then $A \succ B$ denotes $A - B \succ 0$. $A \succeq 0$ denotes A is positive semi-definite.

The probability density function (PDF), as defined in [9] section B2, of the vector $x \in \mathbb{R}^n$ is denoted with $p(x)$ and the conditional PDF of x given $u \in \mathbb{R}^q$ is denoted as $p(x|u)$. The expectation and covariance of x are denoted as $E[x]$ and $cov(x)$ respectively. The conditional expectation of x given u is denoted as $E[x|u]$. The definitions of $E[x]$, $E[x|u]$ and $cov(x)$ can be found in [9] sections B4 and B7.

The Gaussian function (shortly noted as Gaussian) of vectors $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^n$ and matrix $P \in \mathbb{R}^{n\times n}$ is defined as $G(x,u,P) : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^{n\times n} \to \mathbb{R}$, i.e.:

$$G(x,u,P) = \frac{1}{\sqrt{(2\pi)^n|P|}} e^{-0.5(x-u)^\top P^{-1}(x-u)}. \tag{1}$$

If $p(x) = G(x,u,P)$, then by definition it holds that $E[x] = u$ and $cov(x) = P$.

The element-wise Dirac-function of a vector $x \in \mathbb{R}^n$, denoted as $\delta(x) : \mathbb{R}^n \to \{0,1\}$, satisfies:

$$\delta(x) = \begin{cases} 0 & \text{if } x \not\equiv \underline{0}, \\ 1 & \text{if } x \equiv \underline{0}, \end{cases} \quad \text{and} \quad \int_{-\infty}^{\infty} \delta(x)dx = 1. \tag{2}$$

For a vector $x \in \mathbb{R}^n$ and a bounded Borel set [10] $Y \subset \mathbb{R}^n$, the set PDF is defined as $\Lambda_Y(x) : \mathbb{R}^n \to \{0, \nu\}$ with $\nu \in \mathbb{R}$ defined as the Lebesque measure [11] of the set Y, i.e.:

$$\Lambda_Y(x) = \begin{cases} 0 & \text{if } x \notin Y, \\ \nu^{-1} & \text{if } x \in Y. \end{cases} \tag{3}$$

3 Event Sampling

Many different methods for sampling a vector $y(t) \in \mathbb{R}^q$ can be found in literature. The one mostly used is time sampling in which the k_a^{th} sampling instant is defined at time $t_{k_a} := t_{k_a-1} + \tau_{k_a-1}$ for some $\tau_{k_a-1} \in \mathbb{R}_+$. Recall that if $y(t)$ is sampled at t_a it is denoted as y_{k_a}. This method is formalized by defining the observation vector $z_{k_a-1} := (y_{k_a-1}^\top, t_{k_a-1})^\top \in \mathbb{R}^{q+1}$ at sampling instant k_{a-1}. Let us define the set $H_{k_a}(z_{k_a-1}) \subset \mathbb{R}$ containing all the values that t can take between t_{k_a-1} and $t_{k_a-1} + \tau_{k_a-1}$, i.e.:

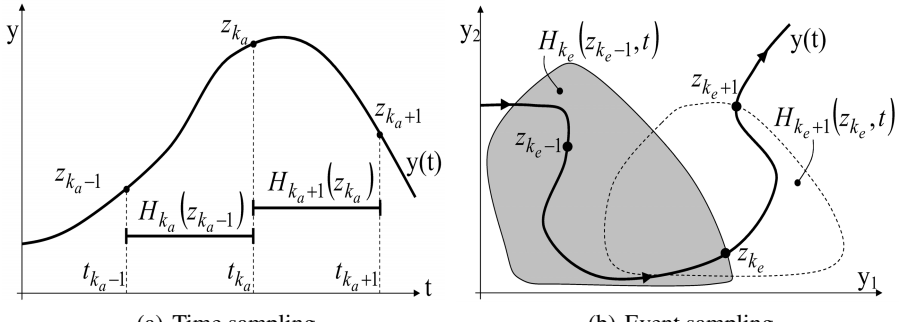
$$H_{k_a}(z_{k_a-1}) := \{t \in \mathbb{R} \,|\, t_{k_a-1} \leq t < t_{k_a-1} + \tau_{k_a-1}\}. \tag{4}$$

Then time sampling defines that the next sampling instant, i.e. k_a, takes place whenever present time t exceeds the set $H_{k_a}(z_{k_a-1})$. Therefore z_{k_a} is defined as:

$$z_{k_a} := (y_{k_a}^\top, t_{k_a})^\top \quad \text{if} \quad t \notin H_{k_a}(z_{k_a-1}). \tag{5}$$

In the case of synchronous time sampling $\tau_{k_a} = t_s$, $\forall k_a \in \mathbb{Z}_+$, which is graphically depicted in Figure 1(a). Notice that with time sampling, the present time t specifies when samples of $y(t)$ are taken, but time t itself is independent of $y(t)$. As a result $y(t)$ in between the two samples can have any value within $\mathbb{R}^q$. Recently, asynchronous sampling methods have emerged, such as, for example "Send-on-Delta" [12, 13] and "Integral sampling" [14]. Opposed to time sampling, these sampling methods are not controlled by time t, but by $y(t)$ itself.

Next, we present a general definition of event based sampling. In this case a sampling instant is specified by an event of $y(t)$ instead of t. As such, one has to constantly check whether the measurement $y(t)$ satisfies certain conditions, which depend on time t and



(a) Time sampling (b) Event sampling

Fig. 1. The two different methods for sampling a signal $y(t)$

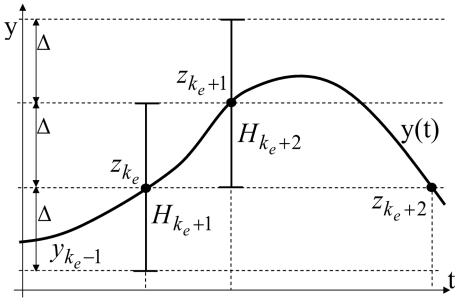


Fig. 2. Event sampling: Send-on-Delta

previous samples of the measurement. This method recovers the above mentioned asynchronous methods, for a particular choice of ingredients. Let us define the observation vector at sampling instant $k_e - 1$ as $z_{k_e-1} := (y_{k_e-1}^\top, t_{k_e-1})^\top \in \mathbb{R}^{q+1}$. With that we define the following bounded Borel set in *time-measurement-space*, i.e. $H_{k_e}(z_{k_e-1}, t) \subset \mathbb{R}^{q+1}$, which depends on both z_{k_e-1} and t. In line with time sampling the next event instant, i.e. k_e, takes place whenever $y(t)$ leaves the set $H_{k_e}(z_{k_e-1}, t)$ as shown in Figure 1(b) for $q = 2$. Therefore z_{k_e} is defined as:

$$z_{k_e} := (y_{k_e}^\top, t_{k_e})^\top \quad \text{if} \quad y(t) \notin H_{k_e}(z_{k_e-1}, t). \tag{6}$$

The exact description of the set $H_{k_e}(z_{k_e-1}, t)$ depends on the actual sampling method. As an example $H_{k_e}(z_{k_e-1}, t)$ is derived for the method "Send-on-Delta", with $y(t) \in \mathbb{R}$. In this case the event instant k_e occurs whenever $|y(t) - y_{k_e-1}|$ exceeds a predefined level Δ, see Figure 2, which results in $H_{k_e}(z_{k_e-1}, t) = \{y \in \mathbb{R} | -\Delta < y - y_{k_e-1} < \Delta\}$.

In event sampling, a well designed $H_{k_e}(z_{k_e-1}, t)$ should contain the set of all possible values that $y(t)$ can take in between the event instants $k_e - 1$ and k_e. Meaning that if $t_{k_e-1} \leq t < t_{k_e}$, then $y(t) \in H_{k_e}(z_{k_e-1}, t)$. A sufficient condition is that $y_{k_e-1} \in H_{k_e}(z_{k_e-1}, t)$, which for "Send-on-Delta" results in $y(t) \in [y_{k_e-1} - \Delta, y_{e-1} + \Delta]$ for all $t_{k_e-1} \leq t < t_{k_e}$.

Besides the event sampling methods discussed above, it is worth to also point out the related works [2,4,3], which focus on event based control systems rather than event based state estimators. Therein event sampling methods are proposed using additional information from the state of the system, which is assumed to be available.

4 Problem Formulation: State Estimation Based on Event Sampling

Assume a perturbed, dynamical system with state-vector $x(t) \in \mathbb{R}^n$, process-noise $w(t) \in \mathbb{R}^m$, measurement-vector $y(t) \in \mathbb{R}^q$ and measurement-noise $v(t) \in \mathbb{R}^q$. This process is

described by a state-space model with $A_\tau \in \mathbb{R}^{n\times n}$, $B_\tau \in \mathbb{R}^{n\times m}$ and $C \in \mathbb{R}^{q\times n}$. An event sampling method is used to sample $y(t)$. The model of this process becomes:

$$x(t+\tau) = A_\tau x(t) + B_\tau w(t), \tag{7a}$$
$$y(t) = Cx(t) + v(t), \tag{7b}$$
$$z_{k_e} = (y_{k_e}^\top, t_{k_e})^\top \quad \text{if} \quad y(t) \notin H_{k_e}(z_{k_e-1}, t), \tag{7c}$$
$$\text{with} \quad p(w(t)) := G(w(t), 0, Q) \quad \text{and} \quad p(v(t)) := G(v(t), 0, V). \tag{7d}$$

The state vector $x(t)$ of this system is to be estimated from the observation vectors $z_{0_e:k_e}$. Notice that the estimated states are usually required at all synchronous time samples k_a, with $t_s = t_{k_a} - t_{k_a-1}$, e.g., as input to a discrete monitoring system (or a discrete controller) that runs synchronously in time. For clarity system (7a) is considered autonomous, i.e. there is no control input. However, the estimation algorithm presented in this paper can be extended to controlled systems.

The goal is to construct an event-based state-estimator (EBSE) that provides an estimate of $x(t)$ not only at the event instants t_{k_e}, at which measurement data is received, but also at the sampling instants t_{k_a}, without receiving any measurement data. Therefore, we define a new set of sampling instants t_n as the combination of sampling instants due to event sampling, i.e. k_e, and time sampling, i.e. k_a:

$$\{t_{0:n-1}\} := \{t_{0_a:k_a-1}\} \cup \{t_{0_e:k_e-1}\} \quad \text{and} \quad t_n := \begin{cases} t_{k_a} & \text{if } t_{k_a} < t_{k_e}, \\ t_{k_e} & \text{if } t_{k_a} \geq t_{k_e}. \end{cases} \tag{8a}$$
$$\text{and} \quad t_0 < t_1 < \cdots < t_n, \quad x_n := x(t_n), \quad y_n := y(t_n). \tag{8b}$$

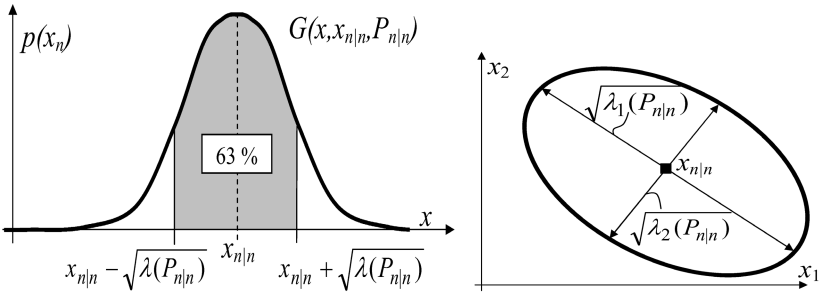
The estimator calculates the PDF of the state-vector x_n given all the observations until t_n. This results in a hybrid state-estimator, for at time t_n an event can either occur or not, which further implies that measurement data is received or not, respectively. In both cases the estimated state must be updated (not predicted) with all information until t_n. Therefore, depending on t_n a different PDF must be calculated, i.e.:

$$\text{if} \quad t_n = t_{k_a} \Rightarrow p(x_n | z_{0_e:k_e-1}) \quad \text{with} \quad t_{k_e-1} < t_{k_a} < t_{k_e}, \tag{9a}$$
$$\text{if} \quad t_n = t_{k_e} \Rightarrow p(x_n | z_{0_e:k_e}). \tag{9b}$$

The performance of the state-estimator is related to the expectation and error-covariance matrix of its calculated PDF. Therefore, from (9) we define:

$$x_{n|n} := \begin{cases} E[x_n | z_{0_e:k_e-1}] & \text{if} \quad t_n = t_{k_a} \\ E[x_n | z_{0_e:k_e}] & \text{if} \quad t_n = t_{k_e} \end{cases} \quad \text{and} \quad P_{n|n} := cov\left(x_n - x_{n|n}\right). \tag{10}$$

The PDFs of (9) are described as the Gaussian $G(x_n, x_{n|n}, P_{n|n})$. Together with $x_{n|n}$, the square root of each eigenvalue of $P_{n|n}$, i.e. $\sqrt{\lambda_i(P_{n|n})}$ (or $\sqrt{\lambda(P_{n|n})}$ if there is only one eigenvalue), indicate the bound which surrounds 63% of the possible values for x_n. This is graphically depicted in Figure 3(a) for the 1D case and Figure 3 for the 2D case, in a top view.



(a) Full view - 1D (b) Top view - 2D

Fig. 3. Two examples of a Gaussian function

As such, the problem of interest in this paper is to construct a state-estimator suitable for the general event sampling method introduced in Section 3 and which is computationally tractable. Also, it is desirable that $P_{n|n}$ has bounded eigenvalues for all n.

Existing state estimators can be divided into two categories. The first one contains estimators based on time sampling: the (a)synchronous Kalman filter [15, 16] (linear process, Gaussian PDF), the Particle filter [17] and the Gaussian sum filter [7, 8] (non-linear process, non-Gaussian PDF). These estimators cannot be directly employed in event based sampling as if no new observation vector z_{k_e} is received, then $t_n - t_{k_e} \to \infty$ and $\lambda_i(P_{n|k_e-1}) \to \infty$. The second category contains estimators based on event sampling. In fact, to the best of our knowledge, only the method proposed in [18] fits this category. However, this EBSE is only applicable in the case of "Send-on-Delta" event sampling and it requires that any PDF is approximated as a single Gaussian function. Moreover, the asymptotic property of $P_{n|n}$ is not investigated in [18].

In the next section we propose a novel event-based state-estimator, suitable for any event sampling method based on the general set-up introduced in Section 3.

5 An Event-Based State Estimator

The EBSE estimates x_n given the received observation vectors until time t_n. Notice that due to the definition of event sampling we can extract information of all the measurement vectors $y_{0:n}$, i.e. also at the instants $t_n = t_{k_a}$, when the estimator does not receive y_{k_a}. For with $t_i \in \{t_{0:n}\}$ and $t_{j_e} \in \{t_{0_e:k_e}\}$ it follows that:

$$\begin{cases} y_i \in H_{j_e}(z_{j_e-1}, t_i) & \text{if } \; t_{j_e-1} \le t_i < t_{j_e}, \\ y_i = y_{j_e} & \text{if } \; t_i = t_{j_e}. \end{cases} \tag{11}$$

Therefore, from the observation vectors $z_{0_e:k_e}$ and (11) the PDF of the hybrid state-estimation of (9), with the bounded, Borel set $Y_i \subset \mathbb{R}^q$, results in:

$$p\left(x_n | y_0 \in Y_0, y_1 \in Y_1, \ldots, y_n \in Y_n\right) \quad \text{with} \tag{12a}$$

$$Y_i := \begin{cases} H_{j_e}(z_{j_e-1}, t_i) & \text{if } \; t_{j_e-1} < t_i < t_{j_e}, \\ \{y_{j_e}\} & \text{if } \; t_i = t_{j_e}. \end{cases} \tag{12b}$$

For brevity (12a) is denoted as $p(x_n|y_{0:n} \in Y_{0:n})$ and with Bayes-rule [19] yields:

$$p\left(x_n|y_{0:n} \in Y_{0:n}\right) := \frac{p\left(x_n|y_{0:n-1} \in Y_{0:n-1}\right) p\left(y_n \in Y_n|x_n\right)}{p\left(y_n \in Y_n|y_{0:n-1} \in Y_{0:n-1}\right)}. \tag{13}$$

To have an EBSE with low processing demand, multivariate probability theory [20] is used to make (13) recursive:

$$p(a|b) := \int_{-\infty}^{\infty} p(a|c)p(c|b)dc \quad \Rightarrow \tag{14a}$$

$$p\left(x_n|y_{0:n-1} \in Y_{0:n-1}\right) = \int_{-\infty}^{\infty} p(x_n|x_{n-1})p\left(x_{n-1}|y_{0:n-1} \in Y_{0:n-1}\right)dx_{n-1}, \tag{14b}$$

$$p\left(y_n \in Y_n|y_{0:n-1} \in Y_{0:n-1}\right) = \int_{-\infty}^{\infty} p\left(x_n|y_{0:n-1} \in Y_{0:n-1}\right) p(y_n \in Y_n|x_n)dx_n. \tag{14c}$$

The calculation of $p(x_n|y_{0:n} \in Y_{0:n})$ is done in three steps:

1. Assimilate $p(y_n \in Y_n|x_n)$ for both $t_n = t_{k_e}$ and $t_n = t_{k_a}$;
2. Calculate $p(x_n|y_{0:n} \in Y_{0:n})$ as a summation of N Gaussians;
3. Approximate $p(x_n|y_{0:n} \in Y_{0:n})$ as a single Gaussian function.

The last step ensures that $p(x_n|y_{0:n} \in Y_{0:n})$ is described by a finite set of Gaussians, which is crucial for attaining computational tractability. Notice that (13) gives a unified description of the hybrid state-estimator.

5.1 Step 1: Measurement Assimilation

This section gives a unified formula of the PDF $p(y_n \in Y_n|x_n)$ valid for both $t_n = t_{k_e}$ and $t_n = t_{k_a}$. From multivariate probability theory [20] and (7b) we have:

$$p(y_n \in Y_n|x_n) := \int_{-\infty}^{\infty} p(y_n|x_n)p(y_n \in Y_n)dy_n \quad \text{and} \quad p(y_n|x_n) = G(y_n, Cx_n, V). \tag{15}$$

The PDF $p(y_n \in Y_n)$ is modeled as a uniform distribution for all $y_n \in Y_n$. Therefore, depending on the type of instant, i.e. event or not, we have:

$$p(y_n \in Y_n) := \begin{cases} \Lambda_{H_{k_e}}(y_n) & \text{if } t_{k_e-1} < t_n < t_{k_e}, \\ \delta(y_n - y_{k_e}) & \text{if } t_n = t_{k_e}. \end{cases} \tag{16}$$

Substitution of (16) into (15) gives that $p(y_n \in Y_n) = G(y_{k_e}, Cx_n, V)$ if $t_n = t_{k_e}$. However, if $t_n = t_{k_a}$ then $p(y_n \in Y_n|x_n)$ equals $\Lambda_{H_{k_e}}(y_n)$, which is not necessarily Gaussian. Moreover, it depends on the set H_{k_e} and therefore on the actual event sampling method that is employed. In order to have a unified expression of $p(y_n \in Y_n|x_n)$ for both types of t_n, independent of the event sampling method, $\Lambda_{H_{k_e}}(y_n)$ can be approximated as a summation of N Gaussians, i.e.

$$\Lambda_{H_{k_e}}(y_n) \approx \sum_{i=1}^{N} \alpha_n^i G(y_n, y_n^i, V_n^i) \quad \text{and} \quad \sum_{i=1}^{N} \alpha_n^i := 1. \tag{17}$$

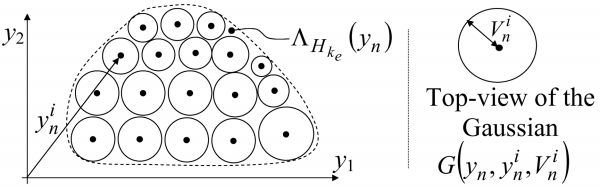


Fig. 4. Approximation of $\Lambda_{H_{k_e}}(y_n)$ as a sum of Gaussian functions

This is graphically depicted in Figure 4 for $y_n \in \mathbb{R}^2$. The interested reader is referred to [7] for more details.

Substituting (17) into (16) yields the following $p(y_n \in Y_n|x_n)$ if $t_n = t_{k_a}$:

$$p(y_n \in Y_n|x_n) \approx \sum_{i=1}^{N} \alpha_n^i \int_{-\infty}^{\infty} G(y_n, Cx_n, V)\, G(y_n, y_n^i, V_n^i) dy_n. \tag{18}$$

Proposition 1. [15,17] *Let there exist two Gaussians of random vectors $x \in \mathbb{R}^n$ and $m \in \mathbb{R}^q$, with $\Gamma \in \mathbb{R}^{q\times n}$: $G(m, \Gamma x, M)$ and $G(x, u, U)$. Then they satisfy:*

$$\int_{-\infty}^{\infty} G(x,u,U)\, G(m,\Gamma x, M) dx = G\left(\Gamma u, m, \Gamma U \Gamma^\top + M\right), \tag{19}$$

$$\begin{aligned} &G(x,u,U)\, G(m,\Gamma x, M) = G(x,d,D)\, G(m, \Gamma u, \Gamma U \Gamma^\top + M), \\ &\text{with} \quad D := \left(U^{-1} + \Gamma^\top M^{-1} \Gamma\right)^{-1} \quad \text{and} \quad d := DU^{-1}u + D\Gamma^\top M^{-1} m. \end{aligned} \tag{20}$$

Applying Proposition 1 ((19) to be precise) and $G(x,y,Z) = G(y,x,Z)$ on (18) yields:

$$p(y_n \in Y_n|x_n) \approx \sum_{i=1}^{N} \alpha_n^i G(y_n^i, Cx_n, V + V_n^i), \quad \text{if} \quad t_n = t_{k_a}. \tag{21}$$

In conclusion we can state that the unified expression of the PDF $p(y_n \in Y_n|x_n)$, at both $t_n = t_{k_e}$ and $t_n = t_{k_a}$, for any event sampling method results in:

$$p(y_n \in Y_n|x_n) \approx \sum_{i=1}^{N} \alpha_n^i G(y_n^i, Cx_n, R_n^i) \quad \text{with} \quad R_n^i := V + V_n^i. \tag{22}$$

If $t_n = t_{k_e}$ the variables of (22) are: $N = 1$, $\alpha_n^1 = 1$, $y_n^1 = y_{k_e}$ and $V_n^1 = \underline{0}$. If $t_n = t_{k_a}$ the variables depend on $\Lambda_{H_{k_e}}(y_n)$ and its approximation. As an example these variables are calculated for the method "Send-on-Delta" with $y \in \mathbb{R}$.

Example 1. In "Send-on-Delta", for certain N, the approximation of $\Lambda_{H_{k_e}}(y_n)$, as presented in (17), is obtained with $i \in \{1, 2, \ldots, N\}$ and:

$$\begin{aligned} &y_n^i = y_{k_e - 1} - \left(\frac{N - 2(i-1) - 1}{2N}\right) 2\Delta, \\ &\alpha_n^i = \frac{1}{N}, \quad V_n^i = \left(\frac{2\Delta}{N}\right)^2 \left(0.25 - 0.05 e^{-\frac{4(N-1)}{15}} - 0.08 e^{-\frac{4(N-1)}{180}}\right), \quad \forall i. \end{aligned} \tag{23}$$

With the result of (22), $p(x_n|y_{0:n} \in Y_{0:n})$ can also be expressed as a sum of N Gaussians.

5.2 Step 2: State Estimation

First the PDF $p(x_n|y_{0:n-1} \in Y_{0:n-1})$ of (14b) is calculated. From the EBSE we have $p(x_{n-1}|y_{0:n-1} \in Y_{0:n-1}) := G(x_{n-1}, x_{n-1|n-1}, P_{n-1,n-1})$ and from (7a) with $\tau_n := t_n - t_{n-1}$ we have $p(x_n|x_{n-1}) := G(x_n, A_{\tau_n} x_{n-1}, B_{\tau_n} Q B_{\tau_n}^\top)$. Therefore using (19) in (14b) yields:

$$\begin{aligned} &p(x_n|y_{0:n-1} \in Y_{0:n-1}) = G(x_n, x_{n|n-1}, P_{n,n-1}) \quad \text{with} \\ &x_{n|n-1} := A_{\tau_n} x_{n-1|n-1} \quad \text{and} \quad P_{n|n-1} := A_{\tau_n} P_{n-1|n-1} A_{\tau_n}^\top + B_{\tau_n} Q B_{\tau_n}^\top. \end{aligned} \tag{24}$$

Next $p(x_n|y_{0:n} \in Y_{0:n})$, defined in (13), is calculated after multiplying (22) and (24):

$$p(x_n|y_{n-1} \in Y_{0:n-1}) p(y_n \in Y_n|x_n) \approx \sum_{i=1}^{N} \alpha_n^i G(x_n, x_{n|n-1}, P_{n|n-1}) G(y_n^i, Cx_n, R_n^i). \tag{25}$$

Equation (25) is explicitly solved by applying Proposition 1:

$$p(x_n|y_{0:n-1} \in Y_{0:n-1}) p(y_n \in Y_n|x_n) \approx \sum_{i=1}^{N} \alpha_n^i \beta_n^i G(x_n, x_n^i, P_n^i) \quad \text{with} \tag{26a}$$

$$\begin{aligned} &x_n^i := P_n^i \left(P_{n|n-1}^{-1} x_{n|n-1} + C^\top \left(R_n^i\right)^{-1} y_n^i \right), \quad P_n^i := \left(P_{n|n-1}^{-1} + C^\top \left(R_n^i\right)^{-1} C \right)^{-1} \\ &\text{and} \quad \beta_n^i := G(y_n^i, Cx_{n|n-1}, CP_{n|n-1}C^\top + R_n^i). \end{aligned} \tag{26b}$$

The expression of $p(x_n|y_{0:n} \in Y_{0:n})$ as a sum of N Gaussians is the result of the following substitutions: (26) into (13), (26) into (14c) to obtain $p(y_n \in Y_n \mid y_{0:n-1} \in Y_{0:n-1})$ and the latter into (13) again. This yields

$$p(x_n|y_{0:n} \in Y_{0:n}) \approx \sum_{i=1}^{N} \frac{\alpha_n^i \beta_n^i}{\sum_{i=1}^{N} \alpha_n^i \beta_n^i} G(x_n, x_n^i, P_n^i). \tag{27}$$

The third step is to approximate (27) as a single Gaussian, as this facilitates a computationally tractable algorithm. For if $p(x_{n-1}|y_{0:n-1} \in Y_{0:n-1})$ is described using M_{n-1} Gaussians and $p(y_n \in Y_n|x_n)$ is described using N Gaussians, the estimate of x_n in (27) is described with $M_n = M_{n-1}N$ Gaussians. Meaning that M_n increases after each sample instant and with it also the processing demand of the EBSE increases.

5.3 Step 3: State Approximation

$p(x_n|y_{0:n} \in Y_{0:n})$ of (27) is approximated as a single Gaussian with an equal expectation and covariance matrix, i.e.:

$$p(x_n|y_{0:n} \in Y_{0:n}) \approx G\left(x_n, x_{n|n}, P_{n|n}\right) \quad \text{with} \tag{28a}$$

$$x_{n|n} := \sum_{i=1}^{N} \frac{\alpha_n^i \beta_n^i x_n^i}{\sum_{i=1}^{N} \alpha_n^i \beta_n^i}, \quad P_{n|n} := \sum_{i=1}^{N} \frac{\alpha_n^i \beta_n^i}{\sum_{i=1}^{N} \alpha_n^i \beta_n^i} \left(P_n^i + \left(x_{n|n} - x_n^i\right) \left(x_{n|n} - x_n^i\right)^\top \right). \tag{28b}$$

The expectation and covariance of (27), equal to $x_{n|n}$ and $P_{n|n}$ of (28), can be derived from the corresponding definitions. Notice that because the designed EBSE is based on the equations of the Kalman filter, the condition of computational tractability is met.

5.4 On Asymptotic Analysis of the Error-Covariance Matrix

In this section we present some preliminary results on the asymptotic analysis of the error-covariance matrix of the developed EBSE, i.e. $\lim_{n\to\infty} P_{n|n}$ which for convenience is denoted as P_∞. The main result of this section is obtained under the standing assumption that $\Lambda_{H_{k_e}}(y_n)$ is approximated using a single Gaussian. Note that the result then also applies to the estimator presented in [18], as a particular case. Recall that H_{k_e} is assumed to be a bounded set. Therefore, it is reasonable to further assume that $\Lambda_{H_{k_e}}(y_n)$ can be approximated using the formula (17), for $N = 1$, and that there exists a constant matrix R such that $V + V_n^1 \preceq R$ for all n.

Note that if the classical Kalman filter (KF) [15] is used to perform a state-update only at the synchronous time instant $t_n = t_{k_a}$ (with a measurement covariance matrix equal to R), then such an analysis is already available. In [21,22] it is proven that if the eigenvalues of A_{t_s} are within the unit circle and (A_{t_s}, C) is observable, then the error-covariance matrix of the synchronous KF, denoted with $P^{(s)}$, converges to P_K, with P_K defined as the solution of:

$$P_K = \left(\left(A_{t_s} P_K A_{t_s}^\top + B_{t_s} Q B_{t_s}^\top \right)^{-1} + C^\top R^{-1} C \right)^{-1}. \tag{29}$$

In case that the classical asynchronous Kalman filter (AKF) [16] is used, then the estimation would occur only at the instants that a measurement is received, i.e. $t_n = t_{k_e}$. As it is not known when a new measurement is available, the time between two samples keeps on growing, as well as the eigenvalues of the AKF's error-covariance matrix, denoted with $\lambda_i(P^{(a)})$. Moreover, in [23] (see also [24]) it is proven that $P^{(a)}$ will diverge if no new measurements are received.

To circumvent this problem, instead of a standard AKF, we consider an artificial AKF (denoted by CKF for brevity) obtained as the combination of a synchronous KF and a standard AKF. By this we mean that the CKF performs a state-update at all time instants t_n with a measurement covariance matrix equal to R. Therefore its error-covariance matrix, denoted with $P_{n|n}^{(c)}$, is updated according to:

$$P_{n|n}^{(c)} := \left(\left(A_{\tau_n} P_{n-1|n-1}^{(c)} A_{\tau_n}^\top + B_{\tau_n} Q B_{\tau_n}^\top \right)^{-1} + C^\top R^{-1} C \right)^{-1}. \tag{30}$$

Notice that because the CKF is updated at more time instants then the KF, it makes sense that its error-covariance matrix is "smaller" than the one of the KF, i.e. $P^{(c)} \preceq P^{(s)}$ holds at the synchronous time instants $t_n = t_{k_a}$. However, this does not state anything about $P^{(c)}$ at the event instants. As also at these sample instants the CKF performs an update rather then just a prediction, the following assumption is needed. Let $P_\infty^{(c)}$ denote $\lim_{n\to\infty} P_{n|n}^{(c)}$.

Assumption 1. *There exists* $\Delta_\lambda \in \mathbb{R}_+$ *such that* $\lambda_{max}\left(P_\infty^{(c)}\right) < \lambda_{max}(P_K) + \Delta_\lambda$.

Next we will employ Assumption 1 to obtain an upper bound on the error-covariance matrix of the developed EBSE. The following technical Lemma will be of use.

Lemma 1. *Let any square matrices $V_1 \preceq V_2$ and $W_1 \preceq W_2$ with $V_1 \succ 0$ and $W_1 \succ 0$ be given. Suppose that the matrices U_1 and U_2 are defined as $U_1 := \left(V_1^{-1} + C^\top W_1^{-1} C\right)^{-1}$ and $U_2 := \left(V_2^{-1} + C^\top W_2^{-1} C\right)^{-1}$, for any C of suitable size. Then it holds that $U_1 \preceq U_2$.*

Proof. As shown in [25], it holds that $V_1^{-1} \succeq V_2^{-1}$ and $C^\top W_1^{-1} C \succeq C^\top W_2^{-1} C$. Hence, it follows that $V_1^{-1} + C^\top W_1^{-1} C \succeq V_2^{-1} + C^\top W_2^{-1} C$, which yields $U_1^{-1} \succeq U_2^{-1}$. Thus, $U_1 \preceq U_2$, which concludes the proof. □

Theorem 1. *Suppose that the EBSE, as presented in Section 5, approximates $\Lambda_{H_{k_e}}(y_n)$ according to (17) with $N = 1$. Then $\lambda_{max}(P_\infty) \leq \lambda_{max}\left(P_\infty^{(c)}\right)$.*

The proof of the above theorem, which makes use of Lemma 1, is given in the Appendix. Obviously, under Assumption 1 the above result further implies that the error-covariance matrix of the developed EBSE is bounded. Under certain reasonable assumptions, including the standard ones (i.e. the eigenvalues of the A_{t_s}-matrix are within the unit-circle and (A_{t_s}, C) is an observable pair), it is possible to derive an explicit expression of Δ_λ, which validates Assumption 1. However, this is beyond the scope of this manuscript.

6 Illustrative Example

In this section we illustrate the effectiveness of the developed EBSE in terms of state-estimation error, sampling efficiency and computational tractability. The case study is a 1D object-tracking system. The states $x(t)$ of the object are position and speed while the measurement vector $y(t)$ is position. The process-noise $w(t)$ represents the object's acceleration. Then given a maximum acceleration of $0.5[m/s^2]$ its corresponding Q, according to [26], equals 0.02. Therefore the model as presented in (7) yields $A = \left(\begin{smallmatrix} 1 & \tau \\ 0 & 1 \end{smallmatrix}\right)$, $B = \left(\frac{\tau^2}{2} \;\; \tau\right)^\top$, $C = \left(1 \;\; 0\right)$ and $D = 0$, which is in fact a discrete-time double integrator. The acceleration, i.e. process noise $w(t)$, in time is shown in Figure 5 together with the object's position and speed, i.e. the elements of the real state-vector $x(t)$. The sampling time is $t_s = 0.1$ and the measurement-noise covariance is $V = 0.1 \cdot 10^{-3}$.

Three different estimators are tested. The first two estimators are the EBSE and the asynchronous Kalman filter (AKF) of [16]. For simplicity, in both estimators we used the "Send-on-Delta" method with $\Delta = 0.1[m]$. For the EBSE we approximated $\Lambda_{H_{k_e}}(y_n)$ using (23) with $N = 5$. The AKF estimates the states only at the event instants t_{k_e}. The states at t_{k_a} are calculated by applying the prediction-step of (14b). The third estimator

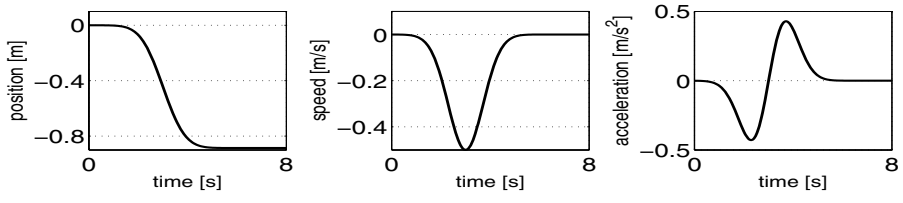
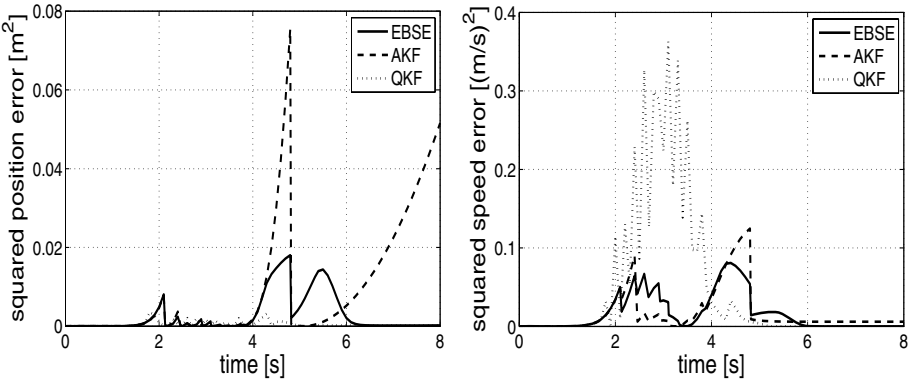


Fig. 5. The position, speed and acceleration of the object



(a) First state (b) Second state

Fig. 6. The squared estimation error of the two states

is based on the quantized Kalman filter (QKF) introduced in [26] that uses synchronous time sampling of y_{k_a}. The QKF can deal with quantized data, which also results in less data transfer, and therefore can be considered as an alternative to EBSE. In the QKF $\bar{y}_{k_a}$ is the quantized version of y_{k_a} with quantization level 0.1, which corresponds to the "Send-on-Delta" method. Hence, a comparison can be made.

In Figure 6(a) and Figure 6(b) the squared state estimation-error of the three estimators is plotted. They show that the QKF estimates the position of the object with the least error. However, its error in speed is worse compared to the EBSE. Further, the plot of the AKF clearly shows that prediction of the state gives a significant growth in estimation-error when the time between the event sampling-instants increases ($t > 4$).

Beside estimation error, sampling efficiency η is also important due to the increased interest in WSNs. For these systems communication is expensive and one aims to have the least data transfer. We define $\eta \in \mathbb{R}_+$ as

$$\eta := \frac{(x_i - x_{i|i})^\top (x_i - x_{i|i})}{(x_i - x_{i|i-1})^\top (x_i - x_{i|i-1})},$$

which is a measure of the change in the estimation-error after the measurement update with either z_{k_e} or $\bar{y}_{k_a}$ was done. Notice that if $\eta < 1$ the estimation error decreased after an update, if $\eta > 1$ the error increased and if $\eta = 1$ the error remained the same. For the EBSE $i = k_e$ with $i-1$ equal to $k_e - 1$ or $k_a - 1$. For the AKF $i = k_e$ with $i - 1 = k_e - 1$. For the QKF $i = k_a$ and $i - 1 = k_a - 1$. Figure 7 shows that for the EBSE $\eta < 1$ at all time instants. The AKF has one instant, $t = 3.4$, at which $\eta > 1$. In case of the QKF the error sometimes decreases but it can also increase considerably after an update. Also notice that η of the QKF converges to 1. Meaning that for $t > 5.6$ the estimation error does not change after an update and new samples are mostly used to bound $\lambda_i(P_{k_a|k_a})$. The EBSE has the same property, although for this method the last sample was received at $t = 4.9$.

The last comparison criterion is the total amount of processing time that was required by each of the three estimators. From the equations of the EBSE one can see that for

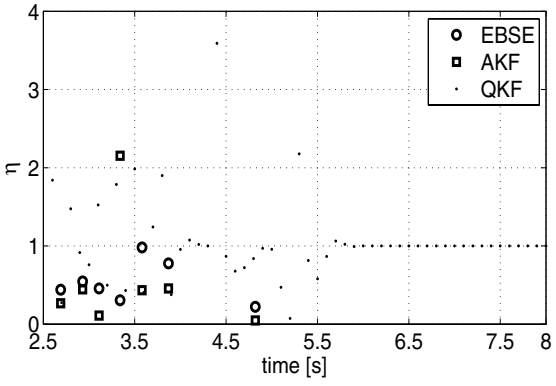


Fig. 7. The factor of increase in estimation error after z_{k_e} or $\bar{y}_{k_a}$

every Gaussian (recall that there are N Gaussians employed to obtain an approximation of $\Lambda_{H_{k_e}}(y_n)$) a state-update is calculated similar to a synchronous Kalman filter. Therefore, a rule of thumb is that the EBSE will require N times the amount of processing time of the Kalman filter [15]. Because the QKF is in fact such a Kalman filter, with a special measurement-estimation, the EBSE of this application example will roughly cost about 5 times more processing time then the QKF. After running all three algorithms in Matlab on an Intell®Pentium®processor of 1.86 GHz with 504 MB of RAM we have obtained the following performances. The AKF estimated x_{k_e} and predicted x_{k_a} in a total time of 0.016 seconds while the QKF estimated x_{k_a} and its total processing time equaled 0.022 seconds. For the EBSE, both x_{k_e} and x_{k_a} were estimated and it took 0.094 seconds, which is less than $0.11 = 5 \times 0.022$. This means that although the EBSE results in the most processing time, it is still computationally comparable to the AKF and QKF. On the overall, it can be concluded that the EBSE provides an estimation-error similar to the one attained by the QKF, but with significantly less data transmission. The application case study also indicate that the number of Gaussians becomes a tuning factor that can be used to achieve a desired tradeoff between numerical complexity (which further translates into energy consumption) and estimation error. As such, the proposed EBSE it is most suited for usage in networks in general and WSNs in particular.

7 Conclusions

In this paper a general event-based state-estimator was presented. The distinguishing feature of the proposed EBSE is that *estimation* of the states is performed at two different type of time instants, i.e. at event instants, when measurement data is used for update, and at synchronous time sampling, when no measurement is received, but an update is performed based on the knowledge that the monitored variable lies within a set used to define the event. As a result, under certain assumptions, it was established that the error-covariance matrix of the EBSE is bounded, even in the situation when no

new measurement is received anymore. Its effectiveness for usage in WSNs has been demonstrated on an application example.

As a final remark we want to indicate that future work, besides a more general proof of asymptotic stability, is focused on determining specific types of WSNs applications where the developed EBSE would be most suitable.

Acknowledgements. Research partially supported by the Veni grant "Flexible Lyapunov Functions for Real-time Control", grant number 10230, awarded by STW (Dutch Science Foundation) and NWO (The Netherlands Organization for Scientific Research).

References

1. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: Wireless Sensor Networks: a survey. Computer Networks 38, 393–422 (2002)
2. Tabuada, P.: Event-Triggered Real-Time Scheduling for Stabilizing Control Tasks. IEEE Transactions on Automatic Control 52, 1680–1685 (2007)
3. Velasco, M., Marti, P., Lozoya, C.: On the Timing of Discrete Events in Event-Driven Control Systems. In: Egerstedt, M., Mishra, B. (eds.) HSCC 2008. LNCS, vol. 4981, pp. 670–673. Springer, Heidelberg (2008)
4. Wang, X., Lemmon, M.: State based self-triggered feedback control systems with $\mathscr{L}_2$ stability. In: 17th IFAC World Congres, Seoul, South Korea (2008) (accepted in IEEE Transactions on Automatic Control)
5. Heemels, W.P.M.H., Sandee, J.H., van den Bosch, P.P.J.: Analysis of event-driven controllers for linear systems. International Journal of Control 81(4) (2008)
6. Henningsson, T., Johannesson, E., Cervin, A.: Sporadic event-based control of first-order linear stochastic systems. Automatica 44(11), 2890–2895 (2008)
7. Sorenson, H.W., Alspach, D.L.: Recursive Bayesian estimation using Gaussian sums. Automatica 7, 465–479 (1971)
8. Kotecha, J.H., Djurić, P.M.: Gaussian sum particle filtering. IEEE Transaction Signal Processing 51(10), 2602–2612 (2003)
9. Johnson, N.L., Kotz, S., Kemp, A.W.: Univariate discrete distributions. John Wiley and Sons, Chichester (1992)
10. Aggoun, L., Elliot, R.: Measure Theory and Filtering. Cambridge University Press, Cambridge (2004)
11. Lebesque, H.L.: Integrale, longueur, aire. PhD thesis, University of Nancy (1902)
12. Åström, K.J., Bernhardsson, B.M.: Comparison of Riemann and Lebesque sampling for first order stochastic systems. In: 41st IEEE Conf. on Dec. and Contr., Las Vegas, USA (2002)
13. Miskowicz, M.: Send-on-delta concept: an event-based data-reporting strategy. Sensors 6, 49–63 (2006)
14. Miskowicz, M.: Asymptotic Effectiveness of the Event-Based Sampling according to the Integral Criterion. Sensors 7, 16–37 (2007)
15. Kalman, R.E.: A new approach to linear filtering and prediction problems. Transaction of the ASME Journal of Basic Engineering 82(D), 35–42 (1960)
16. Mallick, M., Coraluppi, S., Carthel, C.: Advances in Asynchronous and Decentralized Estimation. In: Proceeding of the 2001 Aerospace Conference, Big Sky, MT, USA (2001)
17. Ristic, B., Arulampalam, S., Gordon, N.: Beyond the Kalman filter: Particle filter for tracking applications. Artech House, Boston (2004)
18. Nguyen, V.H., Suh, Y.S.: Improving estimation performance in Networked Control Systems applying the Send-on-delta transmission method. Sensors 7, 2128–2138 (2007)

19. Mardia, K.V., Kent, J.T., Bibby, J.M.: Mutlivariate analysis. Academic Press, London (1979)
20. Montgomery, D.C., Runger, G.C.: Applied Statistics and Probability for Engineers. John Wiley and Sons, Chichester (2007)
21. Hautus, M.L.J.: Controllability and observability conditions of linear autonomous systems. In: Indagationes Mathemathicae, vol. 32, pp. 448–455 (1972)
22. Balakrishnan, A.V.: Kalman Filtering Theory. Optimization Software, Inc., New York (1987)
23. Sinopoli, B., Schenato, L., Franceschetti, M., Poolla, K., Jordan, M., Sastry, S.: Kalman Filter with Intermittent Observations. IEEE Transactions on Automatic Control 49, 1453–1464 (2004)
24. Mo, Y., Sinopoli, B.: A characterization of the critical value for Kalman filtering with intermittent observations. In: 47th IEEE Conference on Decision and Control, Cancun, Mexico, pp. 2692–2697 (2008)
25. Bernstein, D.S.: Matrix Mathematics. Princeton University Press, Princeton (2005)
26. Curry, R.E.: Estimation and control with quantized measurements. MIT Press, Boston (1970)

A Proof of Theorem 1

Under the hypothesis, for the proposed EBSE, $P_{n|n}$ of (28), with $\tau_n := t_n - t_{n-1}$ and $R_n := V + V_n^1$, becomes:

$$P_{n|n} = \left(\left(A_{\tau_n} P_{n-1|n-1} A_{\tau_n}^\top + B_{\tau_n} Q B_{\tau_n}^\top \right)^{-1} + C^\top R_n^{-1} C \right)^{-1}. \tag{31}$$

The upper bound on $\lambda_{max}(P_\infty)$ is proven by induction, considering the asymptotic behavior of a CKF that runs in parallel with the EBSE, as follows. The EBSE calculates $P_{n|n}$ as (31) and the CKF calculates $P_{n|n}^{(c)}$ as (30). Note that this implies that $R_n \preceq R$ for all n. Let the EBSE and the CKF start with the same initial covariance matrix P_0.

The first step of induction is to prove that $P_{1|1} \preceq P_{1|1}^{(c)}$. From (31) and (30) we have that

$$P_{1|1} = \left(\left(A_{\tau_1} P_0 A_{\tau_1}^\top + B_{\tau_1} Q B_{\tau_1}^\top \right)^{-1} + C^\top R_1^{-1} C \right)^{-1},$$
$$P_{1|1}^{(c)} = \left(\left(A_{\tau_1} P_0 A_{\tau_1}^\top + B_{\tau_1} Q B_{\tau_1}^\top \right)^{-1} + C^\top R^{-1} C \right)^{-1}.$$

Suppose we define $V_1 := A_{\tau_1} P_0 A_{\tau_1}^\top + B_{\tau_1} Q B_{\tau_1}^\top$, $V_2 := A_{\tau_1} P_0 A_{\tau_1}^\top + B_{\tau_1} Q B_{\tau_1}^\top$, $W_1 := R_1$ and $W_2 := R$, then $W_1 \preceq W_2$ and $V_1 = V_2$. Therefore applying Lemma 1, with $U_1 := P_{1|1}$ and $U_2 := P_{1|1}^{(c)}$, yields $P_{1|1} \preceq P_{1|1}^{(c)}$.

The second and last step of induction is to show that if $P_{n-1|n-1} \preceq P_{n-1|n-1}^{(c)}$, then $P_{n|n} \preceq P_{n|n}^{(c)}$. Let $V_1 := A_{\tau_n} P_{n-1|n-1} A_{\tau_n}^\top + B_{\tau_n} Q B_{\tau_n}^\top$, $V_2 := A_{\tau_n} P_{n-1|n-1}^{(c)} A_{\tau_n}^\top + B_{\tau_n} Q B_{\tau_n}^\top$, $W_1 := R_n$ and $W_2 := R$. Notice that this gives $W_1 \preceq W_2$ and starting from $P_{n-1|n-1} \preceq P_{n-1|n-1}^{(c)}$ it follows that $V_1 \preceq V_2$ (see, e.g. [25]). Hence, applying Lemma 1, with $U_1 := P_{n|n}$ and $U_2 := P_{n|n}^{(c)}$ yields $P_{n|n} \preceq P_{n|n}^{(c)}$. This proves that $P_\infty \preceq P_\infty^{(c)}$, which yields (see e.g., [25]) $\lambda_{max}(P_\infty) \preceq \lambda_{max}\left(P_\infty^{(c)}\right)$. □

Discrete-State Abstractions of Nonlinear Systems Using Multi-resolution Quantizer

Yuichi Tazaki and Jun-ichi Imura

Tokyo Institute of Technology,
Ōokayama 2-12-1, Meguro, Tokyo, Japan
{tazaki,imura}@cyb.mei.titech.ac.jp
http://www.cyb.mei.titech.ac.jp

Abstract. This paper proposes a design method for discrete abstractions of nonlinear systems using multi-resolution quantizer, which is capable of handling state dependent approximation precision requirements. To this aim, we extend the notion of quantizer embedding, which has been proposed by the authors' previous works as a transformation from continuous-state systems to discrete-state systems, to a multi-resolution setting. Then, we propose a computational method that analyzes how a locally generated quantization error is propagated through the state space. Based on this method, we present an algorithm that generates a multi-resolution quantizer with a specified error precision by finite refinements. Discrete abstractions produced by the proposed method exhibit non-uniform distribution of discrete states and inputs.

1 Introduction

The problem of deriving a finite automaton that abstracts a given continuous-state system is called the discrete abstraction problem. A finite-state system is suitable for an abstract model since various difficult properties of continuous-state systems: nonlinearity, discontinuity, and non-convexity, ... can be handled in a uniform manner in a symbolic space. Until today, various techniques of discrete abstraction has been developed, and many of them are based on partitioning of state space. In [1] [2] [7], conditions for partitions that define discrete abstractions with deterministic transitions are discussed. On the other hand, in [3] [4], instead of considering discrete abstractions of the open-loop behavior of continuous systems, a hierarchical controller composed of a symbolic transition system and a feedback controller that moves the continuous state to one region to another is proposed. There have also been much attention paid on building a symbolic system whose behavior includes the behavior of a continuous system. Such symbolic abstractions have been used for verification problems in [6], and for supervisory control in [8] [9].

Recently, discrete abstraction methods based on approximate bisimulation [10] has gained growing attention. Approximate bisimulation is an extension of the original bisimulation to metric space. It admits equivalence relation between two systems if the distance of output signals can be kept within a given threshold.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 351–365, 2009.

Until now, it has been shown that discrete abstraction of a wide range of systems can be obtained based on approximate bisimulation. In [11], a procedure for constructing approximately bisimilar finite abstractions of stable discrete-time linear systems has been derived. In [12] and [13], it has been shown that a general nonlinear system with the so-called incremental stability property can be abstracted by a uniform grid. The authors have also investigated the application of approximate bisimilar discrete abstractions to optimal control of linear systems with non-convex state constraints in [14], and to interconnected systems in [15]. Discrete abstraction based on approximate bisimulation is especially suitable for control problems with a quantitative performance measure. It also has an advantage that it does not require expensive geometric computations. To date, however, it has the following limitations. First, the error condition in the conventional approximate bisimulation is uniform. From practical perspectives, it is desirable to support non-uniform error margin (for an example, error margin proportional to the norm of the signal itself). Second, the distribution of discrete states is also uniform. This means that the number of discrete states grows exponentially with respect to the dimension of the state space.

Motivated by the above backgrounds, this paper proposes a method for the design of discrete abstractions of nonlinear systems using multi-resolution quantizers. By using multi-resolution quantizers, one can design discrete abstract models with non-uniform distribution of states and inputs that approximate a given continuous-state system under state-dependent approximation precision requirements. Moreover, it also enables us to produce less conservative results compared to other conventional methods based on uniform discretization. To this end, in Section 2 we define the notion of finite-step abstraction. This notion admits an equivalence between two systems if any finite-step state trajectories of two systems generated with the same input signal satisfy a certain error criterion. In Section 3, we extend the notion of quantizer embedding, which has been presented in [14] and [15], to multi-resolution setting. This reduces the design of a discrete abstraction to the design of a pair of multi-resolution meshes, one is defined in the state space and another in the input space. In Section 4, we first discuss how to verify if a discrete model defined as a quantizer embedding with a given mesh satisfies the condition of the finite-step abstraction. This is basically done by computing how a locally generated quantization errors are propagated over the state space. Next, based on this verification method, we propose an algorithm that iteratively refines a multi-resolution mesh until the resultant quantizer-embedding is a finite-step abstraction of the original system. In Section 5, some illustrative examples are shown for demonstrating the effectiveness of the proposed method.

Notation: We write $[i_1 : i_2]$ to express the sequence of integers $i_1, i_1 + 1, \ldots, i_2$. For an integer sequence $I = [i_1 : i_2]$, $\boldsymbol{u}_I = \{\boldsymbol{u}_{i_1}, \boldsymbol{u}_{i_1+1}, \ldots, \boldsymbol{u}_{i_2}\}$. The symbol $\mathbb{R}$ denotes the field of real numbers and the symbol $\mathbb{Z}^+$ denotes the set of non-negative integers. For a vector $\boldsymbol{x} \in \mathbb{R}^n$, the symbol $\|\boldsymbol{x}\|_\infty$ denotes the ∞-norm of $\boldsymbol{x}$; $\|\boldsymbol{x}\|_\infty = \max_{i\in[1:n]} |\boldsymbol{x}_i|$.

2 Finite-Step Abstraction of Discrete-Time Systems

2.1 System Description

In this paper, we address the discrete abstraction of discrete-time continuous-state systems defined below. A discrete-time system is a tuple $\langle X, U, f\rangle$, where $X \subset \mathbb{R}^n$ is the set of states, $U \subset \mathbb{R}^m$ is the set of inputs, and $f : X \times U \mapsto X$ is the state transition function. The state and the input of the system at time $t \in \mathbb{Z}^+$ are expressed as $\boldsymbol{x}_t \in X$, $\boldsymbol{u}_t \in U$, respectively. The state transition at time t is expressed as

$$\boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{u}_t). \tag{1}$$

We use the symbol U^N to express the set of N-step admissible control input sequences.

2.2 Finite-Step Abstraction

In the following, we introduce the notion of finite-step abstraction for the class of systems defined above.

Definition 1. *Finite-step abstraction*
Let $\Sigma\langle X, U, f\rangle$ and $\hat{\Sigma}\langle X, U, \hat{f}\rangle$ be discrete-time dynamical systems. Further, let $\bar{R} \subset X \times X$ be a binary relation between X and X. The system $\hat{\Sigma}$ is an N-step abstraction of Σ with respect to $\bar{R}$ if and only if for any initial state $\boldsymbol{x}_0 \in X$ of Σ, there exists an initial state $\hat{\boldsymbol{x}}_0 \in X$ of $\hat{\Sigma}$ such that the following holds:
for any $\boldsymbol{u}_{[0:N-1]} \in U^N$,

$$\begin{aligned} &(\boldsymbol{x}_t, \hat{\boldsymbol{x}}_t) \in \bar{R} \ \ (t \in [0:N]), \\ &\boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{u}_t),\ \hat{\boldsymbol{x}}_{t+1} = \hat{f}(\hat{\boldsymbol{x}}_t, \boldsymbol{u}_t) \ \ (t \in [0:N-1]) \end{aligned} \tag{2}$$

holds.

The notion of finite-step abstraction can be seen as a variant of approximate bisimulation in the sense that:

i) it requires the similarity of trajectories for only a finite steps,
ii) it assumes common control inputs, whereas for approximate bisimulation two control inputs need not be the same, and
iii) the error condition is given in a more general form of binary relation $\bar{R}$, compared to the constant error bound of conventional approximate bisimulation.

The finite-step formulation could be a restriction. Nevertheless, it still has a wide range of potential applications. It can, of course, be used to solve problems that take place in a finite time interval, such as finite horizon optimal control. Moreover, it can be combined with model predictive control techniques to form a feedback-type controller. The relation $\bar{R}$ encodes an error condition imposed to two systems. The most simple example of $\bar{R}$ is a uniform error condition: $\bar{R} = \{(\boldsymbol{x}, \hat{\boldsymbol{x}}) \,|\, \|\boldsymbol{x} - \hat{\boldsymbol{x}}\| \leq \epsilon\}$, where ϵ is a positive constant. On the other hand, $\bar{R}$ defined as $\bar{R} = \{(\boldsymbol{x}, \hat{\boldsymbol{x}}) \,|\, \|\boldsymbol{x} - \hat{\boldsymbol{x}}\| \leq \eta\|\hat{\boldsymbol{x}}\| + \epsilon\}$, where both ϵ and η are positive constants, expresses a relative error condition.

3 Quantizer Embedding

In this section, we introduce the notion of *quantizer embedding*, which has been recently proposed by the authors in [15].

First of all, we introduce a *mesh* defined over a set $X \in \mathbb{R}^n$. A mesh $\mathcal{M}$ is a finite collection of pairs denoted by

$$\mathcal{M} = \{\{\boldsymbol{\xi}_0, C_0\}, \{\boldsymbol{\xi}_1, C_1\}, \ldots, \{\boldsymbol{\xi}_{\mathcal{S}}, C_{\mathcal{S}}\}\}. \tag{3}$$

Here, $\{C_0, C_1, \ldots, C_{\mathcal{S}}\}$ forms a partition of X; that is, $C_i \cap C_j = \emptyset \quad (i \neq j)$, $\bigcup_i C_i = X$. Each $C_s\,(s \in [0 : \mathcal{S}])$ is called a *cell* of the mesh. Moreover, each $\boldsymbol{\xi}_s \in C_s$ is called a *discrete point* of the s-th cell. A mesh $\mathcal{M}$ defines a quantization function as shown below.

$$\begin{aligned} &Q[\mathcal{M}] \,:\, X \mapsto \{\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_{\mathcal{S}}\}, \\ &Q[\mathcal{M}](\boldsymbol{x}) = \boldsymbol{\xi}_s \;\text{ if }\; \boldsymbol{x} \in C_s. \end{aligned} \tag{4}$$

A quantization function $Q[\mathcal{M}](\cdot)$ maps an arbitrary point $\boldsymbol{x}$ to a discrete point whose corresponding cell includes $\boldsymbol{x}$. We write $Q[\mathcal{M}](X) = \{\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_{\mathcal{S}}\}$. Moreover, for any $\boldsymbol{x} \in X$, we write $Q[\mathcal{M}]^{-1}(\boldsymbol{x}) = C_s$ iff $Q[\mathcal{M}](\boldsymbol{x}) = \boldsymbol{\xi}_s$.

A discrete-time system can be transformed into a finite state system by embedding a pair of quantizers into its state-transition function.

Definition 2. *Quantizer embedding of discrete-time systems*
Let $\Sigma\langle X, U, f\rangle$ be a discrete-time system. Moreover let $Q^x(\cdot) := Q[\mathcal{M}^x](\cdot)$ be a quantizer defined in the state space and let $Q^u(\cdot) := Q[\mathcal{M}^u](\cdot)$ be a quantizer defined in the input space. The quantizer embedding (QE in short) of Σ, denoted by $\mathrm{QE}(\Sigma, Q^x, Q^u)$, is a system $\hat{\Sigma}\langle Q^x(X), U, \hat{f}\rangle$ whose state transition function is defined as

$$\hat{f}(\boldsymbol{x}, \boldsymbol{u}) := Q^x(f(\boldsymbol{x}, Q^u(\boldsymbol{u}))). \tag{5}$$

At every transition, the input of $\hat{\Sigma}$ is mapped to the discrete point of a cell of the input mesh $\mathcal{M}^u$ in which it is included. Moreover, the state of $\hat{\Sigma}$ is reset to the discrete point of a cell of the state mesh $\mathcal{M}^x$ in which the state right after a transition made by f is included. Therefore, as long as the meshes $\mathcal{M}^x$ and $\mathcal{M}^u$ are composed of a finite number of cells, a system with a state-transition of the form (5) can be viewed as a finite automaton. Once we assume that discrete models are expressed in terms of the quantizer embedding of the original system, the problem of discrete abstraction reduces to the design of a state mesh and an input mesh. From now on, to distinguish the cells and discrete points of the state mesh and the input mesh, we denote them by $\mathcal{M}^x = \{\{\boldsymbol{\xi}^x_s, C^x_s\}\}_{[0:\mathcal{S}]}$ and $\mathcal{M}^u = \{\{\boldsymbol{\xi}^u_a, C^u_a\}\}_{[0:\mathcal{A}]}$, respectively. The transition of $\hat{\Sigma}$ can be rewritten in a symbolic form as

$$s \xrightarrow{a} s' \;\;\Leftrightarrow\;\; f(\boldsymbol{\xi}^x_s, \boldsymbol{\xi}^u_a) \in C^x_{s'}. \tag{6}$$

Moreover, for later use, we define the *predecessor set* of a symbolic state s' as $\mathrm{pre}(s') = \{\{s, a\} \mid s \xrightarrow{a} s'\}$. Based on the above discussion, we formulate discrete abstraction as the following problem.

Problem 1. Discrete abstraction
For a discrete-time system $\Sigma\langle X, U, f\rangle$, $N \in \mathbb{Z}^+$ *and a binary relation* $\bar{R} \subset X \times X$, *find a mesh* $\mathcal{M}^x$ *and* $\mathcal{M}^u$ *such that the quantizer-embedding* $\mathrm{QE}(\Sigma, Q[\mathcal{M}^x], Q[\mathcal{M}^u])$ *is an* N*-step abstraction of* Σ *with respect to* $\bar{R}$.

4 Design of Multi-resolution Quantizer

4.1 Preparations

Throughout this section, we will make frequent use of interval operations. Interval computation technique has been used in reachability analysis problems (see [16]). It will be shown that this technique can also be utilized for the verification of discrete abstraction. Let $[x] = [\underline{x}, \overline{x}]$ and $[y] = [\underline{y}, \overline{y}]$ be closed intervals in $\mathbb{R}$. Elementary operations are defined as follows:

$$\begin{aligned}
[x] + [y] &= [\underline{x} + \underline{y},\ \overline{x} + \overline{y}],\\
[x] - [y] &= [\underline{x} - \overline{y},\ \overline{x} - \underline{y}],\\
[x][y] &= [\min\{\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y}\},\ \max\{\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y}\}],\\
[x] \cup [y] &= [\min\{\underline{x}, \underline{y}\},\ \max\{\overline{x}, \overline{y}\}].
\end{aligned}$$

Moreover, $[x] \subseteq [y] \ \Leftrightarrow\ \underline{x} \geq \underline{y}, \overline{x} \leq \overline{y}$. Interval vectors and interval matrices are vectors and matrices whose elements are intervals. They obey the arithmetic rules of conventional matrices and vectors, except that element-wise operations are interval operations defined as above. We denote the set of all interval vectors of length n by $\mathbb{IR}^n$, and the set of all interval matrices with n rows and m columns by $\mathbb{IR}^{n\times m}$. The i-th element of an interval vector $[\boldsymbol{x}]$ is denoted by $[x_i] = [\underline{x}_i, \overline{x}_i]$. The element that lies in the i-th row and the j-th column of an interval matrix $[A]$ is denoted by $[A_{ij}] = [\underline{A}_{ij}, \overline{A}_{ij}]$. Let $[\boldsymbol{x}], [\boldsymbol{y}] \in \mathbb{IR}^n$ and let $\boldsymbol{a} \in \mathbb{R}^n$. We write

$$\begin{aligned}
\boldsymbol{a} \in [\boldsymbol{x}] \ &\Leftrightarrow\ a_i \in [x_i]\ \ \forall i \in [1:n],\\
[\boldsymbol{x}] \subseteq [\boldsymbol{y}] \ &\Leftrightarrow\ [x_i] \subseteq [y_i]\ \ \forall i \in [1:n].
\end{aligned}$$

Moreover, for compact sets $C \subset \mathbb{R}^n$ and $D \subset \mathbb{R}^n$, we define

$$(C, D) = \max_{\boldsymbol{a}\in C, \boldsymbol{b}\in D} \boldsymbol{a}^{\mathrm{T}}\boldsymbol{b}.$$

For interval vectors, we have

$$([\boldsymbol{x}], [\boldsymbol{y}]) = \sum_{i\in[1:n]} \max_{a_i\in[x_i], b_i\in[y_i]} a_i b_i = \sum_{i\in[1:n]} \max\{\underline{x}_i\underline{y}_i, \underline{x}_i\overline{y}_i, \overline{x}_i\underline{y}_i, \overline{x}_i\overline{y}_i\}.$$

Some useful properties of the above operations are listed below for later use.

$$\begin{aligned}
&([\boldsymbol{x}] + [\boldsymbol{y}], \boldsymbol{v}) = ([\boldsymbol{x}], \boldsymbol{v}) + ([\boldsymbol{y}], \boldsymbol{v}) && ([\boldsymbol{x}], [\boldsymbol{y}] \in \mathbb{IR}^n, \boldsymbol{v} \in \mathbb{R}^n),\\
&([\boldsymbol{x}] \cup [\boldsymbol{y}], [\boldsymbol{z}]) = \max\{([\boldsymbol{x}], [\boldsymbol{z}]), ([\boldsymbol{y}], [\boldsymbol{z}])\} && ([\boldsymbol{x}], [\boldsymbol{y}], [\boldsymbol{z}] \in \mathbb{IR}^n),\\
&([A][\boldsymbol{x}], \boldsymbol{e}_i) = ([\boldsymbol{x}], [A]^{\mathrm{T}}\boldsymbol{e}_i) && ([\boldsymbol{x}] \in \mathbb{IR}^n, [A] \in \mathbb{IR}^{n\times n}).
\end{aligned}$$

Here, $\boldsymbol{e}_i$ is a vector whose i-th element is 1 and others are 0.

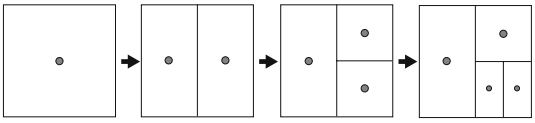


Fig. 1. Multi-resolution mesh

4.2 Multi-resolution Quantizer

In this subsection, we introduce a class of quantizer defined by a multi-resolution mesh. First of all, we assume that the domain X of a quantizer is expressed as an interval vector of length n. A multi-resolution mesh takes the form of a binary tree, and each of its leaf nodes is assigned a cell of the mesh. Each cell is an interval vector of length n. Moreover, we assume that each discrete point is placed in the middle of the corresponding cell. Initially, the tree is composed of a single root node, whose cell represents the entire state set X. The tree can be grown by choosing an arbitrary leaf node, subdividing the corresponding cell into two sub-cells, and assigning each of them to one of two new child nodes that are created below the chosen node. A subdivision can be made in one of the n directions. For an example, in the 2-dimensional case, a cell can be divided either horizontally or vertically. Let $[C]_s$ be a cell expressed as $[C]_s = [[\underline{C}_{s,1}, \overline{C}_{s,1}], [\underline{C}_{s,2}, \overline{C}_{s,2}], \ldots, [\underline{C}_{s,n}, \overline{C}_{s,n}]]^{\mathrm{T}}$ and let $\boldsymbol{\xi}_s$ be the discrete point of $[C]_s$. Here, the i-th element of $\boldsymbol{\xi}_s$ is given by $\boldsymbol{\xi}_{s,i} = (\underline{C}_{s,i} + \overline{C}_{s,i})/2$. Subdividing $[C]_s$ in the i-th direction yields two new sub-cells: $[C]_{s_1} = [[\underline{C}_{s,1}, \overline{C}_{s,1}], \ldots, [\underline{C}_{s,i}, (\underline{C}_{s,i} + \overline{C}_{s,i})/2], \ldots, [\underline{C}_{s,n}, \overline{C}_{s,n}]]^{\mathrm{T}}$ and $[C]_{s_2} = [[\underline{C}_{s,1}, \overline{C}_{s,1}], \ldots, [(\underline{C}_{s,i} + \overline{C}_{s,i})/2, \overline{C}_{s,i}], \ldots, [\underline{C}_{s,n}, \overline{C}_{s,n}]]^{\mathrm{T}}$. The discrete points of the new cells are placed in the middle of them. Finally, since $[C]_s$ is no longer a leaf node after the subdivision, its cell and discrete point are removed from the mesh. Fig. 1 illustrates how a multi-resolution mesh is refined.

4.3 Verification of N-Step Abstraction

In this subsection, we discuss how to verify, for a given discrete-time system $\Sigma\langle X, U, f\rangle$ and a given pair of multi-resolution meshes $\mathcal{M}^x$ and $\mathcal{M}^u$, whether $\hat{\Sigma}$ defined as $\hat{\Sigma} = \mathrm{QE}(\Sigma, Q^x, Q^u)$ ($Q^x = Q[\mathcal{M}^x]$, $Q^u = Q[\mathcal{M}^u]$) is an N-step abstraction of Σ. From later on, we will write $\mathcal{M}^x = \{\{\boldsymbol{\xi}^x_s, [C]^x_s\}\}_{[0:\mathcal{S}]}$, $\mathcal{M}^u = \{\{\boldsymbol{\xi}^u_a, [C]^u_a\}\}_{[0:\mathcal{A}]}$, since the cells of the meshes are assumed to be interval vectors. Let us define the following sequence of binary relations:

$$R_0 = \bigcup_{\boldsymbol{x}\in X} (\boldsymbol{x},\, Q^x(\boldsymbol{x})), \quad R_t = \bigcup_{(\boldsymbol{x},\hat{\boldsymbol{x}})\in R_{t-1},\, \boldsymbol{u}\in U} (f(\boldsymbol{x},\boldsymbol{u}), \hat{f}(\hat{\boldsymbol{x}},\, \boldsymbol{u})). \tag{7}$$

We will transform this recursive expression of R_t in such a way that the discrete-state characteristics of $\hat{\Sigma}$ is made more explicit. Let us define $R_{t,s} := \{\boldsymbol{x} \,|\, (\boldsymbol{x}, \boldsymbol{\xi}^x_s) \in R_t\}$. Then (7) is rewritten using $R_{t,s}$ as follows:

$$R_{0,s} = [C]^x_s, \quad R_{t,s'} = \bigcup_{\{s,a\}\in \mathrm{pre}(s')} f(R_{t-1,s}, [C]^u_a). \tag{8}$$

Here, $f(R_{t-1,s}, [C]^u_a) = \{f(\boldsymbol{x}, \boldsymbol{u}) \,|\, \boldsymbol{x} \in R_{t-1,s}, \boldsymbol{u} \in [C]^u_a\}$. The following lemma provides the simplest way for checking the N-step abstraction.

Lemma 1. *$\hat{\Sigma}$ is an N-step abstraction of Σ if $R_t \subseteq \bar{R}$ holds for all $t \in [0:N]$, or equivalently, if $R_{t,s} \subseteq \bar{R}_s$ holds for all $t \in [0:N]$ and $s \in [0:\mathcal{S}]$, where $\bar{R}_s = \{\boldsymbol{x} \,|\, (\boldsymbol{x}, \boldsymbol{\xi}^x_s) \in \bar{R}\}$.*

It is in general difficult to compute the nonlinear set operation $f(R_{t-1,s}, [C]^u_a)$. Moreover, due to the set union operations, $R_{t,s}$ may become highly non-convex as t increases. In the following, we derive a practical method that computes a conservative approximation of (8), taking advantage of interval computation techniques. The next lemma provides us with a conservative linear approximation of the nonlinear set operation $f(R_{t-1,s}, [C]^u_a)$.

Lemma 2. *Let C^x and C^u be convex subsets of X and U, respectively, and let $\boldsymbol{\xi}^x \in C^x$, $\boldsymbol{\xi}^u \in C^u$. Moreover, let $[A](C^x, C^u)$ and $[B](C^x, C^u)$ be functions that return interval matrices defined as*

$$[A_{ij}](C^x, C^u) = \left[\min_{\boldsymbol{x}\in C^x, \boldsymbol{u}\in C^u} \frac{\partial f_i}{\partial x_j}(\boldsymbol{x}, \boldsymbol{u}), \max_{\boldsymbol{x}\in C^x, \boldsymbol{u}\in C^u} \frac{\partial f_i}{\partial x_j}(\boldsymbol{x}, \boldsymbol{u})\right], \tag{9}$$

$$[B_{ij}](C^x, C^u) = \left[\min_{\boldsymbol{x}\in C^x, \boldsymbol{u}\in C^u} \frac{\partial f_i}{\partial u_j}(\boldsymbol{x}, \boldsymbol{u}), \max_{\boldsymbol{x}\in C^x, \boldsymbol{u}\in C^u} \frac{\partial f_i}{\partial u_j}(\boldsymbol{x}, \boldsymbol{u})\right]. \tag{10}$$

Then, the following inclusion holds:

$$f(C^x, C^u) \subseteq [A](C^x, C^u)(C^x - \boldsymbol{\xi}^x) + [B](C^x, C^u)(C^u - \boldsymbol{\xi}^u) + f(\boldsymbol{\xi}^x, \boldsymbol{\xi}^u). \tag{11}$$

Proof. From the mean value theorem, for any $\boldsymbol{x} \in C^x$, $\boldsymbol{u} \in C^u$ and $i \in [1:n]$, there exists a $\theta \in [0,1]$ such that

$$f_i(\boldsymbol{x}, \boldsymbol{u}) = f_i(\boldsymbol{\xi}^x, \boldsymbol{\xi}^u) + \frac{\partial}{\partial \boldsymbol{x}} f_i(\boldsymbol{x}', \boldsymbol{u}')(\boldsymbol{x} - \boldsymbol{\xi}^x) + \frac{\partial}{\partial \boldsymbol{u}} f_i(\boldsymbol{x}', \boldsymbol{u}')(\boldsymbol{u} - \boldsymbol{\xi}^u)$$

where $f_i(\boldsymbol{x}, \boldsymbol{u})$ denotes the i-th element of $f(\boldsymbol{x}, \boldsymbol{u})$, $\boldsymbol{x}' = \boldsymbol{\xi}^x + \theta(\boldsymbol{x} - \boldsymbol{\xi}^x)$ and $\boldsymbol{u}' = \boldsymbol{\xi}^u + \theta(\boldsymbol{u} - \boldsymbol{\xi}^u)$. From the convexity assumption, $\boldsymbol{x}' \in C^x$, $\boldsymbol{u}' \in C^u$. This means $(\partial/\partial \boldsymbol{x}) f_i(\boldsymbol{x}', \boldsymbol{u}')$ is included in the i-th row interval vector of $[A](C^x, C^u)$ and $(\partial/\partial \boldsymbol{u}) f_i(\boldsymbol{x}', \boldsymbol{u}')$ is included in the i-th row interval vector of $[B](C^x, C^u)$. This completes the proof. □

Now, let $[\mathcal{E}]_{t,s}$ be a sequence of interval vectors defined as

$$[\mathcal{E}]_{0,s} = [C]^x_s - \boldsymbol{\xi}^x_s, \tag{12}$$

$$[\mathcal{E}]_{t,s'} = \bigcup_{\{s,a\}\in \mathrm{pre}(s')} [[A]_{t-1,s,a}[\mathcal{E}]_{t-1,s} + [B]_{t-1,s,a}([C]^u_a - \boldsymbol{\xi}^u_a) + (f(\boldsymbol{\xi}^x_s, \boldsymbol{\xi}^u_a) - \boldsymbol{\xi}^x_{s'})] \tag{13}$$

where $[A]_{t,s,a} = [A]([\mathcal{E}]_{t,s} + \boldsymbol{\xi}_s^x, [C]_a^u)$ and $[B]_{t,s,a} = [B]([\mathcal{E}]_{t,s} + \boldsymbol{\xi}_s^x, [C]_a^u)$. From Lemma 2, we have $[\mathcal{E}]_{t,s} + \boldsymbol{\xi}_s^x \supseteq R_{t,s}$; thus, $[\mathcal{E}]_{t,s} + \boldsymbol{\xi}_s^x$ is an over-approximation of $R_{t,s}$. This means that $[\mathcal{E}]_{t,s}$ can be seen as a conservative estimate of the *accumulated error* between $\boldsymbol{x}_t$ and $\hat{\boldsymbol{x}}_t$ when $\hat{\boldsymbol{x}}_t = \boldsymbol{\xi}_s^x$ ($\hat{\boldsymbol{x}}_t$ denotes the state of $\hat{\Sigma}$ at time t). Furthermore, equation (13) describes how the accumulated error of one symbolic state is propagated to other symbolic states. The term $[A]_{t-1,s,a}[\mathcal{E}]_{t-1,s}$ expresses the error of s being propagated to its successor s', the term $[B]_{t-1,s,a}([C]_a^u - \boldsymbol{\xi}_a^u)$ expresses the input quantization error, and the term $(f(\boldsymbol{\xi}_s^x, \boldsymbol{\xi}_a^u) - \boldsymbol{\xi}_{s'}^x)$ expresses the state quantization error. For later use, we write equation (12), (13) in the form of an algorithm.

propagate_error
for each $s \in \mathcal{S}$ do $[\mathcal{E}]_{0,s} := [C]_s^x - \boldsymbol{\xi}_s^x$
for $t = 1$ to N
for each $s' \in \mathcal{S}$
$[\mathcal{E}]_{t,s'} := \bigcup_{\{s,a\} \in \mathrm{pre}(s')} [[A]_{t-1,s,a}[\mathcal{E}]_{t-1,s} + [B]_{t-1,s,a}([C]_a^u - \boldsymbol{\xi}_a^u) + (f(\boldsymbol{\xi}_s^x, \boldsymbol{\xi}_a^u) - \boldsymbol{\xi}_{s'}^x)]$
end
end

The next lemma gives a sufficient condition for the N-step abstraction.

Lemma 3. *$\hat{\Sigma}$ is an N-step abstraction of Σ if*

$$[\mathcal{E}]_{t,s} \subseteq \bar{\mathcal{E}}_s \tag{14}$$

holds for all $t \in [0:N]$, $s \in [0:\mathcal{S}]$, where $\bar{\mathcal{E}}_s = \bar{R}_s - \boldsymbol{\xi}_s^x$.

From later on, we focus on cases in which $\bar{R}_s$ is expressed as an interval vector. In such cases, the evaluation of $[\mathcal{E}]_{t,s} \subseteq \bar{\mathcal{E}}_s$ is done by comparing real values at most $2n$ times. The following algorithm checks the N-step abstraction.

max_violation
$\{t, s, \mu, i\} := \mathrm{argmax}_{t \in [0:N], s \in [0:\mathcal{S}], \mu \in \{-1,1\}, i \in [1:n]}(([\mathcal{E}]_{t,s}, \mu \boldsymbol{e}_i) - (\bar{\mathcal{E}}_s, \mu \boldsymbol{e}_i))$
$p := ([\mathcal{E}]_{t,s}, \mu \boldsymbol{e}_i) - (\bar{\mathcal{E}}_s, \mu \boldsymbol{e}_i)$
return $\{t, s, \mu, i, p\}$

The **max_violation** algorithm returns a tuple $\{t, s, \mu, i, p\}$, in which t, s and $\mu \boldsymbol{e}_i$ denote the time, the cell index and the direction of the maximum error violation, respectively, and p denotes the corresponding amount of violation. If $p \leq 0$, it means that condition (14) is satisfied and therefore no further refinement of the meshes is needed. In the next subsection, we discuss how to refine the meshes if p returned by **max_violation** has a positive value.

4.4 Iterative Refinement of Meshes

Let us consider how to design a pair of meshes that satisfies (14). We want the meshes not only to satisfy (14), but also to be as coarse as possible. However, it is extremely difficult to guarantee that the obtained meshes are optimal in

terms of the number of cells. Instead, we take an iterative and greedy approach; starting from a state mesh and an input mesh both composed of a single cell, we subdivide a cell in a certain direction one by one until the error condition (14) is satisfied. At each iteration, we first detect a cell and a direction that violate the error margin most significantly. This is done by calling **propagate_error** followed by **max_violation**, defined in the last subsection. Let us denote the return values of **max_violation** by $\{t, s', \mu, i, p\}$. Again, if $p \leq 0$ then no further refinement is needed and we can get out of the loop. Otherwise, we identify a cell-direction pair whose contribution to the error $([\mathcal{E}]_{t,s'}, \mu \boldsymbol{e}_i)$ is the largest. From (13), $([\mathcal{E}]_{t,s'}, \mu \boldsymbol{e}_i)$ is given by the following:

$$
\begin{aligned}
&([\mathcal{E}]_{t,s'}, \mu \boldsymbol{e}_i) = \\
&\begin{cases}
([C]^x_{s'} - \boldsymbol{\xi}^x_{s'}, \mu \boldsymbol{e}_i) & \text{if } t = 0, \\
\max\limits_{\{s,a\} \in \mathrm{pre}(s')} \big(([\mathcal{E}]_{t-1,s}, \mu [A]^{\mathrm{T}}_{t-1,s,a} \boldsymbol{e}_i) + ([C]^u_a - \boldsymbol{\xi}^u_a, \mu [B]^{\mathrm{T}}_{t-1,s,a} \boldsymbol{e}_i) & \\
\qquad + (f(\boldsymbol{\xi}^x_s, \boldsymbol{\xi}^u_a) - \boldsymbol{\xi}^x_{s'}, \mu \boldsymbol{e}_i) \big) & \text{otherwise.}
\end{cases}
\end{aligned}
\tag{15}
$$

For later use, let us denote by $\mathrm{pre}(t, s', \mu, i)$ the pair $\{s, a\}$ that gives the maximum in the right hand side of (15). We can observe from (15) that for $t = 0$, the only way to reduce $([\mathcal{E}]_{t,s'}, \mu \boldsymbol{e}_i)$ is to reduce $([C]^x_{s'} - \boldsymbol{\xi}^x_{s'}, \boldsymbol{e}_i)$; i.e., to subdivide the cell $[C]^x_{s'}$ in the i-th direction. On the other hand, for $t > 0$, $([\mathcal{E}]_{t,s'}, \mu \boldsymbol{e}_i)$ is given by the maximum of the sum of three terms, where the maximum is taken among all the predecessors of s'. Note that those three terms are interpreted as: the accumulated error propagated from s, the input quantization error, and the state quantization error. Therefore, we have three options to reduce $([\mathcal{E}]_{t,s'}, \mu \boldsymbol{e}_i)$: to reduce the accumulated error $([\mathcal{E}]_{t-1,s}, \mu [A]^{\mathrm{T}}_{t-1,s,a} \boldsymbol{e}_i)$, to reduce the input quantization error $([C]^u_a - \boldsymbol{\xi}^u_a, \mu [B]^{\mathrm{T}}_{t-1,s,a} \boldsymbol{e}_i)$, and to reduce the state quantization error $(f(\boldsymbol{\xi}^x_s, \boldsymbol{\xi}^u_a) - \boldsymbol{\xi}^x_{s'}, \mu \boldsymbol{e}_i)$.

To reduce $([C]^u_a - \boldsymbol{\xi}^u_a, \mu [B]^{\mathrm{T}}_{t-1,s,a} \boldsymbol{e}_i)$, we should subdivide $[C]^u_a$. However, a question arises; in which direction should $[C]^u_a$ be subdivided? Now, notice that $([C]^u_a - \boldsymbol{\xi}^u_a, \mu [B]^{\mathrm{T}}_{t-1,s,a} \boldsymbol{e}_i)$ can be further decomposed into the following form:

$$
\begin{aligned}
([C]^u_a - \boldsymbol{\xi}^u_a, \mu [B]^{\mathrm{T}}_{t-1,s,a} \boldsymbol{e}_i) &= ([C], \mu [B^{\mathrm{row}}_i]^{\mathrm{T}}) = \sum_{j \in [1:m]} ([C_j], \mu [B_{ij}]) \\
&= \sum_{j \in [1:m]} \max\{([C_j], \mu \underline{B}_{ij}), ([C_j], \mu \overline{B}_{ij})\} \\
&= \sum_{j \in [1:m]} \max\{([C], \mu \underline{B}_{ij} \boldsymbol{e}_j), ([C], \mu \overline{B}_{ij} \boldsymbol{e}_j)\}.
\end{aligned}
\tag{16}
$$

For ease of notation, we temporarily write $[C] = [C]^u_a - \boldsymbol{\xi}^u_a$. Moreover, $[B^{\mathrm{row}}_i]$ and $[B_{ij}] = [\underline{B}_{ij}, \overline{B}_{ij}]$ denote the i-th row vector and the (i, j)-element of $[B]_{t-1,s,a}$, respectively. From this equation, we observe that by subdividing $[C]^u_a$ in the j-th direction whose corresponding term in the summation is the largest, the overall input quantization error will be reduced the most. Thus, we choose this j as the direction of subdivision.

For reducing $(f(\boldsymbol{\xi}_s^x, \boldsymbol{\xi}_a^u) - \boldsymbol{\xi}_{s'}^x, \mu \boldsymbol{e}_i)$, it is natural to subdivide $[C]_{s'}^x$ However, we need an additional care; subdividing $[C]_{s'}^x$ in the i-th direction does not always reduce $(f(\boldsymbol{\xi}_s^x, \boldsymbol{\xi}_a^u) - \boldsymbol{\xi}_{s'}^x, \mu \boldsymbol{e}_i)$. If $(f(\boldsymbol{\xi}_s^x, \boldsymbol{\xi}_a^u) - \boldsymbol{\xi}_{s'}^x)^{\mathrm{T}} \boldsymbol{e}_i > 0$ then the change of $(f(\boldsymbol{\xi}_s^x, \boldsymbol{\xi}_a^u) - \boldsymbol{\xi}_{s'}^x)$ resulting from the subdivision in the i-th direction is given by $-(([C]_{s'}^x - \boldsymbol{\xi}_{s'}^x, \boldsymbol{e}_i)/2)\boldsymbol{e}_i$. On the other hand, if $(f(\boldsymbol{\xi}_s^x, \boldsymbol{\xi}_a^u) - \boldsymbol{\xi}_{s'}^x)^{\mathrm{T}} \boldsymbol{e}_i < 0$, the change of $(f(\boldsymbol{\xi}_s^x, \boldsymbol{\xi}_a^u) - \boldsymbol{\xi}_{s'}^x)$ will be $(([C]_{s'}^x - \boldsymbol{\xi}_{s'}^x, \boldsymbol{e}_i)/2)\boldsymbol{e}_i$. This means subdivision is effective if and only if $(f(\boldsymbol{\xi}_s^x, \boldsymbol{\xi}_a^u) - \boldsymbol{\xi}_{s'}^x)^{\mathrm{T}} \boldsymbol{e}_i$ and μ have the same sign.

Finally, let us consider reducing $([\mathcal{E}]_{t-1,s}, \mu[A]_{t-1,s,a}^{\mathrm{T}} \boldsymbol{e}_i)$. Like (16), we have

$$([\mathcal{E}]_{t-1,s}, \mu[A]_{t-1,s,a}^{\mathrm{T}} \boldsymbol{e}_i) = \sum_{j \in [1:n]} \max\{([\mathcal{E}]_{t-1,s}, \mu \underline{A}_{ij} \boldsymbol{e}_j), ([\mathcal{E}]_{t-1,s}, \mu \overline{A}_{ij} \boldsymbol{e}_j)\} \tag{17}$$

where $[\underline{A}_{ij}, \overline{A}_{ij}]$ denotes the (i,j)-element of $[A]_{t-1,s,a}$. But in this case, it is not straightforward to determine which cell (and in which direction) should be subdivided in order to reduce $([\mathcal{E}]_{t-1,s}, \mu \boldsymbol{e}_j)$ for given $\mu \in \mathbb{R}$ and $j \in [1:n]$. This is because $[\mathcal{E}]_{t-1,s}$ is itself an accumulated sum of quantization errors caused by all the predecessors of s. Hence, we shall repeat the same discussion as above in order to identify a cell and its direction to be subdivided. This leads us to the following recursive algorithm, which determines a cell-direction pair that has the most significant influence on $([\mathcal{E}]_{t,s'}, \mu \boldsymbol{e}_i)$.

identify_bottleneck(t, s', μ, i)
  if $t = 0$
    return $\{s', i, ([C]_{s'}^x - \boldsymbol{\xi}_{s'}^x, \mu \boldsymbol{e}_i)/2\}$
  else
    $\mathcal{C}_1 := \emptyset$, $\mathcal{C}_2 := \emptyset$, $\mathcal{C}_3 := \emptyset$
    $\{s, a\} := \mathrm{pre}(t, s', \mu, i)$
    $[A] := [A]([\mathcal{E}]_{t-1,s} + \boldsymbol{\xi}_s^x, [C]_a^u)$
    $[B] := [B]([\mathcal{E}]_{t-1,s} + \boldsymbol{\xi}_s^x, [C]_a^u)$
    if $\mathrm{sgn}((f(\boldsymbol{\xi}_s^x, \boldsymbol{\xi}_a^u) - \boldsymbol{\xi}_{s'}^x, \boldsymbol{e}_i)) = \mathrm{sgn}(\mu)$
      $\mathcal{C}_1 := \mathcal{C}_1 \cup \{s', i, ([C]_{s'}^x - \boldsymbol{\xi}_{s'}^x, \mu \boldsymbol{e}_i)/2\}$
    end
    for $j \in [1:m]$
      $\mathcal{C}_2 := \mathcal{C}_2 \cup \{a, j, ([C]_a^u - \boldsymbol{\xi}_a^u, \mu \underline{B}_{ij} \boldsymbol{e}_j)/2\}$
        $\cup \{a, j, ([C]_a^u - \boldsymbol{\xi}_a^u, \mu \overline{B}_{ij} \boldsymbol{e}_j)/2\}$
    end
    for $j \in [1:n]$
      $\mathcal{C}_3 := \mathcal{C}_3 \cup$ **identify_bottleneck**$(t-1, s, \mu \underline{A}_{ij}, j)$
        $\cup$ **identify_bottleneck**$(t-1, s, \mu \overline{A}_{ij}, j)$
    end
    return $\mathrm{argmax}_{\{c,d,p\} \in \mathcal{C}_1 \cup \mathcal{C}_2 \cup \mathcal{C}_3} p$
  end

Let $\{c^*, d^*, p^*\}$ be the return values of **identify_bottleneck**(t, s', μ, i). Here, the variable c^* could contain either a state symbol or an input symbol. In the former case, $[C]_{c^*}^x$ is a cell of $\mathcal{M}^x$, and in the latter case, $[C]_{c^*}^u$ is a cell of

$\mathcal{M}^u$. We may omit the superscript and write $[C]_{c^*}$ when the distinction is not necessary. Connecting the above components all together, we obtain the following algorithm.

refine_mesh
$\mathcal{M}^x := \{\{\mathbf{0}, X\}\}$, $\mathcal{M}^u := \{\{\mathbf{0}, U\}\}$
loop
propagate_error
$\{t, s', \mu, i, p\} :=$ **max_violation**
if $p \leq 0$ then terminate
$\{c^*, d^*, p^*\} :=$ **identify_bottleneck**(t, s', μ, i)
subdivide(c^*, d^*)
end

Here, **subdivide** is a procedure that divides the cell $[C]_{c^*}$ in the direction d^*.

In the following, we will show that the algorithm **refine_mesh** actually terminates in finite iterations and produces a pair of meshes that defines an N-step abstraction of the given original system. First, we introduce the following assumption:

Assumption 1. *There exists a positive constant ϵ that satisfies*

$$\{(\boldsymbol{x}, \hat{\boldsymbol{x}}) \mid \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_\infty \leq \epsilon\} \subseteq \bar{R}. \tag{18}$$

This assumption is fairly natural and is satisfied in most cases including uniform and relative error conditions shown in Section 2. Moreover, let us define

$$\lambda_A = \max_{\boldsymbol{x} \in X, \boldsymbol{u} \in U, i \in [1:n], j \in [1:n]} \left| \frac{\partial f_i}{\partial x_j}(\boldsymbol{x}, \boldsymbol{u}) \right|,$$
$$\lambda_B = \max_{\boldsymbol{x} \in X, \boldsymbol{u} \in U, i \in [1:n], j \in [1:m]} \left| \frac{\partial f_i}{\partial u_j}(\boldsymbol{x}, \boldsymbol{u}) \right|.$$

The next lemma claims that there exists a lower-bound on the size of cells that are subdivided at each iteration of **refine_mesh**.

Lemma 4. *In **refine_mesh**, a pair (c^*, d^*) passed to **subdivide** at each iteration satisfies*

$$([C]_{c^*} - \boldsymbol{\xi}_{c^*}, \boldsymbol{e}_{d^*}) \geq \frac{\epsilon}{\alpha_N \beta_N} \tag{19}$$

where $\alpha_0 = \beta_0 = 1$ and $\alpha_t = \max\{\max_{k \in [0:t-1]} \lambda_A^k \lambda_B, \max_{k \in [0:t]} \lambda_A^k\}$, $\beta_t = (\sum_{k=0}^{t} n^k + m \sum_{k=0}^{t-1} n^k)$ for $t \geq 1$.

Proof. At first, we will prove by construction that the output of **identify_bottleneck**(t, s', μ, i), denoted by $\{c^*, d^*, p^*\}$, satisfies

$$([\mathcal{E}]_{t,s'}, \mu \boldsymbol{e}_i) \leq 2p^* \beta_t, \tag{20}$$
$$([C]_{c^*} - \boldsymbol{\xi}_{c^*}, \boldsymbol{e}_{d^*}) \geq 2p^* / (|\mu| \alpha_t). \tag{21}$$

When $t=0$, **identify_bottleneck** immediately returns $\{s', i, ([C]^x_{s'} - \boldsymbol{\xi}^x_{s'}, \mu \boldsymbol{e}_i)/2\}$. Here, $([\mathcal{E}]_{0,s'}, \mu \boldsymbol{e}_i) = ([C]^x_{s'} - \boldsymbol{\xi}^x_{s'}, \mu \boldsymbol{e}_i) = 2p^*$. Moreover, $([C]^x_{s'} - \boldsymbol{\xi}^x_{s'}, \boldsymbol{e}_i) = 2p^*/|\mu|$. Therefore we confirm that (20) and (21) hold for $t = 0$.

Now, suppose (20) and (21) hold for $t = \tau - 1$ and consider the case of $t = \tau$. Let us denote $\{c_\rho, d_\rho, p_\rho\} = \mathrm{argmax}_{\{c,d,p\} \in \mathcal{C}_\rho} p$ for each $\rho \in \{1, 2, 3\}$ (recall that $\mathcal{C}_\rho$ are temporary variables that appear in **identify_bottleneck**). Then, p^* is given by $p^* = \max\{p_1, p_2, p_3\}$. Here we have

$$(f(\boldsymbol{\xi}^x_s, \boldsymbol{\xi}^u_a) - \boldsymbol{\xi}^x_{s'}, \mu \boldsymbol{e}_i) \le 2p_1, \quad ([C]^u_a - \boldsymbol{\xi}^u_a, \mu [B]^{\mathrm{T}}_{t-1,s,a} \boldsymbol{e}_i) \le 2mp_2.$$

Moreover, from (20) with $t = \tau - 1$ we have

$$([\mathcal{E}]_{\tau-1,s}, \mu [A]^{\mathrm{T}}_{\tau-1,s,a} \boldsymbol{e}_i) \le 2np_3\beta_{\tau-1}.$$

Therefore

$$([\mathcal{E}]_{\tau,s'}, \mu \boldsymbol{e}_i) \le 2p_1 + 2mp_2 + 2np_3\beta_{\tau-1} \le 2p^*\beta_\tau.$$

On the other hand, we have

$$\begin{aligned} &([C]^x_{s'} - \boldsymbol{\xi}^x_{s'}, \boldsymbol{e}_{d_1}) \ge 2p_1/|\mu|, \quad ([C]^u_a - \boldsymbol{\xi}^u_a, \boldsymbol{e}_{d_2}) \ge 2p_2/(|\mu|\lambda_B), \\ &([C]_{c_3} - \boldsymbol{\xi}_{c_3}, \boldsymbol{e}_{d_3}) \ge 2p_3/(|\mu|\lambda_A\alpha_{\tau-1}). \end{aligned}$$

This yields

$$([C]_{c^*} - \boldsymbol{\xi}_{c^*}, \boldsymbol{e}_{d^*}) \ge 2p^*/\max\{|\mu|, |\mu|\lambda_B, |\mu|\lambda_A\alpha_{\tau-1}\} \ge 2p^*/(|\mu|\alpha_\tau),$$

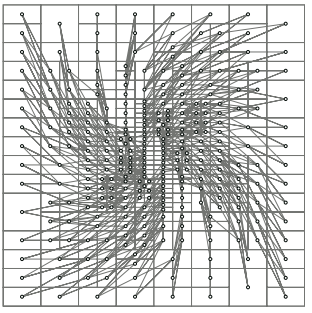
and we conclude that (20) and (21) hold for any $t \ge 0$.

From Assumption 1, we have $(\bar{\mathcal{E}}_{s'}, \mu \boldsymbol{e}_i) \ge \epsilon$ for $\mu \in \{-1, 1\}$. This means $([\mathcal{E}]_{t,s'}, \mu \boldsymbol{e}_i) \ge \epsilon$ holds for a tuple $\{t, s', \mu, i\}$ passed to **identify_bottleneck**. From this together with (20) and (21) we obtain $([C]_{c^*} - \boldsymbol{\xi}_{c*}, \boldsymbol{e}_{d^*}) \ge \epsilon/(\alpha_t\beta_t)$. Since the right hand side of this inequality monotonically decreases with respect to t, $([C]_{c^*} - \boldsymbol{\xi}_{c*}, \boldsymbol{e}_{d^*}) \ge \epsilon/(\alpha_N\beta_N)$ holds for all $t \in [0 : N]$. □

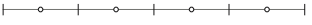
Now we come to our main result.

Theorem 1. *For a system Σ and a binary relation $\bar{R}$ satisfying Assumption 1, the algorithm **refine_mesh** terminates in finite iterations. Moreover, its outputs $\mathcal{M}^x$ and $\mathcal{M}^u$ define a quantizer embedding $\mathrm{QE}(\Sigma, Q[\mathcal{M}^x], Q[\mathcal{M}^u])$ that is an N-step abstraction of Σ with respect to $\bar{R}$.*

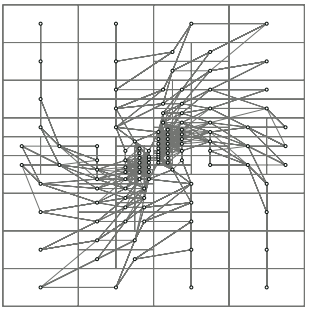
Proof. **refine_mesh** will not terminate before the error condition (14) is satisfied. On the other hand, from Lemma 4, the size of a cell subdivided at each iteration has a positive lower-bound. Since we subdivide a cell in half, the amount a cell shrinks at each subdivision also has a positive lower-bound. Moreover, the state set X and the input set U are both bounded. These facts prove the theorem.



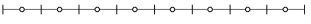
(a) State mesh for uniform error margin



(b) Input mesh for uniform error margin



(c) State mesh for relative error margin



(d) Input mesh for relative error margin

Fig. 2. Discrete abstraction of a linear system

Table 1. Growth of the number of cells

N	0	1	2	3	4	5	6	7	8	-	20
state grid	64	149	208	238	264	283	293	295	296	-	296
input grid	1	2	4	4	4	4	4	4	4	-	4

(a) Uniform error margin

N	0	1	2	3	4	5	6	7	-	10	11	12	-	20
state grid	4	6	14	42	73	97	109	118	-	120	121	121	-	121
input grid	1	1	1	2	4	6	6	8	-	8	8	10	-	10

(b) Relative error margin

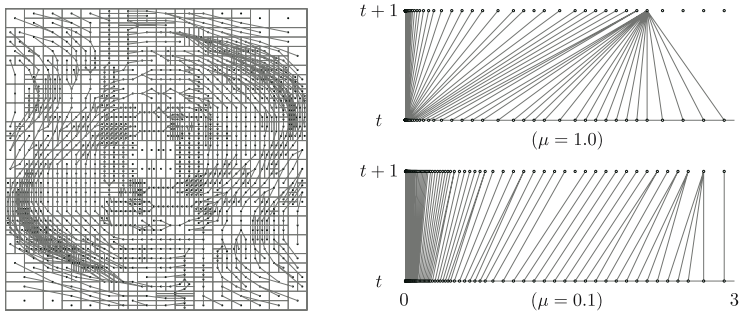
5 Examples

This section shows some simple examples as graphical demonstrations.

Consider the following 2-dimensional linear system:

$$\boldsymbol{x}_{t+1} = A\boldsymbol{x}_t + Bu_t, \quad A = \begin{bmatrix} 0.68 & -0.14 \\ 0.14 & 0.68 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 0.1 \end{bmatrix}.$$

The state set is given as $X = [-1, 1] \times [-1, 1]$ and the input set is given as $U = [-1, 1]$. For this system, at first we compute a discrete abstraction using the uniform error condition $\bar{R} = \{(\boldsymbol{x}, \hat{\boldsymbol{x}}) \,|\, \|\boldsymbol{x} - \hat{\boldsymbol{x}}\| \leq 0.3\}$. The result is shown in Fig. 2(a),(b). The discrete abstraction obtained is composed of 296 states and 4 inputs. Next, for the same system we specify a relative error condition given as $\bar{R} = \{(\boldsymbol{x}, \hat{\boldsymbol{x}}) \,|\, \|\boldsymbol{x} - \hat{\boldsymbol{x}}\| \leq 0.2 + 0.7\|\hat{\boldsymbol{x}}\|\}$. This kind of error condition is particularly useful when only the systems behavior near the origin is of interest. The result is shown in Fig. 2(c),(d). In this case, the discrete abstraction is composed of



(a) Euler stepping (b) Poincaré section

Fig. 3. Discrete abstraction of val der Pol oscillator

120 states and 8 inputs. Table1(a)(b) show how the number of cells grows with respect to the step length N. We can observe that the number of cells does not increase when N becomes larger than a certain value (in these examples, less than 20). This implies a possibility that these discrete abstractions are valid for infinite steps. More theoretical study is needed in a future work.

Next, we consider the discrete abstraction of the van der Pol oscillator as an example of nonlinear periodic systems. The van der Pol oscillator is governed by the following ordinary differential equation:

$$\ddot{x} = \mu(1 - x^2)\dot{x} - x.$$

In order to apply our method, this system should be transformed into a discrete-time system. We investigate two ways of time-discretization; Euler stepping and Poincaré section. This time, we choose the Poincaré section as $x \in [0.0, 3.0], \dot{x} = 0$. Fig.3(a) shows a discrete abstraction of the van der Pol oscillator in the Euler stepping case. The parameters are set as $\mu = 0.5, h = 0.1$ and $N = 6$, where h denotes the step size. The state set is given as $X = [-3, 3] \times [-3, 3]$. The approximation precision is given as $\bar{R} = \{(\boldsymbol{x}, \hat{\boldsymbol{x}}) \mid \|\boldsymbol{x} - \hat{\boldsymbol{x}}\| \leq 0.5\}$. In this result, the discrete abstraction consists of 1294 discrete states. On the other hand, Fig.3(b) shows the case of Poincaré section. We can observe that the mesh is densely subdivided near the origin. This reflects the fact that a small difference of initial states near the origin could cause a significant difference of the number of cycles required before converging to the periodic orbit.

6 Conclusion

In this paper, we have presented a computational approach to the discrete abstraction of nonlinear systems. The presented approach works well even within the framework of discrete-state abstractions of interconnected systems developed in [15]. That is to say, based on the result of [15], we can treat the case in which our approach is applied to only a complex subsystem of the whole system, which produces a kind of hybrid abstraction.

References

1. Lafferriere, G., Pappas, G.J., Sastry, S.S.: Hybrid Systems with Finite Bisimulations. In: Antsaklis, P.J., Kohn, W., Lemmon, M.D., Nerode, A., Sastry, S.S. (eds.) HS 1997. LNCS, vol. 1567, pp. 186–203. Springer, Heidelberg (1999)
2. Broucke, M.E.: A geometric approach to bisimulation and verification of hybrid systems. In: Vaandrager, F.W., van Schuppen, J.H. (eds.) HSCC 1999. LNCS, vol. 1569, pp. 61–75. Springer, Heidelberg (1999)
3. Habets, L.C.G.J.M., Collins, P.J., van Schuppen, J.H.: Reachability and control synthesis for piecewise-affine hybrid systems on simplices. IEEE Transactions on Automatic Control 51(6), 938–948 (2006)
4. Kloetzer, M., Belta, C.: A Fully Automated Framework for Control of Linear Systems from LTL Specifications. In: Hespanha, J.P., Tiwari, A. (eds.) HSCC 2006. LNCS, vol. 3927, pp. 333–347. Springer, Heidelberg (2006)
5. Caines, P.E., Wei, Y.: Hierarchical Hybrid Control Systems: A Lattice Theoretic Formulation. IEEE Trans. on Automatic Control 43(4), 501–508 (1998)
6. Alur, R., Verimag, T.D., Ivančić, F.: Predicate Abstractions for Reachability Analysis of Hybrid Systems. ACM Trans. on Embedded Computing Systems 5(1), 152–199 (2006)
7. Lunze, J., Nixdorf, B., Schröder, J.: Deterministic Discrete-event Representations of Linear Continuous-variable Systems. Automatica 35, 395–406 (1999)
8. Koutsoukos, X.D., Antsaklis, P.J., Stiver, J.A., Lemmon, M.D.: Supervisory Control of Hybrid Systems. In: Antsaklis, P.J. (ed.) Proc. of IEEE, Special Issue in Hybrid Systems, July 2000, pp. 1026–1049 (2000)
9. Raisch, J., O'Young, S.D.: Discrete Approximation and Supervisory Control of Continuous Systems. IEEE Trans. on Automatic Control 43(4), 569–573 (1998)
10. Girard, A., Pappas, G.J.: Approximation metrics for discrete and continuous systems. IEEE Transactions on Automatic Control 52(5), 782–798 (2007)
11. Girard, A.: Approximately bisimilar finite abstractions of stable linear systems. In: Bemporad, A., Bicchi, A., Buttazzo, G. (eds.) HSCC 2007. LNCS, vol. 4416, pp. 231–244. Springer, Heidelberg (2007)
12. Tabuada, P.: Approximate simulation relations and finite abstractions of quantized control systems. In: Bemporad, A., Bicchi, A., Buttazzo, G. (eds.) HSCC 2007. LNCS, vol. 4416, pp. 529–542. Springer, Heidelberg (2007)
13. Pola, G., Girard, A., Tabuada, P.: Symbolic models for nonlinear control systems using approximate bisimulation. In: 46th IEEE Conference on Decision and Control, pp. 4656–4661 (2007)
14. Tazaki, Y., Imura, J.: Finite Abstractions of Discrete-time Linear Systems and Its Application to Optimal Control. In: 17th IFAC World Congress (2008) (in CDROM)
15. Tazaki, Y., Imura, J.-i.: Bisimilar finite abstractions of interconnected systems. In: Egerstedt, M., Mishra, B. (eds.) HSCC 2008. LNCS, vol. 4981, pp. 514–527. Springer, Heidelberg (2008)
16. Ramdani, N., Meslem, N., Candau, Y.: Reachability of uncertain nonlinear systems using a nonlinear hybridization. In: Egerstedt, M., Mishra, B. (eds.) HSCC 2008. LNCS, vol. 4981, pp. 415–428. Springer, Heidelberg (2008)

Event-Triggering in Distributed Networked Systems with Data Dropouts and Delays

Xiaofeng Wang and Michael D. Lemmon*

University of Notre Dame, Department of Electrical Engineering,
Notre Dame, IN, 46556, USA
xwang13,lemmon@nd.edu

Abstract. This paper studies distributed networked systems with data dropouts and transmission delays. We propose an event-triggering scheme, where a subsystem broadcasts its state information to its neighbors only when the subsystem's local state error exceeds a specified threshold. This scheme is completely decentralized, which means that a subsystem's broadcast decisions are made using its local sampled data, the maximal allowable transmission delay of a subsystem's broadcast is predicted based on the local information, a subsystem locally identifies the maximal allowable number of its successive data dropouts, and the designer's selection of the threshold only requires information about an individual subsystem and its immediate neighbors. With the assumption that the number of each subsystem's successive data dropouts is less than the bound identified by that subsystem, if the bandwidth of the network is limited so that the transmission delays are always greater than a positive constant, the resulting system is globally uniformly ultimately bounded using our scheme; otherwise, the resulting system is asymptotically stable.

1 Introduction

A networked control system (NCS) is a system wherein numerous physically coupled subsystems are geographically distributed throughout the system. Control and feedback signals are exchanged through a real-time network among the system's components. Specific examples of NCS include electrical power grids and transportation networks. The networking of control effort can be advantageous in terms of lower system costs due to streamlined installation and maintenance costs. The introduction of real-time network infrastructure, however, raises new challenges regarding the impact that communication reliability has on the control system's performance. In real-time networks, information is transmitted in discrete-time rather than continuous-time. Moreover, all real networks have bandwidth limitation that can cause delays in message delivery that may have a major impact on overall system stability [1].

* The authors gratefully acknowledge the partial financial support of the National Science Foundation (NSF CNS-0720457).

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 366–380, 2009.

For this reason, some researchers began investigating the timing issue in NCS. One packet transmission problem was considered in [2], [3], where a supervisor summarizes all subsystem data into this single packet. As a result such schemes may be impractical for large-scale systems. Asynchronous transmission was considered in [4], [5], [6], which derived bounds on the maximum allowable transfer interval (MATI) between two subsequent message transmissions so that the system stability can be guaranteed. All of this prior work confined its attention to control area network (CAN) buses where centralized computers are used to coordinate the information transmission.

One thing worth mentioning is that these schemes mentioned above require extremely detailed models of subsystem interactions and the execution of communication protocols must be done in a highly centralized manner. Both of these requirements can greatly limit the scalability of centralized approaches to NCS. On the other hand, the MATI is computed before the system is deployed, which means it is independent of the system state. So it must ensure adequate behavior over a wide range of possible system states. As a result, it may be conservative.

To overcome these issues, decentralized event-triggering feedback schemes were proposed in [7] and [8] for linear and nonlinear systems, respectively. Most recently, an implementation of event-triggering in sensor-network was introduced in [11]. By event-triggering, a subsystem broadcasts its state information to its neighbors only when "needed". In this case, "needed" means that some measure of the subsystem's local state error exceeds a specified threshold [9], [10]. In this way, event-triggering makes it possible to reduce the frequency with which subsystems communicate and therefore use network bandwidth in an extremely frugal manner. An important assumption in [7], [8] is that neither data dropouts nor delays occur in such systems. In real-time network, however, especially wireless network, data dropouts and delays always exist. Therefore, it suggests a more complete consideration of such systems.

This paper studies the distributed NCS with data dropouts and transmission delays. Unlike the prior work that modelled data dropouts as stochastic processes using a centralized approach [12], [13], we propose an event-triggering scheme that enables a subsystem to locally identify the maximal allowable number of its successive data dropouts. This scheme is "completely" decentralized. By "complete", it means that (1) a subsystem's broadcast decisions are made using its local sampled data, (2) the maximal allowable transmission delay (also called "deadline") of a subsystem's broadcast can be predicted based on the local information, (3) a subsystem locally identifies the maximal allowable number of its successive data dropouts, and (4) the designer's selection of the threshold only requires information about an individual subsystem and its immediate neighbors.

Our analysis applies to nonlinear continuous systems. With the assumption that the number of each subsystem's successive data dropouts is less than the bound identified by that subsystem, if the bandwidth of the network is limited so that the transmission delays are always greater than a positive constant, the resulting NCS is globally uniformly ultimately bounded using our scheme;

otherwise, the resulting NCS is asymptotically stable. We use an example to illustrate the design procedure.

The paper is organized as follows: section 2 formulates the problem; the decentralized approach to design the local triggering event is introduced in section 3; Transmission delays and data dropouts are considered in section 4 and 5, respectively; Simulation results are presented in section 6; In section 7, the conclusions are drawn.

2 Problem Formulation

Consider a distributed NCS containing N subsystems, denoted as $\mathbf{P}_i$. Let $\mathcal{N} = \{1, 2, \cdots, N\}$. $Z_i \in \mathcal{N}$ denotes the set of subsystems that $\mathbf{P}_i$ can get information from; $D_i \subset \mathcal{N}$ denotes the set of subsystems that directly drive $\mathbf{P}_i$'s dynamics; $U_i \in \mathcal{N}$ denotes the set of subsystems that can receive $\mathbf{P}_i$'s broadcasted information; $S_i \in \mathcal{N}$ denotes the set of subsystems who are directly driven by $\mathbf{P}_i$. For a set $\Phi \in \mathcal{N}$, we use $|\Phi|$ to denote the number of elements in Φ.

The state equation of the ith subsystem is

$$\begin{aligned} \dot{x}_i(t) &= f_i(x_{D_i}(t), u_i) \\ u_i &= \gamma_i(x_{Z_i}(t)) \\ x_i(t_0) &= x_{i0}. \end{aligned} \tag{1}$$

Our analysis applies to the case where the states have different dimensions. However, to outline the main idea, we assume $x_i \in \mathbb{R}^n$ for all i. In equation (1), $x_{D_i} = \{x_j\}_{j \in D_i}$, $x_{Z_i} = \{x_j\}_{j \in Z_i}$, $\gamma_i : \mathbb{R}^{n|Z_i|} \to \mathbb{R}^{m_i}$ is the given feedback strategy of agent i satisfying $\gamma_i(0) = 0$, and $f_i : \mathbb{R}^{n|D_i|} \times \mathbb{R}^{m_i} \to \mathbb{R}^n$ is a given continuous function satisfying $f_i(0, 0) = 0$.

This paper considers a real-time implementation of this distributed NCS. The infrastructure of such an implementation is plotted in figure 1. In such a system,

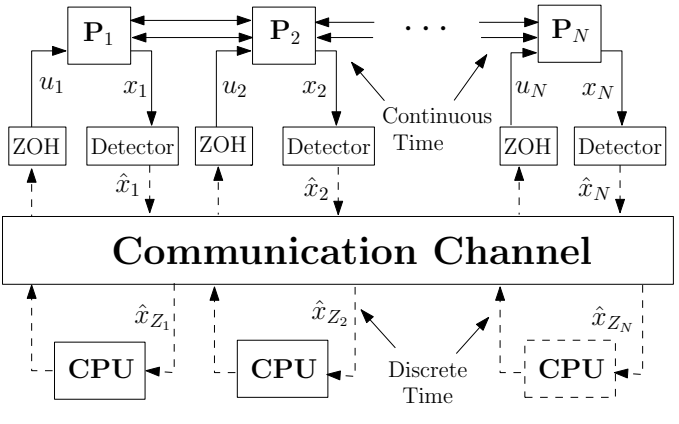


Fig. 1. The infrastructure of the real-time NCS

$\mathbf{P}_i$ can only detect its own state, x_i. If the local "error" signal exceeds some given threshold, which can be detected by hardware detectors, $\mathbf{P}_i$ will sample and broadcast its state information to its neighbors through a real-time network. Therefore, $\mathbf{P}_i$'s control, u_i, at time t is computed based on its neighbors' latest broadcast states (also called "measured states") at time t, denoted as $\hat{x}_{Z_i}(t)$. We assume that the time spent in computing the control and sending the control back to the plant is zero. The control signal used by $\mathbf{P}_i$ is held constant by a zero-order hold (ZOH) unless one of its neighbors makes another broadcast. This means that $\mathbf{P}_i$ has the following state equation,

$$\begin{aligned} \dot{x}_i(t) &= f_i(x_{D_i}(t), u_i) \\ u_i &= \gamma_i(\hat{x}_{Z_i}(t)) \\ x_i(t_0) &= x_{i0}. \end{aligned} \tag{2}$$

Subsystem i's broadcast can characterized by two monotone increasing sequences of time instants: the broadcast release time sequence $\{r_k^i\}_{k=1}^{\infty}$ and the broadcast finishing time $\{f_k^i\}_{k=1}^{\infty}$, where $r_k^i \leq f_k^i \leq r_{k+1}^i$ holds for all $k = 1, 2, \cdots, \infty$. The time r_k^i denotes the time when the kth broadcast is released by $\mathbf{P}_i$ for transmission through the channel. At this time, we assume there is no delay between sampling and broadcast release. The time f_k^i denotes the time when the kth broadcast information by $\mathbf{P}_i$ is received by its neighbors. The objective of this paper is to develop decentralized event-triggering schemes to identify $\{r_k^i\}_{k=1}^{\infty}$, $\{f_k^i\}_{k=1}^{\infty}$ such that the NCS defined in equation (2) is asymptotically stable or globally uniformly ultimately bounded defined as follows.

Definition 1. *The system in equation (2) is said to be globally uniformly ultimately bounded with ultimate bound ϵ, if there exists a positive constant $\epsilon \in \mathbb{R}$, independent of t_0, and for any $s > 0$, there exists $T \geq 0$, independent of t_0, such that*

$$\|x(t_0)\|_2 \leq s \Rightarrow \|x(t)\|_2 \leq \epsilon, \ \forall t \geq t_0 + T \tag{3}$$

3 Decentralized Broadcast-Triggering Events Design

In this section, we study a decentralized approach to characterize the broadcast time sequence. Inequality constraints on each subsystem's broadcast release and finishing time are provided to ensure asymptotic stability of the overall system. These constraints can be locally determined by individual subsystems. To obtain the decentralized method, we first introduce a theorem in [8] that provides a centralized approach to derive the time constraints on r_k^i and f_k^i. For notational convenience, we define $e_k^i : [r_k^i, f_{k+1}^i) \to \mathbb{R}^n$ as $e_k^i(t) = x_i(t) - x_i(r_k^i)$ for $\forall t \in [r_k^i, f_{k+1}^i)$. Notice that $\hat{x}_i(t) = x_i(r_k^i)$ for all $t \in [f_k^i, f_{k+1}^i)$.

Theorem 1 ([8]). *Consider the NCS in equation (2). Assume that there exist a smooth, positive-definite function $V : \mathbb{R}^{nN} \to \mathbb{R}$ and class $\mathcal{K}$ functions $\underline{\zeta}$, $\bar{\zeta}$, ϕ_i, $\psi_i : \mathbb{R} \to \mathbb{R}$ for $i = 1, \cdots, N$ such that*

$$\underline{\zeta}(\|x\|_2) \leq V(x) \leq \bar{\zeta}(\|x\|_2) \tag{4}$$

$$\sum_{i\in\mathcal{N}} \frac{\partial V(x)}{\partial x_i} f_i\left(x_{D_i}, \gamma_i(y_{Z_i})\right) \leq \sum_{i\in\mathcal{N}} -\phi_i(\|x_i\|_2) + \sum_{i\in\mathcal{N}} \psi_i(\|x_i - y_i\|_2) \tag{5}$$

holds for all $x, y \in \mathbb{R}^{nN}$. *If for any* $i \in \mathcal{N}$, *there exists a constant* $\rho_i \in (0,1)$ *such that subsystem* i*'s broadcast release time sequence,* $\{r_k^i\}_{k=1}^{\infty}$, *and finishing time sequence,* $\{f_k^i\}_{k=1}^{\infty}$, *satisfy*

$$-\rho_i\phi_i(\|x_i(t)\|_2) + \psi_i(\|e_k^i(t)\|_2) \leq 0 \tag{6}$$

for all $t \in [f_k^i, f_{k+1}^i)$ *and all* $k \in \mathbb{N}$, *then the NCS is asymptotically stable.*

Theorem 1 shows that the satisfaction of equation (6) guarantees asymptotic stability of the NCS. Based on this theorem, deriving local time constraints is equivalent to constructing class $\mathcal{K}$ functions ϕ_i and ψ_i. The following theorem provides a decentralized approach to design such class $\mathcal{K}$ functions.

Theorem 2. *Consider the NCS defined in equation (2). Assume that there exist smooth, positive-definite functions* $V_i : \mathbb{R}^n \to \mathbb{R}$, *class* $\mathcal{K}$ *functions* $\underline{\zeta_i}, \bar{\zeta}_i : \mathbb{R} \to \mathbb{R}$, *positive constants* $\alpha_i, \beta_i, \kappa_i \in \mathbb{R}$, *and control law* $\gamma_i : \mathbb{R}^{n|Z_i|} \to \mathbb{R}^{m_i}$ *for* $\forall i \in \mathcal{N}$ *satisfying*

$$\underline{\zeta_i}(\|x_i\|_2) \leq V_i(x) \leq \bar{\zeta}_i(\|x_i\|_2) \tag{7}$$

$$\frac{\partial V_i(x_i)}{\partial x_i} f_i(x_{D_i}, \gamma_i(y_{Z_i})) \leq \sum_{j\in D_i\cup Z_i} \beta_j\|x_j\|_2 + \sum_{j\in Z_i} \kappa_j\|x_j - y_j\|_2 - \alpha_i\|x_i\|_2 \tag{8}$$

$$\alpha_i - |S_i \cup U_i|\beta_i > 0 \tag{9}$$

Then $\phi_i, \psi_i : \mathbb{R} \to \mathbb{R}$, *defined by* $\phi_i(s) = a_i s$ *and* $\psi_i(s) = b_i s$, *satisfy equation (6) in theorem 1, where*

$$a_i = \alpha_i - |S_i \cup U_i|\beta_i \tag{10}$$

$$b_i = |U_i|\kappa_i. \tag{11}$$

Proof. It is easy to see that

$$\begin{aligned}
&\sum_{i\in\mathcal{N}} \frac{\partial V(x_i)}{\partial x_i} f_i(x_{D_i}, \gamma_i(y_{Z_i})) \\
\leq &\sum_{i\in\mathcal{N}} -\alpha_i\|x_i\|_2 + \sum_{j\in D_i\cup Z_i} \beta_j\|x_j\|_2 + \sum_{j\in Z_i} \kappa_j\|x_j - y_j\|_2 \\
= &\sum_{i\in\mathcal{N}} \left(-\alpha_i + |S_i \cup U_i|\beta_i\right)\|x_i\|_2 + \sum_{i\in\mathcal{N}} |U_i|\kappa_i\|x_i - y_i\|_2,
\end{aligned}$$

where the equality is obtained by resorting the items according to index i. □

Remark 1. Equation (8) and (9) may have a more general form, where $\alpha_i\|x_i\|_2$, $\beta_j\|x_j\|_2$, and $\kappa_j\|x_j - y_j\|_2$ are replaced by some class $\mathcal{K}$ functions. Using the

more general form, however, will require additional assumptions on those class $\mathcal{K}$ functions in the later discussion, such as a Lipschitz condition. It will make the paper hard to read. To outline the main idea of this paper, we just use equation (8) and (9) as a sufficient condition to construct ϕ_i and ψ_i in theorem 1.

Remark 2. Equation (8) suggests that subsystem i is finite-gain $\mathcal{L}_2$ stable from $(\{x_j\}_{j\in D_i\cup Z_i}, \{x_j - y_j\}_{j\in Z_i})$ to x_i.

We will find that it is convenient in the later work to use a slightly weaker sufficient condition for asymptotic stability where the state error $e_k^i(t)$ is bounded by a function of the sampled data $x_i(r_k^i)$ as stated in the following corollary.

Corollary 1. *Consider the NCS in equation (2). Assume that equation (7), (8), (9) hold. If for any $i \in \mathcal{N}$, subsystem i's broadcast release time sequence, $\{r_k^i\}_{k=1}^{\infty}$, and finishing time sequence, $\{f_k^i\}_{k=1}^{\infty}$, satisfy*

$$\|e_k^i(t)\|_2 \le c_i \|x_i(r_k^i)\|_2 \tag{12}$$

for all $t \in [f_k^i, f_{k+1}^i)$ and all $k \in \mathbb{N}$, where $c_i \in \mathbb{R}$ is defined by

$$c_i = \frac{\rho_i a_i}{\rho_i a_i + b_i}, \tag{13}$$

for some $\rho_i \in (0,1)$ and a_i, b_i are defined in equation (10), (11), respectively, then the NCS is asymptotically stable.

Proof. By the definition of c_i in equation (13), equation (12) is equivalent to

$$b_i \|e_k^i(t)\|_2 + \rho_i a_i \|e_k^i(t)\|_2 \le \rho_i a_i \|x_i(r_k^i)\|_2 \tag{14}$$

for all $t \in [f_k^i, f_{k+1}^i)$ and all $k \in \mathbb{N}$. Therefore, we have

$$\begin{aligned} b_i \|e_k^i(t)\|_2 &\le \rho_i a_i \|x_i(r_k^i)\|_2 - \rho_i a_i \|e_k^i(t)\|_2 \\ &\le \rho_i a_i \|x_i(r_k^i) + e_k^i(t)\|_2 = \rho_i a_i \|x_i(t)\|_2 \end{aligned}$$

for all $t \in [f_k^i, f_{k+1}^i)$ and all $k \in \mathbb{N}$. Since the hypotheses of theorem 1 are satisfied, we can conclude that the NCS is asymptotically stable. □

Remark 3. The inequalities in equations (6) or (12) can both be used as the basis for a decentralized event-triggered feedback control system. Note that both inequalities are trivially satisfied at $t = r_k^i$. If we let the delay be zero for each broadcast ($r_k^i = f_k^i$) and assume there are not any data dropouts, then by triggering the release times $\{r_k\}_{k=0}^{\infty}$ anytime before the inequalities in equations (6) or (12) are violated, we will ensure the sampled-data system's stability.

Theorem 2 and corollary 1 provide ways to identify the broadcast release time, r_k^i. The broadcast release is triggered when the violation of equation (5) or (12) occurs. However, we still do not know how to predict maximal allowable delays for each broadcast. In other words, we do not have an explicit constraint on f_k^i yet. In the following section, we will consider the bounds on f_k^i.

4 Event-Triggering with Delays

In this section, we quantify maximal allowable delays for each subsystem that will not break the stability of the NCS. An upper bound on the kth broadcast finishing time is derived in a decentralized manner as a function of the previously sampled local states.

We assume that there always exist $p_i > 0$ such that

$$\|f_i\left(x_{D_i}(t), \gamma_i\left(\hat{x}_{Z_i}(t)\right)\right)\|_2 \leq p_i \tag{15}$$

holds for any $t \geq 0$ and $i \in \mathcal{N}$.

Remark 4. The assumption in equation (15) requires the state trajectory of the NCS falls into some compact set $S \subset \mathbb{R}^n$. Such an assumption can also be seen in [9].

To obtain the upper bound on the delays, we need a lemma to identify the behavior of $e^i_{k-1}(t)$ and $e^i_k(t)$ over the time interval $[r^i_k, f^i_k)$. Ideally, we hope that $\|e^i_{k-1}(f^i_k)\|_2 \leq c_i\|x_i(r^i_{k-1})\|_2$ holds. In that case, the constraint $\|e^i_{k-1}(t)\|_2 \leq c_i\|x_i(r^i_{k-1})\|_2$ will not be violated over $[f^i_{k-1}, f^i_k)$. At the same time, we require that $\|e^i_k(f^i_k)\|_2 \leq \delta_i c_i\|x_i(r^i_k)\|_2$ holds for some $\delta_i \in (0,1)$. This is to ensure $r^i_{k+1} \geq f^i_k$ when we use the violation of $\|e^i_k(t)\|_2 \leq \delta_i c_i\|x_i(r^i_k)\|_2$ to trigger r^i_{k+1}. The lemma is stated as follows.

Lemma 1. *Consider subsystem i in equation (2). Assume that equation (15) holds for some $p_i \in \mathbb{R}^+$. For any $k \in \mathbb{N}$, if*

$$\|e^i_{k-1}(r^i_k)\|_2 \leq \delta_i c_i\|x_i(r^i_{k-1})\|_2 \tag{16}$$

$$f^i_k - r^i_k \leq \min\left\{\frac{(1-\delta_i)c_i}{p_i}\|x_i(r^i_{k-1})\|_2, \frac{\delta_i c_i}{p_i}\|x_i(r^i_k)\|_2\right\} \tag{17}$$

hold for some $\delta_i \in (0,1)$, then

$$\|e^i_{k-1}(t)\|_2 \leq c_i\|x_i(r^i_{k-1})\|_2 \tag{18}$$

$$\|e^i_k(t)\|_2 \leq \delta_i c_i\|x_i(r^i_k)\|_2 \tag{19}$$

hold for all $t \in [r^i_k, f^i_k)$.

Proof. Consider the derivative of $\left\|e^i_{k-1}(t)\right\|_2$ over the time interval $[r^i_k, f^i_k)$.

$$\frac{d}{dt}\left\|e^i_{k-1}(t)\right\|_2 \leq \left\|\dot{e}^i_{k-1}(t)\right\|_2 = \|\dot{x}_i(t)\|_2 = \|f_i\left(x_{D_i}, \gamma_i\left(\hat{x}_{Z_i}\right)\right)\|_2 \leq p_i$$

holds for all $t \in [r^i_k, f^i_k)$.

Solving the preceding inequality with initial condition $\left\|e^i_{k-1}(r^i_k)\right\|_2$ implies

$$\left\|e^i_{k-1}(t)\right\|_2 \leq p_i(t - r^i_k) + \left\|e^i_{k-1}(r^i_k)\right\|_2 \leq p_i(f^i_k - r^i_k) + \left\|e^i_{k-1}(r^i_k)\right\|_2 \tag{20}$$

holds for all $t \in [r^i_k, f^i_k)$.

By equation (17), we know

$$p_i(f_k^i - r_k^i) \leq (1-\delta_i)c_i\|x_i(r_{k-1}^i)\|_2 \tag{21}$$

Applying equation (16) and (21) into (20), we know equation (18) holds. With a similar analysis, we can show the satisfaction of equation (19). □

With lemma 1, we can present the following theorem where the upper bounds on delays of subsystems' broadcasts are given to guarantee asymptotic stability of the event-triggered NCS.

Theorem 3. *Consider the NCS in equation (2). Assume that equation (7), (8), (9), (15) hold. If, for any $i \in \mathcal{N}$, the broadcast release time r_{k+1}^i is triggered by the violation of the inequality*

$$\|e_k^i(t)\|_2 \leq \delta_i c_i \|x_i(r_k^i)\|_2 \tag{22}$$

for some $\delta_i \in (0,1)$ and the broadcast finishing time, f_{k+1}^i, satisfies

$$f_{k+1}^i - r_{k+1}^i \leq \min\left\{\frac{(1-\delta_i)c_i}{p_i}\|x_i(r_k^i)\|_2, \frac{\delta_i c_i}{p_i}\|x_i(r_{k+1}^i)\|_2\right\}, \tag{23}$$

then the NCS is asymptotically stable.

Proof. Since the hypotheses in lemma 1 hold, we have

$$\|e_k^i(t)\|_2 \leq c_i\|x_i(r_k^i)\|_2 \tag{24}$$

hold for all $t \in [r_{k+1}^i, f_{k+1}^i)$ and all $k \in \mathbb{N}$.

We also know by equation (22) that

$$\|e_k^i(t)\|_2 \leq \delta_i c_i \|x_i(r_k^i)\|_2 \tag{25}$$

holds for all $t \in [r_k^i, r_{k+1}^i)$ and all $k \in \mathbb{N}$.

Combining equation (24), (25) yields

$$\|e_k^i(t)\|_2 \leq c_i\|x_i(r_k^i)\|_2 \tag{26}$$

for all $t \in [r_k^i, f_{k+1}^i)$ and all $k \in \mathbb{N}$. Therefore, by corollary 1, the NCS is asymptotically stable. □

Remark 5. Notice that the maximal allowable delay of subsystem i's $k+1$st broadcast only depend on local information. In other words, subsystem i can design the deadline prediction law and predict the deadline by itself. The cost of such decentralized design is that the deadlines will go to zero as the state converges to the equilibrium. It might be possible to derive deadlines that are greater than a positive constant with the guarantee of asymptotic stability using centralized design. This would be an interesting research topic in the future.

In theorem 3, subsystem i predicts the deadline for its $k+1$st broadcast delay at time r^i_{k+1} when the state $x_i(r^i_{k+1})$ is sampled. It is more reasonable to have subsystem i predict the deadline for its $k+1$st delay ahead of time, such as at time r^i_k. Corollary 2 provides such a deadline as a function of $x_i(r^i_k)$.

Corollary 2. *Assume that all hypotheses in theorem 3 are satisfied except that equation (23) is replaced by*

$$f^i_{k+1} - r^i_{k+1} \leq \min\left\{ \frac{(1-\delta_i)c_i}{p_i}\|x_i(r^i_k)\|_2, \frac{\delta_i c_i(1-\delta_i c_i)}{p_i}\|x_i(r^i_k)\|_2 \right\}, \quad (27)$$

then the NCS is asymptotically stable.

Proof. By the definition of c_i in equation (13), we know $c_i < 1$ and therefore $1-\delta_i c_i > 0$ with $\delta_i \in (0,1)$. Based on equation (22), we know $(1-\delta_i c_i)\|x_i(r^i_k)\|_2 \leq \|x_i(r^i_{k+1})\|_2$. So equation (27) implies the satisfaction of equation (23). □

As we can see from equation (23) in theorem 3, the predicted deadlines for subsystem i's broadcast delays go to zero as the state converges to the equilibrium point. If the channel capacity is not taken into account, this result is acceptable. However, if the bandwidth of the network is limited, the broadcast delays are greater than a positive constant. In that case, the overall system will not be asymptotically stable. Instead, the state will eventually stay in a small neighborhood of the equilibrium, which means that the system is globally uniformly ultimately bounded. The size of the neighborhood depends on the length of the maximal delay. The results are formally stated as follows.

Corollary 3. *Assume that all hypotheses in theorem 3 are satisfied except that equation (23) is replaced by*

$$f^i_{k+1} - r^i_{k+1} \leq \min\left\{ \frac{(1-\delta_i)c_i\epsilon}{p_i}, \frac{\delta_i c_i \epsilon}{p_i} \right\} \quad (28)$$

for some positive constant $\epsilon \in \mathbb{R}^+$, then the NCS is globally uniformly ultimately bounded with an ultimate bound $\frac{\epsilon \sum_{i\in\mathcal{N}}(1-\delta_i)a_i}{\min_i(1-\delta_i)a_i}$, where a_i is defined in (10).

Proof. Following a similar analysis to the proof of theorem 3, we know that

$$\begin{aligned}
\dot{V} &\leq \sum_{i\in\mathcal{N}}(1-\delta_i)a_i(\epsilon - \|x_i(t)\|_2) \\
&\leq \epsilon \sum_{i\in\mathcal{N}}(1-\delta_i)a_i - \min_i(1-\delta_i)a_i \sum_{i\in\mathcal{N}}\|x_i(t)\|_2 \\
&\leq \epsilon \sum_{i\in\mathcal{N}}(1-\delta_i)a_i - \min_i(1-\delta_i)a_i\|x(t)\|_2
\end{aligned}$$

which means that the NCS is globally uniformly ultimately bounded and the invariant set is $\left\{ x \in \mathbb{R}^{nN} \mid \|x\|_2 \leq \frac{\epsilon \sum_{i\in\mathcal{N}}(1-\delta_i)a_i}{\min_i(1-\delta_i)a_i} \right\}$. □

5 Event-Triggering with Data Dropouts

In the previous sections, we did not consider the occurrence of data dropouts. In other words, whenever a broadcast release is triggered, the local state of the related subsystem will be sampled and transmitted to its neighbors successfully. In this section, we take data dropouts into account, which frequently happen in NCS. We assume that data dropouts only happen when the sampled states are sent to the controllers through the network. In other words, there is no dropout when sending controller outputs to the subsystems. This is usually true when each subsystem and its controller are wired together.

Let us take a look at what happens in the system when a data package is lost. We first consider those network protocols where the subsystem will be notified if transmission fails, such as Transmission Control Protocol (TCP). Using such protocols, when data dropouts happens, the subsystem just needs to keep sending the newly sampled state unless it is transmitted successfully. Also, the local triggering event will not be updated until transmission succeeds.

A more interesting thing happens with the network using those protocols where the subsystem will not be notified when transmission fails, such as User Datagram Protocol (UDP). In that case, when the hardware detector located at subsystem i detects the occurrence of the local event, the local state will be sampled and ready to be transmitted to its neighbors through the channel. At the same time, the event will be updated from k to $k+1$ with the newly sampled state. Once the transmission fails (in other words, the sampled state is lost), the controllers will not receive the sampled state. So the control inputs will not be updated. Notice that in this case, the local event will be updated, but the control inputs will not.

In the following discussion, we intend to address the allowable number of data dropouts in such NCS (UDP) with the guarantee of stability. In fact, we provide a decentralized approach that enables each subsystem to locally identify the largest number of its successive data dropouts that the subsystem can tolerate. The idea is to have events happen earlier than the violation of the inequality in equation (22) so that even if some data is lost, equation (22) can still be satisfied.

Before we introduce the results, we need to define two different types of releases: the triggered release, $\hat{r}_j^i$, and the successful release, r_k^i. $\hat{r}_j^i$ is the time when the jth broadcast of subsystem i is released (but not necessarily transmitted successfully). r_k^i is the time when the kth successful broadcast of subsystem i is released. Obviously, $\{r_k^i\}_{k=1}^{\infty}$ is a subsequence of $\{\hat{r}_j^i\}_{j=1}^{\infty}$. For notational convenience, we define $\hat{e}_j^i : \mathbb{R} \to \mathbb{R}^n$ as $\hat{e}_j^i(t) = x_i(t) - x_i(\hat{r}_j^i)$.

Theorem 4. *Consider the NCS in equation (2). Assume that equation (7), (8), (9), (15) hold. If, for any $i \in \mathcal{N}$ and some $\delta_i \in (0,1)$, the next broadcast release time is triggered by the violation of*

$$\|\hat{e}_j^i(t)\|_2 \leq \hat{\delta}_i c_i \|x_i(\hat{r}_j^i)\|_2 \tag{29}$$

for some $\hat{\delta}_i \in (0, \delta_i)$, the kth successful broadcast finishing time, f_k^i, satisfies

$$f_k^i - r_k^i \leq \min\left\{ \frac{(1-\delta_i)c_i}{p_i}\|x_i(r_{k-1}^i)\|_2, \frac{\hat{\delta}_i c_i(1-\delta_i c_i)}{p_i}\|x_i(r_{k-1}^i)\|_2 \right\}, \quad (30)$$

and the largest number of successive data dropouts, $n_i \in \mathbb{Z}$, satisfies

$$n_i \leq log_{(1+\hat{\delta}_i c_i)}(1+\delta_i c_i) - 1 \quad (31)$$

then the NCS is still asymptotically stable.

Proof. Consider subsystem i over the time interval $[r_k^i, f_{k+1}^i)$. For notational convenience, we assume $r_k^i = \hat{r}_0^i < \hat{r}_1^i < \cdots < \hat{r}_{n_i}^i < \hat{r}_{n_i+1}^i = r_{k+1}^i$. Since the hypotheses in lemma 1 hold, we have $\|e_k^i(t)\|_2 \leq \hat{\delta}_i c_i \|x_i(r_k^i)\|_2$ for all $t \in [r_k^i, f_k^i)$ and all $k \in \mathbb{N}$. Since $\|e_k^i(\hat{r}_1^i)\|_2 = \hat{\delta}_i c_i \|x_i(r_k^i)\|_2$, we have $f_k^i \leq \hat{r}_1^i$, namely that subsystem i does not release broadcasts during $[r_k^i, f_k^i)$.

Consider $\|e_k^i(t)\|_2$ for any $t \in [\hat{r}_j^i, \hat{r}_{j+1}^i)$. We have

$$\|e_k^i(t)\|_2 = \|x_i(t) - x_i(r_k^i)\|_2 \leq \sum_{l=0}^{j-1} \|x_i(\hat{r}_{l+1}^i) - x_i(\hat{r}_l^i)\|_2 + \|x_i(t) - x_i(\hat{r}_j^i)\|_2$$

for $\forall t \in [\hat{r}_j^i, \hat{r}_{j+1}^i)$. Applying equation (29) into the preceding equation yields

$$\|e_k^i(t)\|_2 \leq \sum_{l=0}^{j} \hat{\delta}_i c_i \|x_i(\hat{r}_j^i)\|_2 \quad (32)$$

for all $t \in [\hat{r}_j^i, \hat{r}_{j+1}^i)$. Therefore,

$$\|e_k^i(t)\|_2 \leq \sum_{l=0}^{n_i} \hat{\delta}_i c_i \|x_i(\hat{r}_l^i)\|_2 \quad (33)$$

holds for all $t \in [r_k^i, r_{k+1}^i)$.

Because $\|\hat{e}_j^i(\hat{r}_{j+1}^i)\|_2 = \|x_i(\hat{r}_{j+1}^i) - x_i(\hat{r}_j^i)\|_2 = \hat{\delta}_i c_i \|x_i(\hat{r}_j^i)\|_2$, we have

$$\|x_i(\hat{r}_{j+1}^i)\|_2 \leq (1+\hat{\delta}_i c_i)\|x_i(\hat{r}_j^i)\|_2$$

and therefore

$$\|x_i(\hat{r}_{j+1}^i)\|_2 \leq (1+\hat{\delta}_i c_i)^{j+1}\|x_i(\hat{r}_0^i)\|_2 = (1+\hat{\delta}_i c_i)^{j+1}\|x_i(r_k^i)\|_2 \quad (34)$$

for $j = 0, 1, 2, \cdots, n_i$. Applying equation (34) into (33) yields

$$\|e_k^i(t)\|_2 \leq \sum_{l=0}^{n_i} \hat{\delta}_i c_i (1+\hat{\delta}_i c_i)^l \|x_i(r_k^i)\|_2 = \left((1+\hat{\delta}_i c_i)^{n_i+1} - 1\right)\|x_i(r_k^i)\|_2 \quad (35)$$

for all $t \in [r_k^i, r_{k+1}^i)$. By equation (31), we know $(1+\hat{\delta}_i c_i)^{n_i+1} - 1 \leq \delta_i c_i$. Therefore, equation (35) implies $\|e_k^i(t)\|_2 \leq \delta_i c_i \|x_i(r_k^i)\|_2$ for all $t \in [r_k^i, r_{k+1}^i)$. Since the hypotheses in corollary 2 are satisfied, we conclude that the NCS is asymptotically stable. □

Remark 6. By equation (31), we know the maximal allowable number of each subsystem's successive data dropouts can be identified locally, depending on the selection of c_i, δ_i, and $\hat{\delta}_i$. If subsystem i wants the maximal allowable number of data dropouts to be large, $\hat{\delta}_i$ must be small enough. In general, however, small $\hat{\delta}_i$ will result in short broadcast periods. Therefore, there is a tradeoff between the maximal allowable number of data dropouts and the broadcast periods.

Similar to corollary 3, we have the following result for the case with fixed transmission deadlines.

Corollary 4. *Assume that all hypotheses in theorem 4 are satisfied except that equation (30) is replaced by*

$$f_k^i - r_k^i \leq \min\left\{\frac{(1-\delta_i)c_i\epsilon}{p_i}, \frac{\hat{\delta}_i c_i \epsilon}{p_i}\right\} \tag{36}$$

for some small positive constant $\epsilon > 0$, then the NCS is still globally uniformly ultimately bounded with an ultimate bound $\frac{\epsilon \sum_{i\in\mathcal{N}}(1-\delta_i)a_i}{\min_i(1-\delta_i)a_i}$, where a_i is defined in equation (10).

Proof. Following a similar analysis to the proof in theorem 4, we have $\|e_k^i(t)\|_2 \leq \delta_i c_i \|x_i(r_k^i)\|_2$ for all $t \in [r_k^i, r_{k+1}^i)$. Since the hypotheses in corollary 3 are satisfied, we conclude that the NCS is globally uniformly ultimately bounded. □

Based on the preceding results, we are able to present the decentralized event-triggering scheme.

Decentralized Event-Triggering Scheme

1. Select positive constants $\beta_i, \kappa_i \in \mathbb{R}^+$ for $i = 1, \cdots, N$;
2. For subsystem i,
   (1) Find $V_i : \mathbb{R}^n \to \mathbb{R}$, $\alpha_i \in \mathbb{R}^+$, and $\gamma_i : \mathbb{R}^{n|Z_i|} \to \mathbb{R}^{m_i}$ satisfying equation (8), (9);
   (2) Compute p_i satisfying equation (15);
   (3) Select $\rho_i \in (0,1)$ and compute c_i based on equation (13);
   (4) Select $\delta_i \in (0,1)$, $\hat{\delta}_i \in (0, \delta_i)$ and use the violation of the inequality in equation (29) to trigger the broadcast release;
   (5) Predict the deadline for the delay in the kth successful broadcast of subsystem i ($f_k^i - r_k^i$) at r_{k-1}^i by equation (30) or equation (36);
   (6) Identify the maximal allowable number of successive data dropouts by equation (31).

6 An Illustrative Example

This section presents simulation results demonstrating the decentralized event-triggering scheme. The system under study is a collection of coupled carts (figure 2), which are coupled together by springs. The ith subsystem state is the

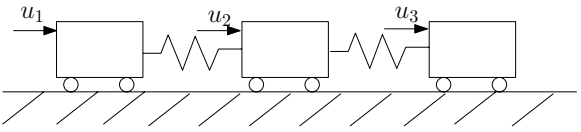


Fig. 2. Three carts coupled by springs

Table 1. Results on Running a Decentralized Event-Triggered Networked System

	Subsystem 1	Subsystem 2	Subsystem 3
Maximal Allowable Number of Successive Data Dropouts	2	3	2
Predicted Deadline	2.226×10^{-4}	1.811×10^{-4}	2.226×10^{-4}
Number of Broadcasts Release	153	229	155
Number of Successful Broadcasts	50	56	50
Average Period of Broadcasts	0.0523	0.0349	0.0516
Average Period of Successful Broadcasts	0.1600	0.1429	0.1600

vector $x_i = \begin{bmatrix} y_i \ \dot{y}_i \end{bmatrix}^T$ where y_i is the ith cart's position. We assume that at the equilibrium of the system, all springs are unstretched. The state equation for the ith cart is $\dot{x}_i = A_i x_i + B_i u_i + H_{i,i-1} x_{i-1} + H_{i,i+1} x_{i+1}$ where $A_i = \begin{bmatrix} 0 & 1 \\ -\mu_i k & 0 \end{bmatrix}, B_i = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, H_{ij} = \begin{bmatrix} 0 & 0 \\ \nu_{ij} k & 0 \end{bmatrix}$.

In the preceding equation, we have $k = 5$ is the spring constant, $\mu_1 = \mu_N = 1$ and $\mu_i = 2$ for $i = 2, \cdots, N-1$. Also $\nu_{ij} = 1$ for $i \notin \{1, N\}$ and $j \in \{i-1, i+1\}$ and $\nu_{12} = \nu_{N,N-1} = 1$. Otherwise, $\nu_{ij} = 0$.

The control input of subsystem i is

$$u_i = K_i \hat{x}_i + L_{i,i-1} \hat{x}_{i-1} + L_{i,i+1} \hat{x}_{i+1}, \tag{37}$$

where $K_1 = K_N = \begin{bmatrix} -4 & -6 \end{bmatrix}$, $K_i = \begin{bmatrix} 1 & -6 \end{bmatrix}$ for $i = 2, \cdots, N-1$, and $L_{i,i-1} = L_{i,i+1} = \begin{bmatrix} -5 & 0 \end{bmatrix}$ except that $L_{10} = L_{N,N+1} = 0$.

We first considered the case with $N = 3$. According to the decentralized event-triggering scheme, we obtained $c_2 = 0.3622$ and $c_1 = c_3 = 0.4451$. The initial state x_{i0} of subsystem i was randomly generated satisfying $\|x_{i0}\|_2 \leq 1$. We set $p_i = 20$, $\epsilon = 0.1$, $\delta_i = 0.9$, and $\hat{\delta}_i = 0.2$. We ran the event-triggered system for 8 seconds with the assumption that the number of successive data dropouts are the same as the maximal allowable number and the delay is equal to the deadline. The simulation results show that the system is asymptotically stable, but not globally uniformly ultimately bounded as we stated in corollary 4. This might be because of the special network topology used in this simulation (linear and $D_i = Z_i$). Another possible explanation is that the decentralization leads to the conservativeness of the theoretical results. The data of this simulation is listed in table 1.

We then examine the relationship between the maximal allowable number of successive data dropouts, n_i, and the predicted deadline. In particular, we

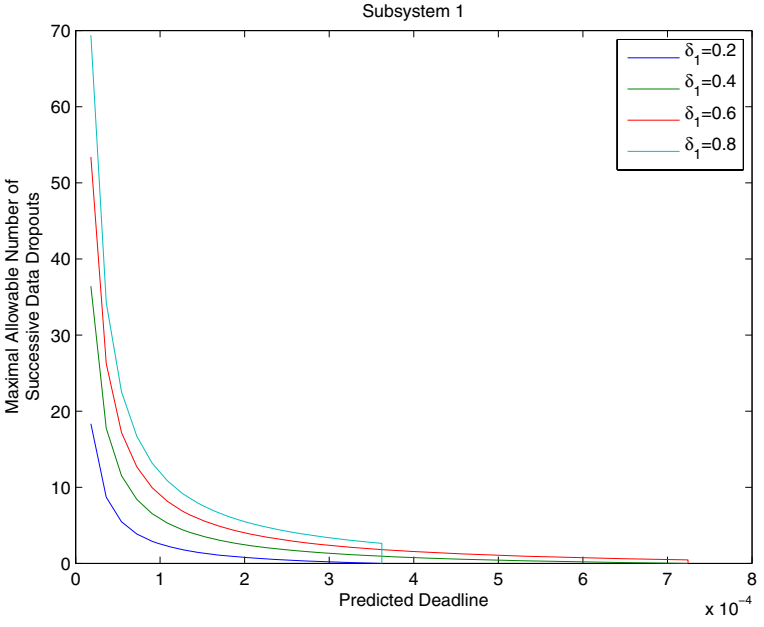


Fig. 3. Maximal allowable number of successive data dropouts versus the deadline

studied subsystem 1. We set $\delta_1 = 0.2, 0.4, 0.6, 0.8$ and $\hat{\delta}_1$ from 0.01 to δ_1. The other parameters remain the same. The resulting changes in n_1 and the deadline are shown in figure 3, where each pair of n_1 and the deadline is associated with a pair of $(\delta_1, \hat{\delta}_1)$. We may see from the plot that as n_1 increases, the predicted deadline decreases. That is because large n_1 suggests tiny $\hat{\delta}_1$ and large δ_1, which results in short deadline according to equation (36).

7 Conclusions

This paper studies distributed NCS with data dropouts and transmission delays. We propose a decentralized event-triggering scheme for such systems. This scheme is completely decentralized, which means that a subsystem's broadcast decisions are made using its local sampled data, the maximal allowable transmission delay of a subsystem's broadcast is predicted based on the local information, a subsystem locally identifies the maximal allowable number of its successive data dropouts, and the designer's selection of the threshold only requires information about an individual subsystem and its immediate neighbors. Our analysis applies to nonlinear continuous systems. With the assumption that the number of each subsystem's successive data dropouts is less than the bound identified by that subsystem, if the bandwidth of the network is limited so that the transmission delays are always greater than a positive constant, the resulting system is globally uniformly ultimately bounded using our scheme; otherwise, the resulting system is asymptotically stable.

References

1. Lian, F., Moyne, J., Tilbury, D.: Network design consideration for distributed control systems. IEEE Transactions on Control Systems Technology 10, 297–307 (2002)
2. Krtolica, R., Ozguner, U., Chan, H., Goktas, H., Winckelman, J., Liubakka, M.: Stability of linear feedback systems with random communication delays. In: Proceedings of American Control Conference (1991)
3. Wong, W.S., Brockett, R.W.: Systems with finite communication bandwidth constraints - Part II: Stabilization with limited information feedback. IEEE Transactions on Automatic Control 44, 1049–1053 (1999)
4. Walsh, G., Ye, H., Bushnell, L.: Stability analysis of networked control systems. IEEE Transactions on Control Systems Technology 10, 438–446 (2002)
5. Nesic, D., Teel, A.: Input-output stability properties of networked control systems. IEEE Transactions on Automatic Control 49, 1650–1667 (2004)
6. Carnevale, D., Teel, A., Nesic, D.: Further results on stability of networked control systems: a lyapunov approach. IEEE Transactions on Automatic Control 52, 892–897 (2007)
7. Wang, X., Lemmon, M.: Event-triggered Broadcasting across Distributed Networked Control Systems. In: American Control Conference (2008)
8. Wang, X., Lemmon, M.: Decentralized Event-triggering Broadcast over Networked Systems. In: Egerstedt, M., Mishra, B. (eds.) HSCC 2008. LNCS, vol. 4981, pp. 674–678. Springer, Heidelberg (2008)
9. Tabuada, P.: Event-triggered real-time scheduling of stabilizing control tasks. IEEE Transactions on Automatic Control 52, 1680–1685 (2007)
10. Wang, X., Lemmon, M.: Self-triggered feedback control systems with finite-gain L_2 stability. To appear in IEEE Transactions on Automatic Control (2009)
11. Mazo, M., Tabuada, P.: On event-triggered and self-triggered control over sensor/actuator networks. To appear in Proceedings of the 47th Conference on Decision and Control (2008)
12. Ling, Q., Lemmon, M.: Soft real-time scheduling of networked control systems with dropouts governed by a Markov Chain. In: American Control Conference (2003)
13. Kawka, P., Alleyne, A.: Stability and Performance of Packet-Based Feedback Control over a Markov Channel. In: American Control Conference (2006)

Specification and Analysis of Network Resource Requirements of Control Systems*

Gera Weiss[1], Sebastian Fischmeister[2], Madhukar Anand[1], and Rajeev Alur[1]

[1] Dept. of Computer and Information Science, University of Pennsylvania, USA
[2] Dept. of Electrical and Computer Engineering, University of Waterloo, Canada

Abstract. We focus on spatially distributed control systems in which measurement and actuation data is sent via a bus shared with other applications. An approach is proposed for specifying and implementing dynamic scheduling policies for the bus with performance guarantees. Specifically, we propose an automata-based scheduler which we automatically generate from a model of the controlled plant and the controller. We show that, in addition to ensuring performance, our approach allows adjustments to dynamic conditions such as varying disturbances and network load. We present a full development path from performance specifications (exponential stability) to a control design and its implementation using Controller Area Network (CAN).

1 Introduction

As control systems grow in both size and complexity, so does the need to spatially distribute control equipment such as sensors, actuator and computational devices. In recent years, implementations of distributed control systems are shifting from traditional hard-wired architectures, where each device is connected via a dedicated wire, to networked architectures, where control data is sent via shared communication buses (e.g., in the automotive [12] and aviation [20] industries, and for process control [24]). While, shared communication buses reduce costs and allow flexible architectures, they also introduce the problem of resource contention and require scheduling mechanisms to resolve them [19, 25].

Existing approaches to bus scheduling in control applications rely on static (periodic) schedules designed to assure performance in worst-case conditions [10, 18, 23]. The main disadvantage of static schedules, in our context, is that they lack a mechanism to adapt to changing conditions. This often leads to trading off average for worst-case performance.

In this paper, we propose a mechanism for generating schedules for shared buses such that a specified stability rate is guaranteed. We use guarded automata as a tool for formalizing the effect of bus scheduling on performance and as a mechanism for scheduling the network such that stability is guaranteed.

A scheduling approach is proposed that provides good performance both in average and worst-case conditions. We show that automata based scheduling allows the schedule to react to dynamic conditions such as the output of the plant or the load on the

* This work was partially supported by NSF CNS 0524059, NSF CPA 0541149, and NSERC DG 357121-2008.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 381–395, 2009.

network and still guarantee high-level requirements such as stability. In particular, we demonstrate how, with our approach, the bus is only used when needed and, by that, good average performance can be obtained together with worst-case guarantees.

While our approach may apply also to other architectures, we concentrate on systems where one control loop shares a communication bus with background applications that use it for non–real-time communication. In this architecture, the network scheduler needs to assign communication bandwidth to the background applications while maintaining the specified stability performance. Focusing on this architecture allows us to present a holistic approach that begins with stability specification and ends with an implementation. The presented approach can be generalized to multiple control loops and to a star architecture where spatially distributed plants are controlled with a central controller.

The remaining of the paper consists of the following parts: in Section 2 we formally define the technical problem that this paper is about and sections 3-6 detail the steps towards its solution. Section 7 puts our work in context with related work. And, in Section 8, we draw conclusions from the results and outline potential next steps.

2 Shared Bus for Control and Background Traffic

Consider the system depicted in Figure 1 below, where the sensor and the actuator of a control system are spatially distributed and a shared communication bus is used to pass information from a computer processor near the sensor to a processor near the actuator.

In this system, a communication bus is used both to close a control loop and for non real-time background applications. We assume TDMA (time division multiple access) arbitration, where messages are transmitted in separated time slots of fixed length (time-triggered message generation). The control loop can be modeled by a discrete-time control system where the sampling interval is the time slot of the network. To simplify notations and avoid orthogonal complications, we consider a single-input, single-output linear time-invariant plant.

Assume that the processor near the sensor has priority over the bus, i.e., when it decides to send data, all other messages are preempted. Suppose a time-varying number

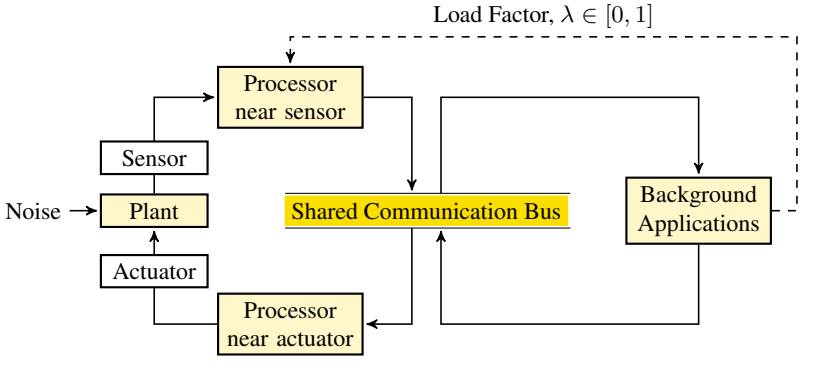


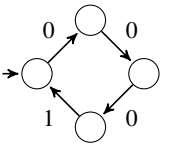
Fig. 1. A bus shared by a control loop and non real-time traffic

$\lambda \in (0, 1)$, called the load factor, is fed to the processor near the sensor. This number is an input parameter that models the fraction of bandwidth that the control system is asked to leave to background applications. The main problem addressed in this paper is how should the processor near the sensor decide when to send data, taking both λ and the output of the plant into account. Since decisions are taken online by devices with low computational power, we are especially interested in decision procedures with low online computational demands.

The scheduling problem can be solved by a static, time-triggered cyclic executive that assigns resources to either the control loop or the background applications [16, 10, 13]. This approach is depicted in Figure 2. Figure 2(a) shows an implementation of the approach using dispatch tables. The table describes, for specific time slots, which consumer uses the resource. This particular schedule shows that the background applications use the resource for the first three steps and the control loop uses the resource in the fourth step.

Time	*Consumer*
1	background
2	background
3	background
4	control loop

(a) Dispatch table



(b) Automaton

Fig. 2. Two ways for encoding a static schedule

We can also encode such a cyclic executive using automata (see Figure 2(b)). The language of this automaton is the set of all sequences over $\{0, 1\}$ with 1 at every fourth position. The symbol 1 means that the control loop gets the network resource and 0 means that any of the background applications gets it. Note that implementing the scheduling policy of an automaton or a dispatch table can be done with a lightweight decision procedure requiring low (constant) time and memory.

This type of static scheduling is a common practice but it is also often wasteful. Static scheduling via dispatch tables and static automata is useful because of analyzability and ease of implementation. However, static scheduling often uses more resources than necessary, because many applications do not require a fixed sampling frequency to assure performance. For example, consider a system with sporadic disturbance bursts. In this case, a periodic sensor update often provides no additional information to the processor near the actuator and therefore is a waste of network resources. An improved version will only send measurements if the plant's output exceeds some threshold discrepancy, as we show in the following example.

On the other extreme, bus arbitration could also be decided by a tailored, fully dynamic software. The main problem with the latter approach is that it is not clear how to analyze and systematically design such software. In this paper, we propose a mid-way between fully dynamic code and dispatch tables. Using guarded automata, we propose a scheduling mechanism that allows analyzability and lightweight implementation (as static scheduling) with adaptability and efficiency (as dynamic scheduling).

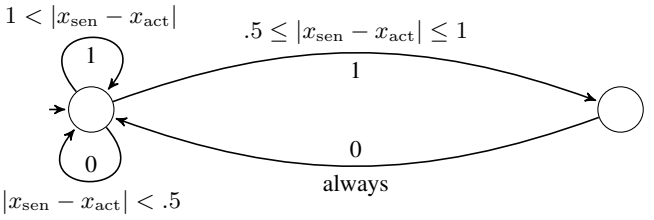


Fig. 3. A guarded (dynamic) schedule

The following example illustrates how guarded automata can be used for scheduling. Note that, while the example is an ad hoc (manually designed) automaton, the methodology we are proposing is automatic generation of automata using the construction described in this paper.

For the system depicted in Figure 1, assume that the processor near the sensor maintains an estimate of the plant state denoted by x_{sen}, and the processor near the actuator maintains its own estimate (based on the information available to it) that we denote by x_{act}. Consider the scheduling scheme depicted in Figure 3. In words: the scheduling decision is based on the difference $|x_{\text{sen}} - x_{\text{act}}|$. If it lies above 1, then data is transmitted. If it lies below .5, then the processor near the sensor will not transmit and leave the slot for the background applications. If $|x_{\text{sen}} - x_{\text{act}}|$ lies between .5 and 1, the processor near the sensor will transmit a reading once and then pause in the next step.

Clearly, this scheduling scheme is more expressive than cyclic scheduling using dispatch tables or unguarded automata. Still, unlike general dynamic techniques, the model is simple enough for formal analysis of system properties such as exponential stability. In this paper we focus on generating guarded automata of the form depicted in Figure 1 that guarantee high-level requirements of the control application. Specifically, we investigate automata generation for exponential stability requirements.

In the following sections we propose steps towards synthesizing a scheduler that guarantees exponential stability and uses the bus only when needed or when the parameter λ is small (low background traffic). The proposed methodology is described in four steps, each described in a separate section and summarized as follows. The first step, described is Section 3, is a construction of an automaton that specifies unstable runs of the control loop. The second step, described in Section 4, is to transform the specification automaton to an executable state machine that identifies when using the bus is critical for ensuring stability. The third step, described in Section 5, is to implement the scheduling scheme with a distributed bus arbitration mechanism. The fourth step, described in Section 6, is to test and validate the mechanism by implementing a switched control strategy and a scheduling scheme that combines the parameter λ with the executable state machine.

3 Step I: Specification Automaton

As a first step towards solving the problem presented in Section 2, we propose an automaton that specifies unstable runs, as follows.

We use switched systems (see e.g. [14]) to model the system depicted in Figure 1. The switched system has two modes: (1) a mode that models the transformation of the state variables when the processor near the sensor is using the bus and (2) a mode that models the transformation when the bus is not used by the processor near the sensor. See [1,2] and the examples given in Section 6 for more details about this modeling approach.

Formally, let $A_0, A_1 \in \mathbb{R}^{n\times n}$ and $c_0, c_1 \in \mathbb{R}^{1\times n}$ be such that

$$\begin{aligned} x(t+1) &= A_{w(t)}x(t); \\ y(t) &= c_{w(t)}x(t), \end{aligned} \tag{1}$$

models the dynamics of the control loop depicted in Figure 1, where $x(t) \in \mathbb{R}^n, y(t) \in \mathbb{R}$ are the state and the observation at time t, respectively. The infinite word $w \in \{0,1\}^\omega$ (called the switching sequence) is such that the processor near the sensor sends data to the processor near the actuator at time t iff $w(t) = 1$. A run of the system is a solution of the equations.

As a performance measure we choose exponential stability. The standard definition of exponential stability requires that behaviours converge to the origin faster than a given exponentially decaying function. In [1], a system is defined to be (ρ, l)-exponentially-stable if in every l time units the distance to the origin (norm) decreases by a factor of ρ. In this paper we add two more parameters. Specifically, for the parameters $0 < \rho \leq 1, l \in \mathbb{N}$ and $0 < \varepsilon < \delta$,

Definition 1. A run of the system (1) is $(\rho, l, \varepsilon, \delta)$-exponential-stable if $\varepsilon < |x(t)| < \delta \implies |x(t+l)| < \rho|x(t)|$ for every $t \in \mathbb{N}$.

Namely, a run is exponentially stable if any state, in the δ-ball and not in the ε-ball around the origin, gets closer to the origin by a factor ρ, every l steps.

In the following definition, a regular language (over an infinite alphabet) is used to specify runs of the system that are not exponentially stable.

Definition 2. A language over the alphabet $\Sigma = \{0,1\} \times \mathbb{R}$ is a $(\rho, l, \varepsilon, \delta)$-safe-monitor for the system (1) if for every run that is not $(\rho, l, \varepsilon, \delta)$-exponentially-stable there exists $k \in \mathbb{N}$ such that the word $\langle w(1), y(1)\rangle \cdots \langle w(k), y(k)\rangle$ is in the language.

Namely, a safe-monitor is an automaton such that if an accepting state is not reached (when the outputs and modes of the plants are fed as inputs to the automaton) then the run is exponentially stable with the required parameters. If the accepting state is reached then the run may not be safe. The approach that we are proposing in this paper is to avoid accepting states and, by that, assure, for instance, exponential stability.

In the rest of this section, we give a construction of a non-deterministic automaton whose accepted language is a $(\rho, l, \varepsilon, \delta)$-safe-monitor called $(\rho, l, \varepsilon, \delta)$-specification-automaton (because it specifies $(\rho, l, \varepsilon, \delta)$-exponentially-stable runs). Note that, while we propose a specific construction, it is not the only $(\rho, l, \varepsilon, \delta)$-specification-automaton. However, we are not assuming the specific construction in the rest of the paper (only the properties given in Definition 2).

Construction 3. Let $0 < \rho \leq 1, l \in \mathbb{N}$ and $0 < \varepsilon < \delta$ be the required exponential stability parameters.

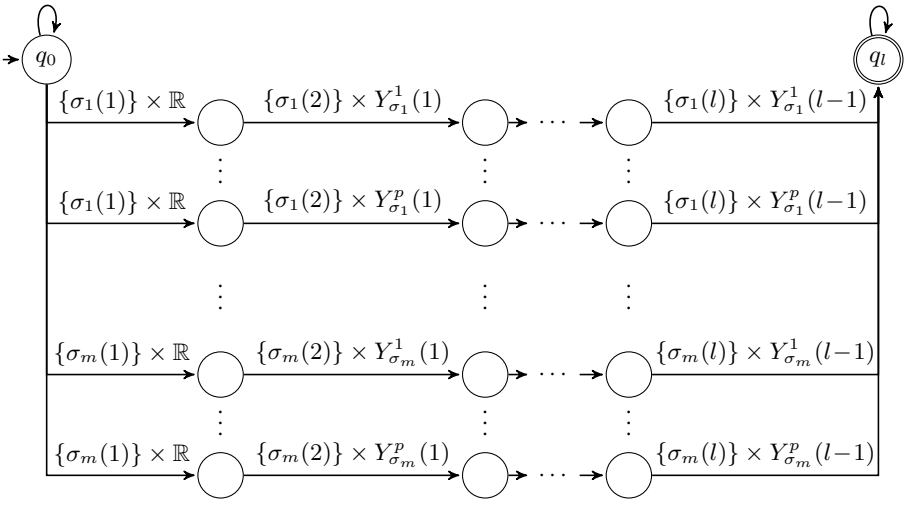


Fig. 4. A non-deterministic automaton over the alphabet $\{0,1\} \times \mathbb{R}$ such that if no prefix of $\langle w(1), y(1)\rangle, \langle w(2), y(2)\rangle, \ldots$ is accepted then any run of the system (1) with switching signal $w(1), w(2), \ldots$ and outputs $y(1), y(2), \ldots$ is exponentially stable

- For each $\sigma = \sigma(1) \cdots \sigma(l) \in \{0,1\}^l$:
  - Let B_σ be the set of all $x \in \mathbb{R}^n$ such that $|A_{\sigma(l)} \cdots A_{\sigma(1)} x| \geq \rho |x|$ and $\varepsilon \leq |x| \leq \delta$. In words, B_σ is the set of all vectors, in the δ–ball but not in the ε–ball, whose distance to the origin does not shrink by a factor of ρ when the transformation $A_{\sigma(l)} \cdots A_{\sigma(1)}$ is applied. We focus on this set because, to guarantee exponential stability, we can make sure that whenever the current state is in B_σ the switching signal for the next l steps is not σ. Let $B^1_\sigma, \ldots, B^p_\sigma$ be a finite cover of B_σ by compact convex sets.
  - Let $Y^j_\sigma(k) := \{o(k)x : x \in B^j_\sigma\}$ where $o(k) := c_{\sigma(k)} A_{\sigma(k-1)} \cdots A_{\sigma(1)}$. In words, $Y^j_\sigma(k)$ is the set of possible outputs that we may observe at time $t+k-1$ if $x(t)$ is in B^j_σ and $w(t) \cdots w(t+l-1) = \sigma$. Note that $Y^j_\sigma(k)$ is an interval whose bounds can be effectively computed, because B^j_σ is compact and convex.
- Let $\sigma_1, \ldots, \sigma_m$ be the set of words of length l such that $|A_{\sigma_i(l)} \cdots A_{\sigma_i(1)}| \geq \rho$, i.e. the words such that $B_{\sigma_i} \neq \emptyset$.
- For the switched system (1), we define a non-deterministic automaton (depicted in Figure 4). The states of the automaton are $\{q_{i,j}(k) \colon i = 1, \ldots, m, j = 1, \ldots, p$ and $k = 1, \ldots, l-1\} \cup \{q_0, q_l\}$. The transition from q_0 to every $q_{i,j}(1)$ is guarded by the condition $w(t) = \sigma_i(1)$. For, $k = 2, \ldots, l-1$, the transition from $q_{i,j}(k-1)$ to $q_{i,j}(k)$ is guarded by the condition $w(t) = \sigma_i(k) \wedge y(t-1) \in Y^j_{\sigma_i}(k-1)$. The transition from every $q_{i,j}(l-1)$ to q_l is guarded by $w(t) = \sigma_i(l) \wedge y(t-1) \in Y^j_{\sigma_i}(l-1)$. Finally, the self loops on states q_0 and q_l are unconditioned. The initial state is q_0 and the only accepting state is q_l.
- By construction, if no run of the automaton gets to $F = \{q_l\}$ at time t then the product of the last l matrices takes $x(t-l)$ closer to the origin by a factor of

at least ρ; assuming that $\varepsilon < |x(t)| < \delta$. In particular, since this is true for every t, we get that the system (1) is exponentially stable.

The only non-constructive step in the above description is covering B_σ by a finite number of compact convex sets (where $\sigma = \sigma(1)\cdots\sigma(l)$ is an arbitrary word). Let $A_\sigma = A_{\sigma(l)}\cdots A_{\sigma(1)}$. Towards a cover, we explore the geometry of the set B_σ, as depicted in Figure 5. Consider the sphere $S_\delta = \{x \in \mathbb{R}^n : |x| = \delta\}$. The linear transformation A_σ maps this sphere onto an ellipsoid $E = \{A_\sigma x : x \in S_\delta\}$. The intersection $B_\sigma \cap S_\delta$ can be visualized as two symmetric arcs on the sphere (the part of the sphere that is mapped to the part of E that is outside of the $\rho\delta$-sphere). By linearity, $B_\sigma = \{\lambda x : \lambda \in [\varepsilon/\delta, 1), x \in B_\sigma \cap S_\delta\}$ which can be visualized as a pair of two symmetric trimmed cones. Let a be the largest semi-axes of the ellipsoid E and h be such that $A_\sigma h = a$. Then, the cover is $B^1_\sigma = \{x \in B_\sigma : h^T x \geq \gamma\}$ and $B^2_\sigma = \{x \in B_\sigma : h^T x \leq -\gamma\}$ where γ is the largest number such that these two sets cover B_σ. In practice, one can compute h using singular value decomposition of A_σ.

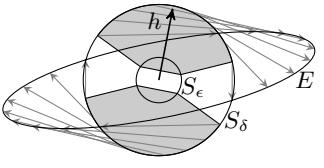


Fig. 5. A coverage of B_σ by two compact convex sets, B^1_σ and B^2_σ (grayed)

Example 1. Consider the system (1) where $A_0 = \begin{pmatrix} -1 & 1/2 \\ 0 & 1 \end{pmatrix}$, $A_1 = \begin{pmatrix} 1/2 & -1 \\ 1 & 0 \end{pmatrix}$ and $c_0 = c_1 = \begin{pmatrix} 1/2, 1/2 \end{pmatrix}$. Let $\rho = 1$, $l = 10$, $\varepsilon = 1/10$, and $\delta = 1$. Assume that the switching signal is zero for 10 consecutive steps. A stability monitor that can only base its decision on the switching signal (as the one considered e.g. in [1]) must accept the run as potentially unstable because the norm of A_0 to the power 10 is bigger than one (which means that there exists x such that $\|A_0^{10}x\| > \|x\|$). However, a closer examination reveals that we only have a problem if the polar angle $\alpha = \tan^{-1}(x_2/x_1)$ of the initial state satisfies $-\cos^{-1}\left(-38/\sqrt{1973}\right) \leq \alpha \leq -\cos^{-1}\left(18/\sqrt{2173}\right)$ or $\cos^{-1}\left(38/\sqrt{1973}\right) \leq \alpha \leq \cos^{-1}\left(-18/\sqrt{2173}\right)$. The first case, corresponding to the set $B^1_{\sigma_1}$ in the construction (considering also $\varepsilon < \|x\| < \delta$), is depicted in Figure 6. Figure 6(a) shows an over approximation of the bad set based on the first two observation, $y(1) = c_0 x(0), y(2) = c_0 A_0 x(0)$. Figure 6(b) shows an over approximation of the bad set based on the first five observation, $y(1) = c_0 x(0), \ldots, y(5) = c_0 A_0^4 x(0)$. In the automaton depicted in Figure 4, these are the initial states for which the automaton will get to the third and sixth states on the first horizontal path, respectively. More generally, the first horizontal path of the automaton corresponds to the "non deterministic guess" that the next 10 values of the switching signal are going to be zeroes. The guards are designed such that the path is abandoned if this guess turns to be wrong or the observations show that the initial state was not in $B^1_{\sigma_1}$.

(a) Inequalities over $y(1)$ and $y(2)$ (b) Inequalities over $y(1), \ldots, y(5)$

Fig. 6. Over-approximations of bad initial states by linear inequalities over the observations

4 Step II: Executable State Machine

The specification automaton, described in the preceding section allows to detect errors, but is not directly applicable for scheduling. In this section we use the specification automaton to obtain an executable state machine that identifies times where sending a message from the processor near the sensor to the processor near the actuator is essential for keeping the control-loop stable.

Towards such an executable state machine, we define some of the states of the specification automaton as `bad` states. Specifically, q is marked as a `bad` state if there is a path $q = q(1), \cdots, q(k)$ (of arbitrary length k) from q to an accepting state and a sequence $y_1, \cdots, y_{k-1} \in \mathbb{R}$ such that $q(i+1) \in \delta(q(i), \langle 1, y_i \rangle)$ for each $i = 1, \ldots, k-1$ (where δ is the transition function).

Since every accepting state is a `bad` state, avoiding `bad` states ensures exponential stability. But, unlike the case for accepting states, we now have also the following property: if q is not a `bad` state then all the states in $\delta(q, \langle 1, y \rangle)$ are not `bad`, for any y. This property is useful for scheduling because it means that we can always avoid a transition to a `bad` state by scheduling mode 1. Note that we are assuming that the initial state is not `bad`. This assumption makes sense because if the initial state is already `bad`, we will not get the required stability even if the bus is dedicated to the control loop. Practically, we are assuming that the control design is such that the requirements are met when the bus is always available.

An executable state machine is obtained by simulating the automaton. For a finite trace $T = \langle w(1), y(1) \rangle, \ldots, \langle w(t-1), y(t-1) \rangle$ of the system (1), the state of the state-machine, at time t, is the set $\delta^*(q_0, T) \subseteq Q$ consisting of the end states of all runs of the automaton up to time t. We say that the state-machine is in a `must` state if choosing $w(t) = 0$ will make that state contain a `bad` state of the automaton. As the transition from a state that is not `bad` to a `bad` state is conditioned on $w(t) = 0$ (otherwise the source is also `bad`), we are guaranteed that we can avoid `bad` states by scheduling mode 1 whenever the executable state machine (described in the previous section) is in a `must` state.

Note that our approach can be directly generalized to any number of concurrent control loops. Imagine, for example, a second control loop (another triplet of sensor, actuator and plant) that shares the same bus for communicating data from sensor to actuator. In this case, the scheduler of each loop is going to get to a `must` state if it must send in one of the next two communication slots.

5 Step III: Stateful Priority Assignment

The executable state machine, described in the previous section, detects times when sending a message is essential (when the machine is in a `must` state), but it does so in a centralized manner. In this section we discuss an implementation of it using priority based bus arbitration.

For implementation, we propose the use of priority based bus arbitration, but, instead of assigning priorities to nodes or to message types, we propose to assign priorities to states of the scheduler. Specifically, the priority of a message from the processor near the sensor to the processor near the actuator is high when the scheduler is in a `must` state and low otherwise.

Assigning priorities to messages based on system state has not been considered so far, because the developer must assure that, at any point in time, each priority level is used by at most one node. In this paper, a formal model (automaton) assist the developer in guaranteeing this property for large, complex systems.

Using CAN. Controller Area Network (CAN) [6] is the most widespread priority based networking technology for control applications. It provides eleven or 29 bits for encoding an unsigned integer priority level for individual messages and requires two wires for the physical communication layer. One wire implements the dominant bits while the other implements recessive bits. Using a logical-AND between those two bits, the bus implements a bit-wise arbitration mechanism, which allows the winning node to continue sending its message during the arbitration.

The first step, towards applying CAN, is to annotate the transitions of the scheduler with priorities (highest and lowest), as follows: if the scheduler is in a `must` state at time t, then it will have the highest priority. All other transitions in the scheduler get assigned some value other than the highest priority level. The second step is to assign priorities to background applications. The specific priorities assigned the background applications and the sensor can be tweaked to fit bandwidth defined by λ (see Section 6).

Note that implementing our approach with CAN requires no extra hardware. With eleven bits, the control engineer can assign 2046 priority levels (one for broadcast), which means that the control application uses the priority levels `0x07FF` and `0x001`. This leaves plenty of additional priorities for background applications. Using 29 bits raises this range even further.

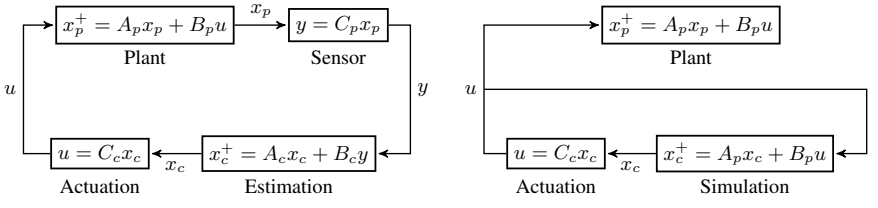
Our approach assigns different priorities to the same message depending on the message context—high priority if the message is important, low priority if it is unimportant. In priority-driven arbitration such as CAN, nodes that simultaneously access the bus must use different priorities, otherwise the bus arbitration will fail and cause data corruption. Our approach still follows this rule, because instead of assigning one priority level to a message, we assign two. Both priority levels are exclusively used for this message. A straightforward way to extend our approach is to share priorities among messages and use formal verification of the schedulers to guarantee that two schedulers never use the same message priority at the same time.

6 Step IV: Testing and Validation

In this section we revisit the initial example, provide more technical details on controller and scheduler designs and present some performance analysis that illustrates the advantages and disadvantages of our approach.

Controller Design. As a specific example, we examine how an LQG controller can be implemented with the architecture depicted in Figure 1, above, where a shared communication bus separates the processor near the sensor from the processor near the actuator. The control loop is designed as a switched system with the following two modes.

Figure 7(a) shows the feedback mode, active when the processor near the sensor sends data on the bus. The matrices A_p, B_p, C_p model the controlled plant and the matrices A_c, B_c, C_c are computed using a standard technique for LQG design (e.g., by MATLAB's [17] `lqg` command).



(a) Feedback Mode (b) Feedforward Mode

Fig. 7. Two modes of the control loop. The feedback mode, active when processor near the sensor sends data to the processor near the actuator, is a full LQG based feedback. In the feedforward mode, data is not sent. Instead, the processor near the actuator simulates the dynamics of the plant based on earlier data.

The second mode of the controller corresponds to times at which the processor near the sensor does not transmit its reading. In these times, the processor near the actuator simulates the dynamics of the plant. Figure 7(b) shows this second mode. A simulation block replaces the estimation block and the output of the plant remains unused (because data is not sent).

The composition of the system with the controller modes results in the closed-loop switched system described by equation (1), where $A_0 = \begin{pmatrix} A_p & B_p C_c \\ 0 & A_p + B_c C_p \end{pmatrix}$, $A_1 = \begin{pmatrix} A_p & B_p C_c \\ B_c C_p & A_c \end{pmatrix}$ and $x(t) = (x_p^T, x_c^T)^T$. The switching signal $w \in \{0,1\}^{\mathbb{N}}$ is such that $w(t)$ is one iff the processor near the sensor sends data to the processor near the actuator at time t.

As an example, consider the plant $\dot{x}_p = \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix} x_p + \begin{pmatrix} 1 \\ 0 \end{pmatrix} u, y = (0,1) x_p$. In MATLAB, the matrices A_0 and A_1 can be computed as follows: (1) get a discrete-time model using `c2d`, (2) compute an LQG compensator using `lqg`. The closed loop matrices obtained by this procedure (using $Ts = 1$ and $QXU = QWV = .1$) are:

$$A_1 = \begin{pmatrix} 0.568 & 0.432 & -0.357 & -0.339 \\ 0.432 & 0.568 & -0.142 & -0.134 \\ 0 & 0.412 & 0.210 & -0.319 \\ 0 & 0.461 & 0.291 & -0.028 \end{pmatrix} \text{ and } A_0 = \begin{pmatrix} 0.568 & 0.432 & -0.357 & -0.339 \\ 0.432 & 0.568 & -0.142 & -0.134 \\ 0 & 0 & 0.210 & 0.094 \\ 0 & 0 & 0.291 & 0.433 \end{pmatrix}.$$

The network is scheduled based on the difference between the output and the estimated output, i.e., $c_0 = c_1 = (0, 1, 0, -1)$.

Scheduling Scheme. As described in Section 2, the parameter λ specifies the fraction of slots that the processor near the sensor should leave for background applications. To achieve this requirement, we propose the following scheduling scheme. The executable state machine, described in Section 4, is used in the following way. If the machine is at a `must` state, the processor near the sensor sends data unconditionally. Otherwise, a Bernoulli trial with $1 - \lambda$ probability of successes is conducted and a message is sent only if the experiment successes. The input to state-machine is the sequence of decisions of the processor near the sensor and the output of the plant.

This scheduling scheme respects both λ and the needs of the control loop. The parameter λ determines the likelihood of using the bus when the conditions of the control loop allow not to. Note that an implementation of this scheme in a distributed environment means that decisions are made at the processor near the sensor. This may require the processor near the sensor to also compute the estimation as the processor near the actuator does, to get the estimated output.

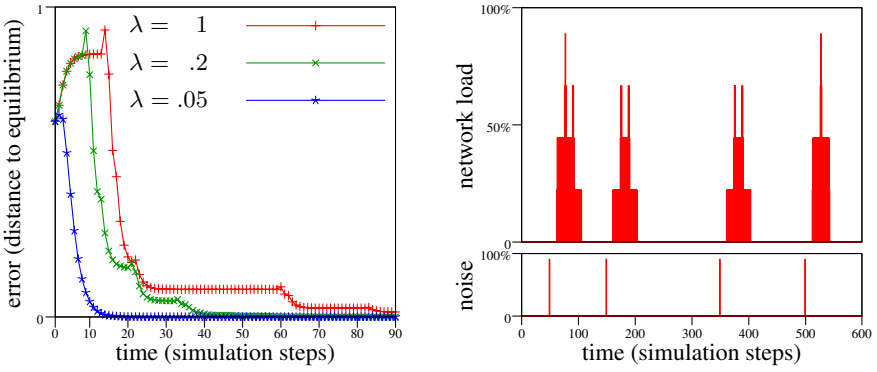
Simulation Data. To test our approach, the automaton described in Section 3 is constructed with the parameters $l = 15, \delta = 1, \epsilon = .1, \rho = 1$ and the matrices A_0 and A_1 above.

The graphs in Figure 8(a) show how our approach allows dynamic adaptation of control performance. The plots demonstrate an improvement in control performance (faster convergence) when the competition for resources is lower (smaller values of λ). This type of adaptation is not achievable with static scheduling, when the schedule is planned only for the worst-case scenario.

In another experiment, we executed the scheduler with the parameter $\lambda = 0$, and injected random noise to the control system at irregular intervals. The plot in Figure 8(b) shows that the network is only used some time after each disturbance. The upper part shows the network bandwidth used by the control loop and the lower part shows the introduced disturbances. Static approaches (including the one described in [1,2]) that do not use the output of the plant to direct scheduling decisions cannot achieve this type of dynamic adaptation.

The conclusion from the simulations is that our approach to scheduling gives best benefits for systems that operates in dynamic conditions. While static scheduling may give good results for systems with constant network load and evenly distributed disturbances, our approach delivers better performance when varying load and disturbances that come in irregular bursts are present.

Integration Into Simulink. Simulink is the de-facto standard for modeling and analysing control system. We integrate our guarded automata approach into Simulink via the Network Code Machine extension [9] in the TrueTime library. TrueTime [7] is a Simulink simulator library for embedded and networked control systems. It can



(a) Faster convergence for smaller λ. (b) Higher load after noise injection.

Fig. 8. Simulation results showing how dynamic scheduling allows adjusting control performance to network availability and adjusting network usage to control needs

simulate a number of different communication arbitration mechanisms. We extended the TrueTime library with a block for the Network Code Machine, which takes a node identifier as input and provide access to the network to the specified node. Our extension is available on the project web site and is planned to be part of the next major TrueTime release.

Figure 9 provides a sample model that shows how to run the original model shown in Figure 1 in this Simulink environment. The left part shows the control application with the actuator, plant, and sensor blocks. The middle shows the networking part consisting of the TrueTime Network, the messaging and reception blocks. The Guarded Automaton-based Scheduler block and the Network Code Machine block also belong

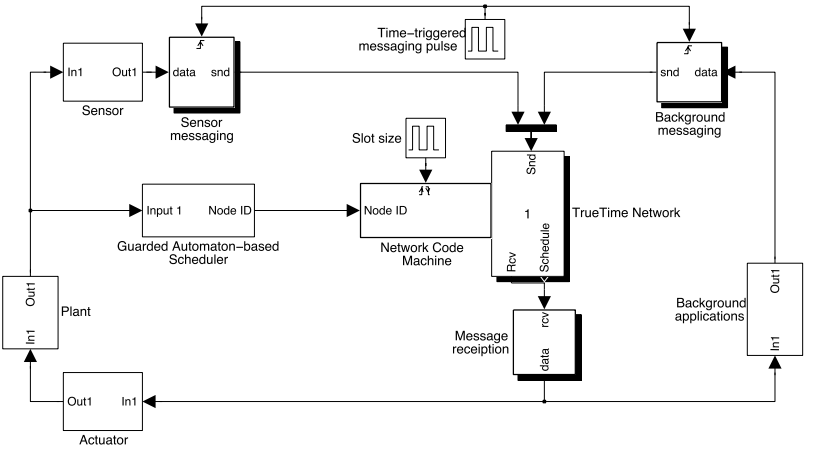


Fig. 9. Guarded automaton in Simulink with TrueTime and Network Code Machine

to the network part. The former implements the $(\rho, l, \varepsilon, \delta)$-specification-automaton as a state machine and the latter is an S-function extension for TrueTime to schedule the networked as specified in the automaton. The right side of the figure shows background application.

7 Related Work

Methodology. The concept of automata-based scheduling was introduced in [1] and [2] in the context of CPU scheduling. The main contribution of the current paper over the previous work on automata based scheduling is that it studies the use of guards as a way to direct the schedules based on dynamic data.

Another, more technical, contribution is the elimination of the need to determinize the automaton. In [1] and [2], the proposed methodologies involve computing an automaton similar (in spirit) to the automaton depicted in Figure 4 and then determinizing it. As determinization does not scale, this paper proposes a direct way of using the non-deterministic automaton for scheduling. Determinization is especially problematic with guarded automata, because they usually have more states and because the formal language they induce is over an infinite alphabet (the real numbers).

The concept of automata-based scheduling generalizes tree schedules [9]. The main structural difference is that tree schedules require the underlying structure to be a tree resulting in periodic resets to the root location. While the work on tree schedules mainly concentrated on analyzing [3], composing [4], verifying, and implementing [9] them, this work concentrates on the generation of schedulers.

Network Control Systems. Form the control perspective, the insertion of communication networks in the control loop is usually viewed as a source of random time delays and information loss (see e.g., [7,5,11,15,22,26]). For scheduling, this view leads to mechanisms such as [18,23], where a periodic schedule that can cope with the worst case delays is proposed. Our approach views the network as a shared resource. Particularly, we do not view the other users of the network as introducing random delays but as components of the system that we need to take into consideration. This perspective allows more efficient use of the resource.

Communication Arbitration. In distributed control systems, nodes must access the network mutually exclusively. The dominating approaches use either temporal isolation or priority-based mechanisms. System such as the TTA [13], TTCAN [10], or FTT-CAN [8] provide temporal isolation where each node accesses the network at predefined time slots. Our approach follows a similar line in that it uses temporal isolation in its TDMA scheme. However, in contrast to works such as the TTA and TT-Ethernet [21], our approach uses temporal isolation to synchronize steps in the automata and then uses priority-based arbitration for resource contentions instead of globally defined schedules.

Common architectures using priority-based mechanisms for resolving resource contentions either assign priorities to individual nodes or to individual messages. It is mandatory that each concurrent access must use a unique priority level, because otherwise the collision-avoidance mechanism will fail. Thus, the common architectures such as CANopen usually use a static global database assigning each message its unique priority. Our approach differs from such architectures in that we assign individual

priorities to messages based on the context of the application; meaning as a message becomes more important to the application, its priority level changes. Although the method sounds intuitive and simple, prior approaches had to rely on a quasi-static assignments of priority levels to nodes or messages, because dynamic assignment must guarantee unique priority levels. Our system can guarantee this, because we can statically check whether two nodes on the same network will ever try to communicate simultaneously with the same priority level.

8 Conclusions and Future Work

We proposed a dynamic scheduling scheme for network control systems. The main idea is using automata with guards to decide resource assignments—in our case network slots. The work contains a full walk through the development process comprising of:

1. Specifying stability parameters as high-level performance requirements.
2. Generating an automaton that specifies traces of the system implying that the requirements are not met.
3. Constructing a scheduler state-machine that guarantees avoidance of the specified traces.
4. Implementing the scheduler with priority-based congestion arbitration.

The approach is demonstrated with experiments that show its advantages compared to standard approaches and previous work. Furthermore, the experimental data demonstrates the unique ability of our approach to adjust the priority level to its context *and* stay verifiable: after disturbances the control application requires more bandwidth to adjust the plant and therefore uses elevated priority levels. While during calm operations the priority levels remain low. Future work can look into multiple-output systems, more elaborate guards on the specification automaton and more sophisticated, possibly optimal, construction algorithms for the scheduler.

References

1. Alur, R., Weiss, G.: Automata Based Interfaces for Control and Scheduling. In: Bemporad, A., Bicchi, A., Buttazzo, G. (eds.) HSCC 2007. LNCS, vol. 4416, pp. 601–613. Springer, Heidelberg (2007)
2. Alur, R., Weiss, G.: Regular Specifications of Resource Requirements for Embedded Control Software. In: Proc. 14th IEEE Real Time and Embedded Technology and Applications Symposium (RTAS) (2008)
3. Anand, M., Fischmeister, S., Lee, I.: An Analysis Framework for Network-Code Programs. In: Proc. of the 6th Annual ACM Conference on Embedded Software (EMSOFT), Seoul, South Korea, October 2006, pp. 122–131 (2006)
4. Anand, M., Fischmeister, S., Lee, I.: Composition Techniques for Tree Communication Schedules. In: Proc. of the 19th Euromicro Conference on Real-Time Systems (ECRTS), Pisa, Italy, July 2007, pp. 235–246 (2007)
5. Antsaklis, P., Baillieul, J.: Guest editorial. Special Issue on Networked Control Systems. IEEE Trans. Automat. Control 49(9), 1421–1423 (2004)
6. Bosch. CAN Specification, Version 2. Robert Bosch GmbH (September 1991)

7. Cervin, A., Henriksson, D., Lincoln, B., Eker, J., Årzén, K.-E.: How Does Control Timing Affect Performance? IEEE Control Systems Magazine 23(3), 16–30 (2003)
8. Ferreira, J., Pedreiras, P., Almeida, L., Fonseca, J.: The FTT-CAN Protocol For Flexibility in Safety-critical Systems. IEEE Micro. 22(4), 46–55 (2002)
9. Fischmeister, S., Sokolsky, O., Lee, I.: A Verifiable Language for Programming Communication Schedules. IEEE Trans. on Comp. 56(11), 1505–1519 (2007)
10. Führer, T., Müller, B., Dieterle, W., Hartwich, F., Hugel, R., Walther, M.: Time Triggered Communications on CAN (Time Triggered CAN–TTCAN). In: Proc. 7th International CAN Conference, Amsterdam, Netherlands (2000)
11. Hristu-Varsakelis, D., Levine, W.S. (eds.): Handbook of Networked and Embedded Control Systems. Birkhäuser, Basel (2005)
12. Kawamura, S., Furukawa, Y.: Automotive Electronics System, Software, and Local Area Network. In: Proc. of the International Conference on Hardware/Software Codesign and System Synthesis (2006)
13. Kopetz, H.: Real-time Systems: Design Principles for Distributed Embedded Applications. Kluwer Academic Publishers, Dordrecht (1997)
14. Liberzon, D.: Switching in Systems and Control. In: Systems & Control: Foundations & Applications, Birkhäuser Boston Inc., Boston (2003)
15. Lin, H., Antsaklis, P.J.: Stability and Persistent Disturbance Attenuation Properties for a Class of Networked Control Systems: Switched System Approach. Internat. J. Control 78(18), 1447–1458 (2005)
16. Liu, J.: Real-Time Systems. Prentice-Hall, New Jersey (2000)
17. http://www.mathworks.com/products/matlab
18. Park, H., Kim, Y., Kim, D., Kwon, W.: A Scheduling Method For Network-based Control Systems. IEEE Trans. on Control Systems Technology 10(3), 318–330 (2002)
19. Ray, A., Halevi, Y.: Integrated Communication and Control Systems: Part II–Design Considerations. ASME Journal of Dynamic Systems, Measurements and Control 110, 374–381 (1988)
20. Sánchez-Puebla, M.A., Carretero, J.: A New Approach for Distributed Computing in Avionics Systems. In: Proc. of International Symposium on Instrumentation and Control Technology (ISICT), pp. 579–584. Trinity College Dublin (2003)
21. Steinhammer, K., Grillinger, P., Ademaj, A., Kopetz, H.: A Time-Triggered Ethernet (TTE) Switch. In: Proc. of the Conference on Design, Automation and Test in Europe (DATE), Munich, Germany, pp. 794–799. European Design and Automation Association (2006)
22. Walsh, G., Ye, H., Bushnell, L.: Stability Analysis of Networked Control Systems. IEEE Transactions on Control Systems Technology 10(3), 438–446 (2002)
23. Wen, P., Cao, J., Li, Y.: Design of High-performance Networked Real-time Control Systems. IET Control Theory and Applications 1(5), 1329–1335 (2007)
24. Yliniemi, L., Leiviskä, K.: Process Control Across Network. In: Proc. of Parallel and Distributed Computing and Networks (PDCN), Anaheim, CA, USA, pp. 168–173. ACTA Press (2006)
25. Zhang, W., Branicky, M., Phillips, S.: Stability of Networked Control Systems. IEEE Control Systems Magazine 21(1), 84–99 (2001)
26. Zhang, W., Yu, L.: Output Feedback Stabilization of Networked Control Systems with Packet Dropouts. IEEE Trans. Automat. Control 52(9), 1705–1710 (2007)

Periodically Controlled Hybrid Systems

Verifying a Controller for an Autonomous Vehicle

Tichakorn Wongpiromsarn[1], Sayan Mitra[2],
Richard M. Murray[1], and Andrew Lamperski[1]

[1] California Institute of Technology
[2] University of Illinois at Urbana Champaign

Abstract. This paper introduces Periodically Controlled Hybrid Automata (PCHA) for describing a class of hybrid control systems. In a PCHA, control actions occur roughly periodically while internal and input actions may occur in the interim changing the discrete-state or the setpoint. Based on periodicity and subtangential conditions, a new sufficient condition for verifying invariance of PCHAs is presented. This technique is used in verifying safety of the planner-controller subsystem of an autonomous ground vehicle, and in deriving geometric properties of planner generated paths that can be followed safely by the controller under environmental uncertainties.

1 Introduction

Alice, an autonomous vehicle built at Caltech, successfully accomplished two of the three tasks at the National Qualifying Event of the 2007 DARPA Urban Challenge [4], [17], [5]. In executing the third task, which involved making left-turns while merging into traffic, its behavior was unsafe and almost led to a collision. Alice was stuck at the corner of a sharp turn dangerously stuttering in the middle of an intersection.

This behavior, it was later diagnosed, was caused by bad interactions between the *reactive obstacle avoidance subsystem (ROA)* and the relatively slowly reacting *path planner*. The planner incrementally generates a sequence of waypoints based on the road map, obstacles, and the mission goals. The ROA is designed to rapidly decelerate the vehicle when it gets too close to (possibly dynamic) obstacles or when the deviation from the planned path gets too large. Finally, for protecting the steering wheel, Alice's low-level controller limits the rate of steering at low speeds. Thus, accelerating from a low speed, if the planner produces a path with a sharp left turn, the controller is unable to execute the turn closely. Alice deviates from the path; the ROA activates and slows it down. This cycle continues leading to stuttering. For avoiding this behavior, the planner needs to be aware of the constraints imposed by the controller.

Finding this type of design bugs in hybrid control systems is important and challenging. While real world hybrid systems are large and complex, they are also

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 396–410, 2009.

engineered, and hence, have more structure than general hybrid automata [1]. Although restricted subclasses that are amenable to algorithmic analysis have been identified, such as rectangular-initialized [6], o-minimal [8], planar [13], and stormed [15] hybrid automata, they are not representative of restrictions that arise in engineered systems. With the motivation of abstractly capturing a common design pattern in hybrid control systems, such as Alice, and other motion control systems [11], in this paper, we study a new subclass of hybrid automata. Two main contributions of this paper are the following:

First, we define a class of hybrid control systems in which certain *control actions* occur roughly periodically. Each control action sets the *controlling input* to the plant or the physical process. In the interval between two consecutive control actions, the state of the system evolves continuously and discretely, but the control input remains constant. In particular, discrete state changes triggered by an external source may changes the waypoint or the set-point of the controller, which in turn may influence the computation of the next control input. For this class of *periodically controlled hybrid systems*, we present a sufficient condition for verifying invariant properties. The key requirement in applying this condition is to identify subset(s) C of the candidate invariant set $\mathcal{I}$, such that if the control action occurs when the system state is in C, then the subsequent control output guarantees that the system remains in $\mathcal{I}$ for the next period. The technique does not require one to solve the differential equations, instead, it relies on checking conditions on the periodicity and the subtangential condition at the boundary of $\mathcal{I}$. We are currently exploring the possibility of automating such checks using quantifier elimination [3] and optimization [14].

Secondly, we apply the above technique to verify a sequence of invariant properties of the planner-controller subsystem of Alice. From these invariants, we are able to deduce safety. That is, the deviation —distance of the vehicle from the planned path—remains within a certain constant bound. In the process, we also derive geometric properties of planner paths that guarantee that they can be followed safely by the vehicle.

The remainder of the paper is organized as follows: In Section 2 we briefly present the key definitions for the hybrid I/O automaton framework. In Section 3 we present PCHA and a sufficient condition for proving invariance. In Sections 4 and 5 we present the formal model and verification of Alice's Controller-Vehicle subsystem. Owing to limited space, complete proofs for identifying the class of safe planner paths appear in the full version of the paper available from [16].

2 Preliminaries

We use the Hybrid Input/Output Automata (HIOA) framework of [9,7] for modeling hybrid systems and the state model-based notations introduced in [10]. A Structured Hybrid I/O Automaton (SHIOA) is a non-deterministic state machine whose state may change instantaneously through a transition, or continuously over an interval of time following a *trajectory*.

Let V be a set of variables. Each variable $v \in V$ is associated with a *type*. The set of valuations of V is denoted by $val(V)$. For a valuation $\mathbf{v} \in Val(V)$

of set of variables V, its restriction to a subset of variables $Z \subseteq V$ is denoted by $\mathbf{v} \restriction Z$. A variable may be *discrete* or *continuous*. A *trajectory* for a set of variables V models continuous evolution of the values of the variables over an interval of time. Formally, a trajectory τ is a map from a left-closed interval of $\mathbb{R}_{\geq 0}$ with left endpoint 0 to $val(V)$. The domain of τ is denoted by $\tau.dom$. The *first state* of τ, $\tau.\mathsf{fstate}$, is $\tau(0)$. A trajectory τ is *closed* if $\tau.dom = [0, t]$ for some $t \in \mathbb{R}_{\geq 0}$, in which case we define $\tau.\mathsf{ltime} \triangleq t$ and $\tau.\mathsf{lstate} \triangleq \tau(t)$. For a trajectory τ for V, its restriction to a subset of variables $Z \subseteq V$ is denoted by $\tau \downarrow Z$.

For given sets of input U, output Y, and internal X variables, a *state model* $\mathcal{S}$ is a triple $(\mathscr{F}, Inv, Stop)$, where (a) $\mathscr{F}$ is a collection of Differential and Algebraic Inequalities (DAIs) involving the continuous variables in U, Y, and X, and (b) Inv and $Stop$ are predicates on X called *invariant condition* and *stopping condition* of $\mathcal{S}$. Components of $\mathcal{S}$ are denoted by $\mathscr{F}_{\mathcal{S}}$, $Inv_{\mathcal{S}}$ and $Stop_{\mathcal{S}}$. $\mathcal{S}$ defines a set of trajectories, denoted by $traj(\mathcal{S})$, for the set of variables $V = X \cup U \cup Y$. A trajectory τ for V is in the set $trajs(\mathcal{S})$ iff (a) the discrete variables in $X \cup Y$ remain constant over τ; (b) the restriction of τ on the continuous variables in $X \cup Y$ satisfies all the DAIs in $\mathscr{F}_{\mathcal{S}}$; (c) at every point in time $t \in dom(\tau)$, $(\tau \downarrow X)(t) \in Inv$; and (d) if $(\tau \downarrow X)(t) \in Stop$ for some $t \in dom(\tau)$, then τ is closed and $t = \tau.\mathsf{ltime}$.

Definition 1. *A* Structured Hybrid I/O Automaton (SHIOA) $\mathcal{A}$ *is a tuple* $(V, Q, Q_0, A, \mathcal{D}, \mathscr{S})$ *where (a)* V *is a set of variables partitioned into sets of* internal X, output Y *and* input U *variables; (b)* $Q \subseteq val(X)$ *is a set of* states *and* $Q_0 \subseteq Q$ *is a nonempty set of* start states*; (c)* A *is a set of actions partitioned into sets of* internal H, output O *and* input I *actions; (d)* $\mathcal{D} \subseteq Q \times A \times Q$ *is a set of* discrete transitions*; and (e)* $\mathscr{S}$ *is a collection of* state models *for* U, Y, *and* X, *such that for every* $\mathcal{S}, \mathcal{S}' \in \mathscr{S}$, $Inv_{\mathcal{S}} \cap Inv_{\mathcal{S}'} = \emptyset$ *and* $Q \subseteq \bigcup_{\mathcal{S} \in \mathscr{S}} Inv_{\mathcal{S}}$. *In addition,* $\mathcal{A}$ *satisfies:* **E1** *Every input action is enabled at every state.* **E2** *Given any trajectory* υ *of the input variables* U, *any* $\mathcal{S} \in \mathscr{S}$, *and* $\mathbf{x} \in Inv_{\mathcal{S}}$, *there exists* $\tau \in trajs(\mathcal{S})$ *starting from* $\mathbf{x}$, *such that either (a)* $\tau \downarrow U = \upsilon$, *or (b)* $\tau \downarrow U$ *is a proper prefix of* υ *and some action in* $H \cup O$ *is enabled at* $\tau.\mathsf{lstate}$.

For a set of state variables X, a state $\mathbf{x}$ is an element of $Val(X)$. We denote the valuation of a variable $y \in X$ at state $\mathbf{x}$, by the usual (.) notation $\mathbf{x}.y$. A transition $(\mathbf{x}, a, \mathbf{x}') \in \mathcal{D}$ is written in short as $\mathbf{x} \xrightarrow{a}_{\mathcal{A}} \mathbf{x}'$ or as $\mathbf{x} \xrightarrow{a} \mathbf{x}'$ when $\mathcal{A}$ is clear from the context. An action a is said to *enabled* at $\mathbf{x}$ if there exists $\mathbf{x}'$ such that $\mathbf{x} \xrightarrow{a} \mathbf{x}'$. We denote the components of a SHIOA $\mathcal{A}$ by $X_{\mathcal{A}}, Y_{\mathcal{A}}$, etc.

An execution of $\mathcal{A}$ records the valuations of all its variables and the occurrences of all actions over a particular run. An *execution fragment* of $\mathcal{A}$ is a finite or infinite sequence $\alpha = \tau_0 a_1 \tau_1 a_2 \ldots$ such that for all i in the sequence, $a_i \in A$, $\tau \in trajs(\mathcal{S})$ for some $\mathcal{S} \in \mathscr{S}$, and $\tau_i.\mathsf{lstate} \xrightarrow{a_{i+1}} \tau_{i+1}.\mathsf{fstate}$. An execution fragment is an *execution* if $\tau_0.\mathsf{fstate} \in Q_0$. An execution is *closed* if it is finite and the last trajectory in it is closed. The first state of α, $\alpha.\mathsf{fstate}$, is $\tau_0.\mathsf{fstate}$, and for a closed α, its last state, $\alpha.\mathsf{lstate}$, is the last state of its last trajectory. The *limit time* of α, $\alpha.\mathsf{ltime}$, is defined to be $\sum_i \tau_i.\mathsf{ltime}$. The set of executions and reachable states of $\mathcal{A}$ are denoted by $\mathsf{Execs}_{\mathcal{A}}$ and $\mathsf{Reach}_{\mathcal{A}}$. A set of states $I \subseteq Q$ is said to be an *invariant* of $\mathcal{A}$ iff $\mathsf{Reach}_{\mathcal{A}} \subseteq I$.

3 Periodically Controlled Hybrid Systems

In this section, we define a subclass of SHIOAs frequently encountered in applications involving sampled control systems and embedded systems with periodic sensing and actuation. The main result of this section, Theorem 1, gives a sufficient condition for proving invariant properties of this subclass.

A *Periodically Controlled Hybrid Automaton (PCHA)* is an SHIOA with a set of (control) actions which occur roughly periodically. For the sake of simplicity, we consider the PCHAs of the form shown in Figure 1, however, Theorem 1 generalizes to PCHAs with other input, output, and internal actions.

Let $\mathcal{X} \subseteq \mathbb{R}^n$, for some $n \in \mathbb{N}$, and $\mathcal{L}, \mathcal{Z}$, and $\mathcal{U}$ be arbitrary types. Four key variables of PCHA $\mathcal{A}$ are (a) *continuous state* variable s of type $\mathcal{X}$, initialized to x_0, (b) discrete state (*location* or *mode*) variable loc of type $\mathcal{L}$, initialized to l_0, (c) *command* variable z of type $\mathcal{Z}$, initialized to z_0, and (d) *control* variable u of type $\mathcal{U}$, initialized to u_0. The *now* and *next* variables together trigger the control action periodically.

PCHA $\mathcal{A}$ has two types of actions: (a) through input action update $\mathcal{A}$ learns about new externally produced input commands such as set-points, waypoints. When an update(z') action occurs, z' is recorded in the command variable z. (b) The control action changes the control variable u. This action occurs roughly periodically starting from time 0; the time gap between two successive occurrences is within $[\Delta_1, \Delta_1 + \Delta_2]$ where $\Delta_1 > 0, \Delta_2 \geq 0$. When control occurs, loc and s are computed as a function of their current values and that of z, and u is computed as a function of the new values of loc and s.

For each $l \in \mathcal{L}$ the continuous state s evolves according to the trajectories specified by state model $smodel(l)$, i.e., according to the differential equation $\dot{s} = f_l(s, u)$. The timing of control behavior is enforced by the precondition of control and the stopping condition of the state models.

Describing and proving invariants. Given a candidate invariant set $\mathcal{I} \subseteq Q$, we are interested in verifying that $\mathsf{Reach}_A \subseteq \mathcal{I}$. For continuous dynamical systems, checking the well-known subtangential condition (see, for example [2]) provides a

```
signature                                              1  internal control
  internal control, input update(z' : Z)                  pre now ≥ next                          2
                                                       3  eff next := now + Δ1
variables                                                    ⟨loc, s⟩ := h(loc, s, z); u := g(loc, s)  4
  internal s : X := x0                                 5
  internal discrete loc : L := l0,                        trajectories                            6
    z : Z := z0, u : U := u0                           7  trajdef smodel(l : L)
  internal now : R≥0 := 0,                                 invariant loc = l                     8
    next : R≥0 := −Δ2                                  9  evolve d(now) = 1; d(s) = fl(s, u)
                                                            stop when now = next + Δ2             10
transitions                                            11
  input update(z')
    eff z := z'                                        13
```

Fig. 1. PHCA with parameters Δ_1, Δ_2, g, h, $\{f_l\}_{l \in \mathcal{L}}$. See, for example, [10] for the description of the language

sufficient condition for proving invariance of a set $\mathcal{I}$ that is bounded by a closed surface. Theorem 1 provides an analogous sufficient condition for PCHAs. In general, however, invariant sets $\mathcal{I}$ for PCHAs have to be defined by a collection of functions instead of a single function. For each mode $l \in \mathcal{L}$, we assume that the invariant set $I_l \subseteq \mathcal{X}$ for the continuous state is defined by a collection of m *boundary functions* $\{F_{lk}\}_{k=1}^{m}$, where m is some natural number and each $F_{lk} : \mathcal{X} \to \mathbb{R}$ is a differentiable function[1]. Formally,

$$I_l \triangleq \{s \in \mathcal{X} \mid \forall k \in \{1, \ldots, m\}, F_{lk}(s) \geq 0\} \text{ and } \mathcal{I} \triangleq \{\mathbf{x} \in Q \mid \mathbf{x}.s \in I_{\mathbf{x}.loc}\}.$$

Note that $\mathcal{I}$ does not restrict the values of the command or the control variables. Lemma 1 modifies the standard inductive technique for proving invariance, so that it suffices to check invariance with respect to Control transitions and Control-free execution fragments. The proof appears in the full version [16].

Lemma 1. *Suppose $Q_0 \subseteq \mathcal{I}$ and the following two conditions hold:*

(a) (Control steps) For each state $\mathbf{x}, \mathbf{x}' \in Q$, if $\mathbf{x} \overset{\mathsf{control}}{\to} \mathbf{x}'$ and $\mathbf{x} \in \mathcal{I}$ then $\mathbf{x}' \in \mathcal{I}$,
(b) (Control-free fragments) For each closed execution fragment $\beta = \tau_0\ \mathsf{update}(z_1)\ \tau_1\ \mathsf{update}(z_2)\ \ldots\ \tau_n$ starting from a state $\mathbf{x} \in \mathcal{I}$ where each $z_i \in \mathcal{Z}$, if $\mathbf{x}.next - \mathbf{x}.now = \Delta_1$ and $\beta.\mathsf{ltime} \leq \Delta_1 + \Delta_2$, then $\beta.\mathsf{lstate} \in \mathcal{I}$.

Then $\mathsf{Reach}_{\mathcal{A}} \subseteq \mathcal{I}$.

Invariance of control steps can often be checked through case analysis which can be partially automated using a theorem prover [12]. The next key lemma provides a sufficient condition for proving invariance of control-free fragments. Since, control-free fragments do not change the valuation of the *loc* variable, for this part, we fix a value $l \in \mathcal{L}$. For each $j \in \{1, \ldots m\}$, we define the set ∂I_j to be part of the set I_l where the function F_{lj} vanishes. That is, $\partial I_j \triangleq \{s \in \mathcal{X} \mid F_{lj}(s) = 0\}$. In this paper, we call ∂I_j the j^{th} boundary of I_l even though strictly speaking, the j^{th} boundary of I_l is only a subset of ∂I_j according on the standard topological definition. Similarly, we say that the *boundary* of I_l, is $\partial I_l = \bigcup_{j \in \{1,\ldots,m\}} \partial I_j$.

Lemma 2. *Suppose that there exists a collection $\{C_j\}_{j=1}^{m}$ of subsets of I_l such that the following conditions hold:*

(a) (Subtangential) For each $s_0 \in I_l \setminus C_j$ and $s \in \partial I_j$, $\frac{\partial F_{lj}(s)}{\partial s} \cdot f_l(s, g(l, s_0)) \geq 0$.
(b) (Bounded distance) $\exists\ c_j > 0$ such that $\forall\ s_0 \in C_j, s \in \partial I_j$, $||s - s_0|| \geq c_j$.
(c) (Bounded speed) $\exists\ b_j > 0$ such that $\forall\ s_0 \in C_j, s \in I_l$, $||f_l(s, g(l, s_0))|| \leq b_j$,
(d) (Fast sampling) $\Delta_1 + \Delta_2 \leq \min_{j \in \{1,\ldots,m\}} \frac{c_j}{b_j}$.

Then, any control*-free execution fragment starting from a state in I_l where $next - now = \Delta_1$, remains within I_l.*

[1] Identical size m of the collections simplifies our notation; different number of boundary functions for different values of l can be handled by extending the theorem in an obvious way.

In Figure 2, the control and control-free fragments are shown by bullets and lines. A fragment starting in $\mathcal{I}$ and leaving $\mathcal{I}$, must cross ∂I_1. Condition (a) guarantees that if u is evaluated outside C_1, then the fragment does not leave I_l because when it reaches ∂I_1, the vector field governing its evolution points inwards with respect to ∂I_1. For a fragment starting inside C_1, condition (b) and (c) guarantee that it takes finite time before it reaches ∂I_1 and condition (d) guarantees that this finite time is at least $\Delta_1 + \Delta_2$; thus, before the trajectory crosses ∂I_1, u is evaluated again.

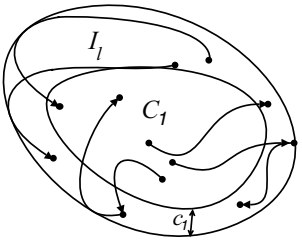


Fig. 2. An illustration for Lemma 2 with $m = 1$

Proof. We fix a control-free execution fragment $\beta = \tau_0 \mathsf{update}(z_1)\tau_1 \mathsf{update}(z_2)\ldots\tau_n$ such that at $\beta.\mathsf{fstate}$, $next - now = \Delta_1$. Without loss of generality we assume that at $\beta.\mathsf{fstate}$, $z = z_1$, $loc = l$, and $s = x_1$, where $z_1 \in \mathcal{Z}, l \in \mathcal{L}$ and $x_1 \in I_l$. We have to show that at $\beta.\mathsf{lstate}$, $s \in I_l$.

First, observe that for each $k \in \{0,\ldots,n\}$, $(\tau_k \downarrow s)$ is a solution of the differential equation(s) $d(s) = f_l(s, g(l, x_1))$. Let τ be the pasted trajectory $\tau_0 \frown \tau_1 \frown \ldots \tau_n$[2]. Let $\tau.\mathsf{ltime}$ be T. Since the update action does not change s, $\tau_k.\mathsf{lstate} \lceil s = \tau_{k+1}.\mathsf{fstate} \lceil s$ for each $k \in \{0,\ldots,n-1\}$. As the differential equations are time invariant, $(\tau \downarrow s)$ is a solution of $d(s) = f_l(s, g(l, x_1))$. We define the function $\gamma : [0,T] \to \mathcal{X}$ as $\forall\, t \in [0,T]$, $\gamma(t) \stackrel{\Delta}{=} (\tau \downarrow s)(t)$. We have to show that $\gamma(T) \in I_l$. Suppose, for the sake of contradiction, that there exists $t^* \in [0,T]$, such that $\gamma(t^*) \notin I_l$. By the definition of I_l, there exists i such that $F_{li}(\gamma(0)) \geq 0$ and $F_{li}(\gamma(t^*)) < 0$. We pick one such i and fix it for the remainder of the proof. Since F_{li} and γ are continuous, from intermediate value theorem, we know that there exists a time t_1 before t^* where F_{li} vanishes and that there is some finite time $\epsilon > 0$ after t_1 when F_{li} is strictly negative. Formally, there exists $t_1 \in [0,t^*)$ and $\epsilon > 0$ such that for all $t \in [0,t_1]$, $F_{li}(\gamma(t)) \geq 0$ and $F_{li}(\gamma(t_1)) = 0$ and for all $\delta \in (0,\epsilon]$, $F_{li}(\gamma(t_1+\delta)) < 0$.

Case 1: $x_1 \in I_l \setminus C_i$. Since $F_{li}(\gamma(t_1)) = 0$, by definition, $\gamma(t_1) \in \partial I_i$. But from the value of $F_{li}(\gamma(t))$ where t is near to t_1, we get that $\frac{\partial F_{li}}{\partial t}(t_1) = \frac{\partial F_{li}}{\partial s}(\gamma(t_1)) \cdot f_l(\gamma(t_1), g(l,x_1)) < 0$. This contradicts condition (a).

Case 2: $x_1 \in C_i$. Since for all $t \in [0,t_1]$, $F_{li}(\gamma(t)) \geq 0$ and $F_{li}(\gamma(t_1)) = 0$, we get that for all $t \in [0,t_1]$, $\gamma(t) \in I_l$ and $\gamma(t_1) \in \partial I_i$. So from condition (b) and (c), we get $c_i \leq \|\gamma(t_1) - x_1\| = \left\| \int_0^{t_1} f_l(\gamma(t), g(l,x_1))dt \right\| \leq b_i t_1$. That is, $t_1 \geq \frac{c_i}{b_i}$. But we know that $t_1 < t^* \leq T$ and periodicity of Control actions $T \leq \Delta_1 + \Delta_2$. Combining these, we get $\Delta_1 + \Delta_2 > \frac{c_i}{b_i}$ which contradicts condition (d). ∎

For PCHAs with certain properties, the following lemma provides sufficient conditions for the existence of the bounds b_j and c_j which satisfy the bounded distance and bounded speed conditions of Lemma 2.

[2] $\tau_1 \frown \tau_2$ is the trajectory obtained by concatenating τ_2 at the end of τ_1.

Lemma 3. *For a given $l \in L$, let $U_l = \{g(l,s) \mid l \in \mathcal{L}, s \in I_l\} \subseteq \mathcal{U}$ and suppose I_l is compact and f_l is continuous in $I_l \times U_l$. The bounded distance and bounded speed conditions (of Lemma 2) are satisfied if $C_j \subset I_l$ satisfies the following conditions: (a) C_j is closed, and (b) $C_j \cap \partial I_j = \emptyset$.*

Theorem 1 combines the above lemmas.

Theorem 1. *Consider a* PCHA $\mathcal{A}$ *and a set $\mathcal{I} \subseteq Q_{\mathcal{A}}$. Suppose $Q_{0\mathcal{A}} \subseteq \mathcal{I}$, $\mathcal{A}$ satisfies control invariance condition of Lemma 1, and conditions (a)-(d) of Lemma 2 for each $l \in \mathcal{L}_{\mathcal{A}}$. Then* $\mathsf{Reach}_{\mathcal{A}} \subseteq \mathcal{I}$.

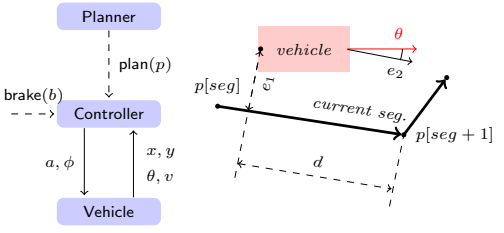
Although the PCHA of Figure 1 has one action of each type, Theorem 1 can be extended for periodically controlled hybrid systems with arbitrary number of input and internal actions. For PCHAs with polynomial vector-fields, given the semi-algebraic sets I_l and C_j, checking condition (a) and finding the c_j and b_j which satisfy conditions (b) and (c) of Lemma 2 can be formulated as a sum-of-squares optimization problem (provided that C_j and $I_l \setminus C_j$ are basic semi-algebraic sets) or as an emptiness checking problem for a semi-algebraic set. We are currently exploring the possibility of automatically checking these conditions using SOSTOOLS [14] and QEPCAD [3].

4 System Model

In this section, we describe a subsystem of an autonomous ground vehicle (Alice) consisting of the physical vehicle and the controller (see, Figure 3(a)). Vehicle captures its the position, orientation, and the velocity of the vehicle on the plane. Controller receives information about the state of the vehicle and periodically computes the input steering (ϕ) and the acceleration (a). Controller also receives an infinite[3] sequence of waypoints from a Planner and its objective is to compute a and ϕ such that the vehicle (a) remains within a certain bounded distance e_{max} of the planned path, and (b) makes progress towards successive waypoints at a target speed. Property (a) together with the assumption (possibly guaranteed by Planner) that all planned paths are at least e_{max} distance away from obstacles, imply that the Vehicle does not collide with obstacles. While the Vehicle makes progress towards a certain waypoint, the subsequent waypoints may change owing to the discovery of new obstacles, short-cuts, and changes in the mission plan. Finally, the Controller may receive an externally triggered brake input, to which it must react by slowing the vehicle down.

Vehicle. The Vehicle automaton of Figure 3 specifies the dynamics of the autonomous ground vehicle with acceleration (a) and steering angle (ϕ) as inputs. It has two parameters: (a) $\phi_{max} \in (0, \frac{\pi}{2}]$ is the physical limit on the steering angle, and (b) L is the wheelbase. The main output variables of Vehicle are (a) x and y coordinates of the vehicle with respect to a global coordinate system,

[3] The verification technique can be extended in an obvious way to handle the case where the vehicle has to follow a finite sequence of waypoints and halt at the end.




```
variables                                                    1
output x: ℝ:= x0; y: ℝ:= y0;
  θ: ℝ:= θ0; v: ℝ:= v0                                       3
input a, φ: ℝ
                                                             5
trajectories
evolve d(x) = v cos(θ)                                       7
  d(y) = v sin(θ)
  if |u.φ| ≤ φmax                                            9
    then d(θ) = (v/L) tan(φ)
    else d(θ) = (v/L) tan((φ/|φ|) φmax) fi                   11
  if v > 0 ∨ a ≥ 0
    then d(v) = a                                            13
    else d(v) = 0  fi
```

Fig. 3. (a) Planner-Controller system. (b) Deviation & disorientation. (c) Vehicle.

(b) orientation θ of the vehicle with respect to the positive direction of the x axis, and (c) vehicle's velocity v. These variables evolve according to the differential equations of lines 7-14. If the input steering angle ϕ is greater than the maximum limit ϕ_{max} then the maximum steering in the correct direction is applied. The acceleration can be negative only if the velocity is positive, and therefore the vehicle cannot move backwards. The controller ensures that the input acceleration is always within such a bound.

Controller. Figure 4 shows the SHIOA specification of the Controller automaton which reads the state of the Vehicle periodically and issues acceleration and steering outputs to achieve the aforementioned goals.

Controller is parameterized by: (a) the sampling period $\Delta \in \mathbb{R}_+$, (b) the target speed $v_T \in \mathbb{R}_{\geq 0}$, (c) proportional control gains $k_1, k_2 > 0$, (d) a constant $\delta > 0$ relating the maximum steering angle and the speed, and (e) maximum and braking accelerations $a_{max} > 0$ and $a_{brake} < 0$. Restricting the maximum steering angle instead of the maximum steering rate is a simplifying but conservative assumption. Given a constant relating the maximum steering rate and the speed, there exists δ as defined above which guarantees that the maximum steering rate requirement is satisfied.

A *path* is an infinite sequence of points $p_1, p_2, \ldots$ where $p_i \in \mathbb{R}^2$, for each i. The main state variables of Controller are the following: (a) *brake* and *new_path* are command variables, (b) *path* is the current path being followed by Controller, (c) *seg* is the index of the last waypoint visited in the current *path*. That is, $path[seg + 1]$ is the current waypoint. The straight line segment joining $path[seg]$ and $path[seg + 1]$ is called the *current segment*. (d) *deviation* e_1 is the signed perpendicular distance of the vehicle to the current segment (see, Figure 3(b)). (e) *disorientation* e_2 is the difference between the current orientation of the vehicle (θ) and the angle of the current segment. (f) *waypoint-distance* d is the signed distance of the vehicle to the current waypoint measured parallel to the current segment.

The brake(b) action is an externally controlled input action which informs the Controller about the application of an external brake ($b = On$) or the removal of

```
signature                                                          1
  input plan(p:Seq[ℝ^]), brake(b:{On,Off})                         2
  internal main
                                                                   4
variables
  input x, y, θ, v: ℝ                                              6
  output a, φ: ℝ := (0, 0)
  internal brake: {On, Off} := Off                                 8
    path: Seq[ℝ^2] := arbitrary, seg: ℕ := 1
    new_path: Seq[ℝ^2] := path                                     10
    e1, e2, d : ℝ := [e1,0, e2,0, d0]
    now: ℝ := 0; next:ℝ≥0 := 0                                     12
transitions                                                        14
  input plan(p) eff new_path := p
                                                                   16
  input brake(b) eff brake := b
                                                                   18
  internal main
  pre now = next                                                   20
  eff next := now + Δ
  if path ≠ new_path ∨ d ≤ 0 then                                  22
    if path ≠ new_path
      then seg := 1; path := new_path                              24
    elseif d ≤ 0
      then seg := seg + 1 fi                                       26
      let p = [ path[seg+1].x − path[seg].x
                path[seg+1].y − path[seg].y ]
          q = [ path[seg+1].y − path[seg].y                        28
              −(path[seg+1].x − path[seg].x) ]
          r = [ path[seg+1].x − x
                path[seg+1].y − y) ]
          e1 := 1/‖q‖ q · r;   e2 := θ − ∠p                        30
          d := 1/‖p‖ p · r     fi
                                                                   32
    let φd = −k1 e1 − k2 e2
    φ = φd/|φd| min(δ × v, |φd|)                                   34

    if brake = On then a := abrake                                 36
    elseif brake = Off ∧ v < vT
      then a := amax else a := 0 fi                                38
trajectories                                                       40
  d(now) = 1;  d(d) = -v cos(e2)
  d(e1) = v sin(e2);  d(e2) = v/L tan(φ)                           42
  stop when now = next
```

Fig. 4. Controller with parameters $v_T \in \mathbb{R}_{\geq 0}$, $k_1, k_2, \delta, \Delta \in \mathbb{R}_+$ and $a_{brake} < 0$

the brake ($b = Off$). When brake(b) occurs, b is recorded in *brake*. The plan(p) action is controlled by the external Planner and it informs the Controller about a newly planned path p. When this action occurs, the path p is recorded in *new_path*. The main action occurs once every Δ time starting from time 0 and updates $e_1, e_2, d, path, seg, a$ and ϕ as follows: A. if *new_path* is different from *path* then *seg* is set to 1 and *path* is set to *new_path*. B. Otherwise, if the waypoint-distance d is less than or equal to 0, then *seg* is set to $seg+1$ (line 26). For both of the above cases several temporary variables are computed which are in turn used to update e_1, e_2, d as specified in Lines 30-31; otherwise these variables remain unchanged. C. The steering output to the vehicle ϕ is computed using proportional control law and it is restricted to be at most δ times the velocity of the vehicle. This constraint is enforced for the mechanical protection of the steering. The steering output ϕ is set to the minimum of $-k_1e_1 - k_2e_2$ and $v \times \delta$ (line 34). D. The acceleration output a is computed using bang bang control law. If *brake* is *On* then a is set to the braking deceleration a_{brake}; otherwise, it executes a_{max} until the vehicle reaches the target speed, at which point a is set to 0.

Along a trajectory, the evolution of the variables are specified by the differential equations on lines 41-43. These differential equations are derived from the update rules described above and the differential equations governing the evolution of x, y, θ and v.

Complete System. Let $\mathcal{A}$ be the composition of the Controller and the Vehicle automata. It can be checked easily that the composed automaton $\mathcal{A}$ is a PCHA.

The key variables of $\mathcal{A}$ corresponding to those of PCHA are (a) a continuous variable $\langle x, y, \theta, v, e_1, e_2, d\rangle$ of type $\mathcal{X} = \mathbb{R}^7$, (b) a discrete variable $\langle brake, path, seg\rangle$ of type $\mathcal{L} = \texttt{Tuple}[\{On, Off\}, \texttt{Seq}[\mathbb{R}^2], \mathbb{N}]$, (c) a control variable $\langle a, \phi\rangle$ of type $\mathcal{U} = \mathbb{R}^2$, and (d) two command variables $z_1 \triangleq brake$ of type $\mathcal{Z}_1 = \{On, Off\}$ and $z_2 = path$ of type $\mathcal{Z}_2 = \texttt{Seq}[\mathbb{R}^2]$. For convenience, we define a single derived variable $s \triangleq \langle x, y, \theta, v, e_1, e_2, d\rangle$ encapsulating the continuous variable of $\mathcal{A}$. The input update actions of $\mathcal{A}$ are $\mathsf{brake}(b)$ and $\mathsf{plan}(p)$. The command variables z_1 and z_2 store the values b and p, respectively, when these actions occur. An internal control action main occurs every Δ time, starting from time 0. That is, values of Δ_1 and Δ_2 as defined in a generic PCHA are $\Delta_1 = \Delta$ and $\Delta_2 = 0$. The control law function g and the state transition function h of $\mathcal{A}$ can be derived from the specification of main action in Figure 4. Let $g = \langle g_a, g_\phi\rangle$ where $g_a : \mathcal{L} \times \mathcal{X} \to \mathbb{R}$ and $g_\phi : \mathcal{L} \times \mathcal{X} \to \mathbb{R}$ represent the control law for a and ϕ, respectively, and let $h = \langle h_{s,1}, \ldots, h_{s,7}, h_{l,1}, h_{l,2}, h_{l,3}\rangle$ where $h_{s,1}, \ldots, h_{s,7} : \mathcal{L} \times \mathcal{X} \times \mathcal{Z}_1 \times \mathcal{Z}_2 \to \mathbb{R}$ describe the discrete transition of x, y, θ, v, e_1, e_2 and d components of s, and $h_{l,1} : \mathcal{L} \times \mathcal{X} \times \mathcal{Z}_1 \times \mathcal{Z}_2 \to \{On, Off\}$, $h_{l,2} : \mathcal{L} \times \mathcal{X} \times \mathcal{Z}_1 \times \mathcal{Z}_2 \to \texttt{Seq}[\mathbb{R}^2]$ and $h_{l,3} : \mathcal{L} \times \mathcal{X} \times \mathcal{Z}_1 \times \mathcal{Z}_2 \to \mathbb{N}$ describe the discrete transition of $brake$, $path$ and seg, respectively. The definition of g and h appears in [16]. From the state models of Vehicle and Controller automata specified on line 14 of Figure 3 and lines 42-41 of Figure 4, we see that $\mathcal{A}$ only has one state model. For any value of $l \in \mathcal{L}$, the continuous state s evolves according to the differential equation $\dot{s} = f(s, u)$ where $f = \langle f_1, f_2, \ldots, f_7\rangle$ and $f_1, \ldots, f_7 : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$ are associated with the evolution of the x, y, θ, v, e_1, e_2 and d components of s, respectively.

5 Analysis of the System

Overview. The informally stated goals of the system translate to the following:

A. (*safety*) At all reachable states of $\mathcal{A}$, the deviation (e_1) of the vehicle is upper-bounded by e_{max}, where e_{max} is determined in terms of system parameters.
B. (*segment progress*) There exist certain threshold values of deviation, disorientation, and waypoint-distance such that from any state $\mathbf{x}$ with greater deviation, disorientation and waypoint-distance, the vehicle reduces its deviation and disorientation with respect to the current segment, while making progress towards its current waypoint.
C. (*waypoint progress*) The vehicle reaches successive waypoints.

In Sections 5.1 and 5.2, we define a family $\{\mathcal{I}_k\}_{k\in\mathbb{N}}$ of subsets of $Q_\mathcal{A}$ and using Lemma 2 and Lemma 3, we conclude that they are invariant with respect to the control-free execution fragments of $\mathcal{A}$. From the specification of main action, we see that the continuous state changes only occurs if $path \neq new\_path$ or waypoint-distance $d \leq 0$. Hence, using Theorem 1, we conclude that any execution fragment starting in $\mathcal{I}_k$ remains within $\mathcal{I}_k$, provided that path and current segment do not change. In Section 5.3, we discuss the proofs for properties (B) and (C) and the derivation of geometric properties of planner paths that can be followed by $\mathcal{A}$ safely. Complete proofs appear in the full version [16].

5.1 Assumptions and Family of Invariants

We define, for each $k \in \mathbb{N}$, the set $\mathcal{I}_k$ which bounds the deviation of the vehicle e_1 to be within $[-\epsilon_k, \epsilon_k]$. This bound on deviation alone, of course, does not give us an inductive invariant. If the deviation is ϵ_k and the vehicle is highly disoriented, then it would violate $\mathcal{I}_k$. Thus, $\mathcal{I}_k$ also bounds the disorientation such that the steering angle computed based on the proportional control law is within $[-\phi_k, \phi_k]$. To prevent the vehicle from not being able to turn at low speed and to guarantee that the execution speed of the controller is fast enough with respect to the speed of the vehicle, $\mathcal{I}_k$ also bounds the speed of the vehicle. $\mathcal{I}_k$ is defined in terms of $\epsilon_k, \phi_k \geq 0$ as $\mathcal{I}_k \triangleq \{\mathbf{x} \in Q \mid \forall i \in \{1, \ldots 6\}, F_{k,i}(\mathbf{x}.s) \geq 0\}$ where $F_{k,1}, \ldots, F_{k,6} : \mathbb{R}^7 \to \mathbb{R}$ are defined as follows:

$$F_{k,1}(s) = \epsilon_k - s.e_1; \quad F_{k,2}(s) = \epsilon_k + s.e_1; \quad F_{k,3}(s) = \phi_k + k_1 s.e_1 + k_2 s.e_2;$$
$$F_{k,4}(s) = \phi_k - k_1 s.e_1 - k_2 s.e_2; \quad F_{k,5}(s) = v_{max} - s.v; \quad F_{k,6}(s) = \delta s.v - \phi_b.$$

Here $v_{max} = v_T + \Delta a_{max}$ and $\phi_b > 0$ is an arbitrary constant. As we shall see shortly, the choice of ϕ_b affects the minimum speed of the vehicle and also the requirements of a brake action. We examine a state $\mathbf{x} \in \mathcal{I}_k$, that is, $F_{k,i}(\mathbf{x}.s) \geq 0$ for any $i \in \{1, \ldots 6\}$. $F_{k,1}(s), F_{k,2}(s) \geq 0$ means $s.e_1 \in [-\epsilon_k, \epsilon_k]$. $F_{k,3}(s), F_{k,4}(s) \geq 0$ means that the steering angle computed based on the proportional control law is in the range $[-\phi_k, \phi_k]$. Further, if $\phi_k \leq \phi_{max}$, then the computed steering satisfies the physical constraint of the vehicle. If, in addition, we have $\phi_b \geq \phi_k$ and $F_{k,6}(s) \geq 0$, then the vehicle actually executes the computed steering command. $F_{k,5}(s) \geq 0$ means that the speed of the vehicle is at most v_{max}.

For each $k \in \mathbb{N}$, we define $\theta_{k,1} = \frac{k_1}{k_2}\epsilon_k - \frac{1}{k_2}\phi_k$ and $\theta_{k,2} = \frac{k_1}{k_2}\epsilon_k + \frac{1}{k_2}\phi_k$, that is, the values of e_2 at which the proportional control law yields the steering angle of ϕ_k and $-\phi_k$ respectively, given that the value of e_1 is $-\epsilon_k$. From the above definitions, we make the following observations about the boundary of the $\mathcal{I}_k$ sets: for any $k \in \mathbb{N}$ and $\mathbf{x} \in \mathcal{I}_k$, $\mathbf{x}.e_2 \in [-\theta_{k,2}, \theta_{k,2}]$, $F_{k,1}(\mathbf{x}.s) = 0$ implies $\mathbf{x}.e_2 \in [-\theta_{k,2}, -\theta_{k,1}]$, $F_{k,2}(\mathbf{x}.s) = 0$ implies $\mathbf{x}.e_2 \in [\theta_{k,1}, \theta_{k,2}]$, $F_{k,3}(\mathbf{x}.s) = 0$ implies $\mathbf{x}.e_2 \in [-\theta_{k,2}, \theta_{k,1}]$, and $F_{k,4}(\mathbf{x}.s) = 0$ implies $\mathbf{x}.e_2 \in [-\theta_{k,1}, \theta_{k,2}]$.

We assume that ϕ_b and all the $\epsilon_k's$ and ϕ_k's satisfy the following assumptions that are derived from physical and design constraints on the controller. The region in the ϕ_k,ϵ_k plane which satisfies Assumption 1 can be found in [16].

Assumption 1. *(Vehicle and controller design) (a)* $\phi_k \leq \phi_b \leq \phi_{max}$ *and* $\phi_k < \frac{\pi}{2}$ *(b)* $0 \leq \theta_{k,1} \leq \theta_{k,2} < \frac{\pi}{2}$ *(c)* $L \cot \phi_k \sin \theta_{k,2} < \frac{k_2}{k_1}$ *(d)* $\Delta \leq \frac{c}{b}$ *where* $c = \frac{1}{\sqrt{k_1^2 + k_2^2}}(\phi_k - \tilde{\phi})$, $b = v_{max}\sqrt{\sin^2 \theta_{k,2} + \frac{1}{L^2}\tan^2(\tilde{\phi})}$ *and* $\tilde{\phi} = \cot^{-1}\left(\frac{k_2}{k_1 L \sin \theta_{k,2}}\right)$.

If the vehicle is forced to slow down too much at the boundary of an $\mathcal{I}_k$ by the brakes, then it may not be able to turn enough to remain inside $\mathcal{I}_k$. Thus, in verifying the above properties we need to restrict our attention to *good executions* in which brake inputs do not occur at low speeds and are not too persistent. This is formalized by the next definition.

Definition 2. *A* good execution *is an execution α that satisfies: if a* brake(On) *action occurs at time t then (a) $\alpha(t).v > \frac{\phi_b}{\delta} + \Delta|a_{brake}|$, (b)* brake($Off$) *must occur within time $t + \frac{1}{|a_{brake}|}(\alpha(t).v - \frac{\phi_b}{\delta} - \Delta|a_{brake}|)$.*

For the remainder of this section, we only consider good executions. A state $\mathbf{x} \in Q_{\mathcal{A}}$ is reachable if there exists a good execution α with $\alpha.\mathsf{lstate} = \mathbf{x}$.

5.2 Invariance Property

We fix a $k \in \mathbb{N}$ for the remainder of the section and denote $\mathcal{I}_k, F_{k,i}$ as $\mathcal{I}$ and F_i, respectively, for $i \in \{1, \ldots, 6\}$. As in Lemma 2, we define $I = \{s \in \mathcal{X} \mid F_i(s) \geq 0\}$ and for each $i \in \{1, \ldots, 6\}$, $\partial I_i = \{s \in \mathcal{X} \mid F_i(s) = 0\}$ and let the functions $f_1, f_2, \ldots, f_7 : \mathbb{R}^7 \times \mathbb{R}^2 \to \mathbb{R}$ describe the evolution of x, y, θ, v, e_1, e_2 and d, respectively. We prove that I satisfies the control-free invariance condition of Lemma 1 by applying Lemma 2.

First, we show that all the assumptions in Lemma 2 are satisfied. All the proof appears in the full version [16] which do not involve solving differential equations but require algebraic simplification of the expressions defining the vector fields and the boundaries $\{\partial I_i\}_{i \in \{1,\ldots 6\}}$ of the invariant set.

The next lemma shows that the subtangential, bounded distance and bounded speed conditions (of Lemma 2) are satisfied. The proof for $j = 5$ is presented here as an example. The rest of the proof is provided in [16].

Lemma 4. *For each $l \in \mathcal{L}$ and $j \in \{1, \ldots, 6\}$, the subtangential, bounded distance, and bounded speed conditions (of Lemma 2) are satisfied.*

Proof. Define $C_5 \stackrel{\Delta}{=} \{s \in I \mid s.v \leq v_T\}$. We apply Lemma 3 to prove the bounded distance and the bounded speed conditions. First, note that the projection of I onto the (e_1, e_2, v) space is compact and C_5 is closed. Let $\mathcal{U}_I = \{g(l, s) \mid l \in \mathcal{L},\ s \in I\}$. From the definition of I, it can be easily checked that f is continuous in $I \times \mathcal{U}_I$. In addition, $s.v = v_{max}$ for any $s \in \partial I_5$. Since $a_{max}, \Delta > 0$, $v_{max} = v_T + \Delta a_{max} > v_T$. Therefore, $C_5 \cap \partial I_5 = \emptyset$. Hence, from Lemma 3, the bounded distance and bounded speed conditions are satisfied. To prove the subtangential condition, we pick an arbitrary $s \in \partial I_5$ and $s_0 \in I \setminus C_5$. From the definition of I and C_5, $v_T < s_0.v \leq v_{max}$. Therefore, for any $l \in \mathcal{L}$, either $f_4(s, g(s_0, l)) = 0$ or $f_4(s, g(s_0, l)) = a_{brake}$ and we can conclude that $\frac{\partial F_5}{\partial s} \cdot f(s, g(s_0, l)) = -f_4(s, g(s_0, l)) \geq 0$. ∎

From the definition of each C_j, we can derive the lower bound c_j on the distance from C_j to ∂I_j and the upper bound b_j on the length of the vector field f where the control variable u is evaluated when the continuous state $s \in C_j$. Using these bounds and Assumption 1(d), we prove the sampling rate condition.

Lemma 5. *For each $l \in \mathcal{L}$, the sampling rate condition is satisfied.*

Thus, all assumptions in the hypothesis of Lemma 2 are satisfied; from Theorem 1 we obtain that good execution fragments of $\mathcal{A}$ preserve invariance of $\mathcal{I}$, provided that the path and current segment do not change over the fragment.

Theorem 2. *For any* plan*-free execution fragment β starting at a state $\mathbf{x} \in \mathcal{I}$ and ending at $\mathbf{x}' \in Q_{\mathcal{A}}$, if $\mathbf{x}.path = \mathbf{x}.new\_path$ and $\mathbf{x}.seg = \mathbf{x}'.seg$, then $\mathbf{x}' \in \mathcal{I}$.*

5.3 Identifying Safe Planner Paths: An Overview

From invariance of $\mathcal{I}_k$'s, we first show progress from $\mathcal{I}_k$ to $\mathcal{I}_{k+1}$ and then identify a class of planner paths that can be safely followed by $\mathcal{A}$. Owing to limited space, we describe the key steps in this analysis. The complete proofs appear in [16].

From Theorem 2, we know that for each $k \in \mathbb{N}$, $\mathcal{I}_k$ is an invariant of $\mathcal{A}$ with respect to execution fragments in which the *path* and the current segment do not change. First, we show that for each k, starting from any reachable state $\mathbf{x}$ in $\mathcal{I}_k$, any reachable state $\mathbf{x}'$ is in $\mathcal{I}_{k'} \subseteq \mathcal{I}_k$, where $k' = k + n$ and $n = \max(\lfloor \frac{\mathbf{x}.d - \mathbf{x}'.d}{v_{max}\Delta} \rfloor - 1, 0)$. Recall that $\mathcal{I}_k$ and $\mathcal{I}_{k'}$ are defined in terms of the deviation and the disorientation bounds ϵ_k, ϕ_k and $\epsilon_{k'}$, $\phi_{k'}$, respectively. We show that for each k, there exist nonnegative constants a_k and b_k, with $\epsilon_{k+1} = \epsilon_k - a_k$ and $\phi_{k+1} = \phi_k - b_k$, for which the above progress condition is satisfied. Furthermore, for k smaller than the threshold value k^*, we show that a_k and b_k are strictly positive, that is, $I_{k'} \subsetneq I_k$. This essentially establishes property (B), that is, upto a constant threshold, the vehicle makes progress towards reducing the deviation and disorientation with respect to its current waypoint, provided the waypoint distance is large enough. Figure 5 shows a sequence of shrinking $\mathcal{I}_k$'s visited by $\mathcal{A}$ in making progress towards a waypoint.

Next, we derive a sufficient condition on planner paths that can be safely followed with respect to a chosen set $\mathcal{I}_k$ whose parameters $\epsilon_k \in [0, e_{max}]$ and $\phi_k \in [0, \phi_{max}]$ satisfy Assumption 1. The choice of $\mathcal{I}_k$ is made such that it is the smallest invariant set containing the initial state $Q_{0\mathcal{A}}$. The key idea in the condition is: *longer path segments can be succeeded by sharper turns*. The proof is based on an invariant relationship $\mathcal{R}$ amongst the deviation, the disorientation, and waypoint distance. Following a long segment, the vehicle reduces its deviation and disorientation by the time it reaches the end, and thus, it is possible for the vehicle to turn more sharply at the end without breaking the invariance of $\mathcal{I}_k$ and the relationship $\mathcal{R}$.

Assumption 2. *(Planner paths) Let $p_0, p_1, \ldots$ be a planner path; for $i \in \{0, 1, \ldots\}$, let λ_i be the length of the segment $\overline{p_i p_{i+1}}$ and σ_i be the difference in orientation of $\overline{p_i p_{i+1}}$ and that of $\overline{p_{i+1} p_{i+2}}$. Then,*

(a) $\lambda_i \geq 2v_{max}\Delta + \epsilon_k$.

(b) Let $n = k + \lfloor \frac{\lambda_i - \epsilon_k - 2v_{max}\Delta}{v_{max}\Delta} \rfloor$. *Then, λ_i and σ_i satisfy the following conditions: (a)* $\epsilon_n \leq \frac{1}{|\cos\sigma_i|}(\epsilon_k - v_{max}\Delta|\sin(\sigma_i|)$ *and (b)* $\phi_n \leq \phi_k - k_1 v_{max}\Delta \sin|\sigma_i| - k_1\epsilon_n(1 - \cos\sigma_i) - k_2|\sigma_i|$ *where, given ϵ_k and ϕ_k, ϵ_j and ϕ_j are defined recursively for any $j > k$ by $\epsilon_j = \epsilon_{j-1} - a_{j-1}$ and $\phi_j = \phi_{j-1} - b_{j-1}$.*

The relationship between λ and the maximum value of σ which satisfies this assumption is shown in Figure 5. The choice of ϵ_k's and ϕ_k's affects both the requirements on a safe path (Assumption 2) and the definition of good executions. Larger ϵ_k's and ϕ_k's allow sharper turns in planned paths but forces brakes to

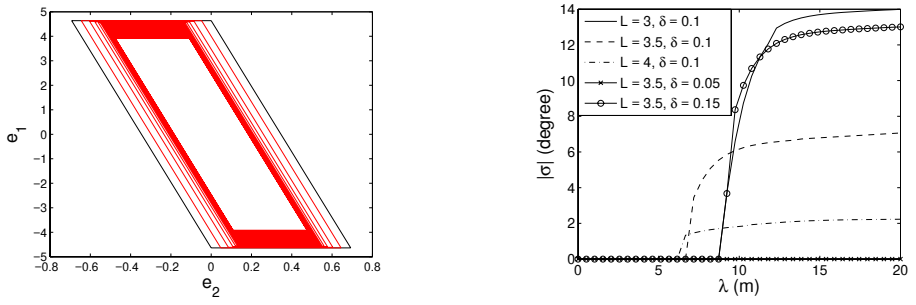


Fig. 5. *Left.* $\mathcal{I}_k$ in black, $\mathcal{I}_{k+i}$ in red for $i > 0$. *Right.* Segment length vs. maximum difference between consecutive segment orientations, for different values of L and δ.

occur only at higher speeds. This tradeoff is related to the design flaw of Alice as discussed in the introduction of the paper. Without having quantified the tradeoff, we inadvertently allowed a path to have sharp turns and also brakes at low speeds—thus violating safety.

Consider a path that satisfies Assumption 2. Further assuming that (a) a new planner path begins at the current position, (b) Vehicle is not too disoriented with respect to the new path, and (c) Vehicle speed is not too high, we establish that $\mathcal{I}_k$ is an invariant of $\mathcal{A}$. Since for any state $\mathbf{x} \in \mathcal{I}_k$, $|\mathbf{x}.e_1| \leq \epsilon_k \leq e_{max}$, invariance of $\mathcal{I}_k$ guarantees the safety property (A). For property (C), we note that for any state $\mathbf{x} \in \mathcal{I}_k$, there exists $v_{min} > 0$ such that $\mathbf{x}.v \geq v_{min} > 0$ and $|\mathbf{x}.e_2| \leq \theta_{k,2} < \frac{\pi}{2}$, that is, $\dot{d} = f_7(\mathbf{x}.s, u) \leq -v_{min} \cos\theta_{k,2} < 0$ for any $u \in \mathcal{U}$. Thus, it follows that the waypoint distance decreases and the vehicle makes progress towards its waypoint.

The simulation results are shown in Figure 6 which illustrate that the vehicle is capable of making a sharp left turn, provided that the path satisfies

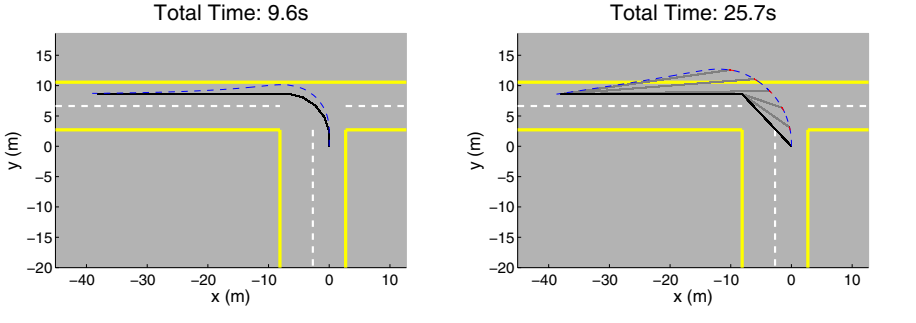


Fig. 6. The positions of the vehicle as it follows a path to execute a sharp left turn. The solid line and the dash line represent, respectively, the path and the positions of the vehicle. The initial path is drawn in black. The positions of the vehicle is plotted in blue except when *brake* is triggered in which case it is plotted in red. *Left.* The path satisfies Assumption 2. *Right.* The path does not satisfy Assumption 2 and the replan occurs due to excessive deviation. The replanned paths are drawn in grey.

Assumption 2. In addition, we are able to replicate the stuttering behavior described in the Introduction when Assumption 2 is violated.

References

1. Alur, R., Courcoubetis, C., Halbwachs, N., Henzinger, T.A., Ho, P.-H., Nicollin, X., Olivero, A., Sifakis, J., Yovine, S.: The algorithmic analysis of hybrid systems. Theoretical Computer Science 138(1), 3–34 (1995)
2. Bhatia, N.P., Szegö, G.P.: Dynamical Systems: Stability Theory and Applications. Lecture notes in Mathematics, vol. 35. Springer, Heidelberg (1967)
3. Brown, C.W.: QEPCAD B: a program for computing with semi-algebraic sets using CADs. SIGSAM Bull. 37(4), 97–108 (2003)
4. Burdick, J.W., DuToit, N., Howard, A., Looman, C., Ma, J., Murray, R.M., Wongpiromsarn, T.: Sensing, navigation and reasoning technologies for the DARPA Urban Challenge. Technical report, DARPA Urban Challenge Final Report (2007)
5. DuToit, N.E., Wongpiromsarn, T., Burdick, J.W., Murray, R.M.: Situational reasoning for road driving in an urban environment. In: International Workshop on Intelligent Vehicle Control Systems (IVCS) (2008)
6. Henzinger, T.A., Kopke, P.W., Puri, A., Varaiya, P.: What's decidable about hybrid automata? In: STOC, pp. 373–382 (1995)
7. Kaynar, D.K., Lynch, N., Segala, R., Vaandrager, F.: The Theory of Timed I/O Automata. Synthesis Lectures on Computer Science. Morgan Claypool (2005)
8. Lafferriere, G., Pappas, G.J., Yovine, S.: A new class of decidable hybrid systems. In: Vaandrager, F.W., van Schuppen, J.H. (eds.) HSCC 1999. LNCS, vol. 1569, pp. 137–151. Springer, Heidelberg (1999)
9. Lynch, N., Segala, R., Vaandrager, F.: Hybrid I/O automata. Information and Computation 185(1), 105–157 (2003)
10. Mitra, S.: A Verification Framework for Hybrid Systems. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA 02139 (September 2007)
11. Mitra, S., Wang, Y., Lynch, N.A., Feron, E.: Safety verification of model helicopter controller using hybrid input/Output automata. In: Maler, O., Pnueli, A. (eds.) HSCC 2003. LNCS, vol. 2623, pp. 343–358. Springer, Heidelberg (2003)
12. Owre, S., Rajan, S., Rushby, J.M., Shankar, N., Srivas, M.K.: PVS: Combining specification, proof checking, and model checking. In: Alur, R., Henzinger, T.A. (eds.) CAV 1996. LNCS, vol. 1102, pp. 411–414. Springer, Heidelberg (1996)
13. Prabhakar, P., Vladimerou, V., Viswanathan, M., Dullerud, G.E.: A decidable class of planar linear hybrid systems. In: Egerstedt, M., Mishra, B. (eds.) HSCC 2008. LNCS, vol. 4981, pp. 401–414. Springer, Heidelberg (2008)
14. Prajna, S., Papachristodoulou, A., Parrilo, P.A.: Introducing SOSTOOLS: A general purpose sum of squares programming solver. In: CDC 2002, pp. 741–746 (2002)
15. Vladimerou, V., Prabhakar, P., Viswanathan, M., Dullerud, G.E.: Stormed hybrid systems. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part II. LNCS, vol. 5126, pp. 136–147. Springer, Heidelberg (2008)
16. Wongpiromsarn, T., Mitra, S., Murray, R.M., Lamperski, A.: Periodically controlled hybrid systems: Verifying a controller for an autonomous vehicle. TechReport CaltechCDSTR:2008.003, California Inst. of Tech., http://resolver.caltech.edu/CaltechCDSTR:2008.003
17. Wongpiromsarn, T., Murray, R.M.: Distributed mission and contingency management for the DARPA urban challenge. In: International Workshop on Intelligent Vehicle Control Systems (IVCS) (2008)

Stabilization of Discrete-Time Switched Linear Systems: A Control-Lyapunov Function Approach*

Wei Zhang[1], Alessandro Abate[2], and Jianghai Hu[1]

[1] School of Electrical and Computer Engineering, Purdue University, IN 47907, USA
{zhang70,jianghai}@purdue.edu
[2] Department of Aeronautics and Astronautics, Stanford University, CA 94305, USA
aabate@stanford.edu

Abstract. This paper studies the exponential stabilization problem for discrete-time switched linear systems based on a control-Lyapunov function approach. A number of versions of converse control-Lyapunov function theorems are proved and their connections to the switched LQR problem are derived. It is shown that the system is exponentially stabilizable if and only if there exists a finite integer N such that the N-horizon value function of the switched LQR problem is a control-Lyapunov function. An efficient algorithm is also proposed which is guaranteed to yield a control-Lyapunov function and a stabilizing strategy whenever the system is exponentially stabilizable.

1 Introduction

One of the basic problems for switched systems is to design a switched-control feedback strategy that ensures the stability of the closed-loop system [1]. The stabilization problem for switched systems, especially autonomous switched linear systems, has been extensively studied in recent years [2]. Most of the previous results are based on the existence of a switching strategy and a Lyapunov or Lyapunov-like function with decreasing values along the closed-loop system trajectory [3,4]. These existence results have also led to some controller synthesis methods for finding the stabilizing switching strategy [5,6]. The main idea is to parameterize the switching strategy and the Lyapunov function in terms of some matrices and then translate the Lyapunov theorem to some matrix inequalities. The solution of these matrix inequalities, when existing, will define a stabilizing switching strategy. However, these matrix inequalities are usually NP-hard to solve and relaxations and heuristic methods are often required. A similar idea is used to study the stabilization problem of nonautonomous switched linear systems [7,8]. By assuming a linear state-feedback form for the continuous control of each mode, the problem is also formulated as a matrix inequality problem,

* This work was partially supported by the National Science Foundation under Grant CNS-0643805.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 411–425, 2009.

where the feedback-gain matrices are part of the design variables. Although some sufficient and necessary conditions are derived for quadratic stabilizability [4,9,10], most of the previous stabilization results are far from necessary in the sense that the system may be asymptotically or exponentially stabilizable without satisfying the proposed conditions or the derived matrix inequalities.

In this paper, we study the exponential stabilization problem for discrete-time switched linear systems. Our goal is to develop a computationally appealing way to construct both a switching strategy and a continuous control strategy to exponentially stabilize the system when none of the subsystems is stabilizable but the switched system is exponentially stabilizable. Unlike most previous methods, we propose a controller synthesis framework based on the control-Lyapunov function approach which embeds the controller design in the design of the Lyapunov function. The control-Lyapunov function approach has been widely used for studying the stabilization problem of general nonlinear systems [11,12]. However, its application in switched linear systems has not been adequately investigated. Another novelty of this paper is the derivation of some nice connections between the stabilization problem and the switched LQR problem. In particular, we show that the switched linear system is exponentially stabilizable if and only if there exists a finite integer N such that the N-horizon value function of the switched LQR problem is a control-Lyapunov function. This result not only serves as a converse control-Lyapunov function theorem, but also transforms the stabilization problem into the switched LQR problem. Motivated by the results of the switched LQR problem recently developed in [13,14,15], an efficient algorithm is proposed which is guaranteed to yield a control-Lyapunov function and a stabilizing strategy whenever the system is exponentially stabilizable. A numerical example is also carried out to demonstrate the effectiveness of the proposed algorithm.

2 Problem Formulation

We consider the discrete-time switched linear systems described by:

$$x(t+1) = A_{v(t)}x(t) + B_{v(t)}u(t),\ t \in \mathbb{Z}^+, \tag{1}$$

where $\mathbb{Z}^+$ denotes the set of nonnegative integers, $x(t) \in \mathbb{R}^n$ is the continuous state, $v(t) \in \mathbb{M} \triangleq \{1, \ldots, M\}$ is the discrete mode, and $u(t) \in \mathbb{R}^p$ is the continuous control. The integers n, M and p are all finite and the control u is unconstrained. The sequence of pairs $\{(u(t), v(t))\}_{t=0}^{\infty}$ is called the *hybrid control sequence*. For each $i \in \mathbb{M}$, A_i and B_i are constant matrices of appropriate dimensions and the pair (A_i, B_i) is called a subsystem. This switched linear system is time invariant in the sense that the set of available subsystems $\{(A_i, B_i)\}_{i=1}^{M}$ is independent of time t. We assume that there is no internal forced switchings, i.e., the system can stay at or switch to any mode at any time instant. At each time $t \in \mathbb{Z}^+$, denote by $\xi_t \triangleq (\mu_t, \nu_t) : \mathbb{R}^n \to \mathbb{R}^p \times \mathbb{M}$ the *hybrid control law* of system (1), where $\mu_t : \mathbb{R}^n \to \mathbb{R}^p$ is called the *continuous control law* and $\nu_t : \mathbb{R}^n \to \mathbb{M}$ is called the *switching control law*. A sequence of hybrid control laws constitutes an *infinite-horizon feedback policy*: $\pi \triangleq \{\xi_0, \xi_1, \ldots, \ldots\}$. If

system (1) is driven by a feedback policy π, then the closed-loop dynamics is governed by

$$x(t+1) = A_{\nu_t(x(t))}x(t) + B_{\nu_t(x(t))}\mu_t(x(t)), \quad t \in \mathbb{Z}^+. \tag{2}$$

In this paper, the policy π is allowed to be time-varying and the feedback law $\xi_t = (\mu_t, \nu_t)$ at each time step can be an arbitrary function of the state. The special policy $\pi = \{\xi, \xi, \ldots\}$ with the same feedback law $\xi_t = \xi$ at each time t is called a *stationary policy*.

Definition 1. *The origin of system (2) is exponentially stable if there exist constants $a > 0$ and $0 < c < 1$ such that the system trajectory starting from any initial state x_0 satisfies:*

$$\|x(t)\| \leq ac^t\|x_0\|.$$

Definition 2. *The system (1) is called* exponentially stabilizable *if there exists a feedback policy $\pi = \{(\mu_t, \nu_t)\}_{t\geq 0}$ under which the closed-loop system (2) is exponentially stable.*

Clearly, system (1) is exponentially stabilizable if one of the subsystems is stabilizable. A nontrivial problem is to stabilize the system when none of the subsystems are stabilizable. The main purpose of this paper is to develop an efficient and constructive way to solve the following stabilization problem.

Problem 1 (Stabilization Problem). Suppose that (A_i, B_i) is not stabilizable for any $i \in \mathbb{M}$. Find, if possible, a feedback policy π under which the closed-loop system (2) is exponentially stable.

3 A Control-Lyapunov Function Framework

We first recall a version of the Lyapunov theorem for exponential stability.

Theorem 1 (Lyapunov Theorem [16]). *Suppose that there exist a policy π and a nonnegative function $V : \mathbb{R}^n \to \mathbb{R}^+$ satisfying:*

1. *$\kappa_1\|z\|^2 \leq V(z) \leq \kappa_2\|z\|^2$ for some finite positive constants κ_1 and κ_2;*
2. *$V(x(t)) - V(x(t+1)) \geq \kappa_3\|x(t)\|^2$ for some constant $\kappa_3 > 0$, where $x(t)$ is the closed-loop trajectory of system (2) under policy π.*

Then system (2) is exponentially stable under π.

To solve the stabilization problem, one usually needs to first propose a valid policy and then construct a Lyapunov function that satisfies the conditions in the above theorem. A more convenient way is to combine these two steps together, resulting in the control-Lyapunov function approach.

Definition 3 (ECLF). *The nonnegative function $V : \mathbb{R}^n \to \mathbb{R}^+$ is called an* exponentially stabilizing control Lyapunov function *(ECLF) of system (1) if*

1. $\kappa_1\|z\|^2 \leq V(z) \leq \kappa_2\|z\|^2$ *for some finite positive constants* κ_1 *and* κ_2*;*
2. $V(z) - \inf_{\{v\in\mathbb{M}, u\in\mathbb{R}^p\}} V(A_v z + B_v u) \geq \kappa_3\|z\|^2$ *for some constant* $\kappa_3 > 0$.

The ECLF, if exists, represents certain abstract energy of the system. The second condition of Definition 3 guarantees that by choosing proper hybrid controls, the abstract energy decreases by a constant factor at each step. This together with the first condition implies the exponential stabilizability of system (1).

Theorem 2. *If system (1) has an ECLF, then it is exponentially stabilizable.*

Proof. Follows directly from Theorem 1 and Definition 3. □

If $V(z)$ is an ECLF, then one can always find a feedback law ξ that satisfies the conditions of Theorem 1. Such a feedback law is exponentially stabilizing, but may result in a large control action. A systematic way to stabilize the system with a reasonable control effort is to choose the hybrid control (u, v) that minimizes the abstract energy at the next step $V(A_v z + B_v u)$ plus certain kind of control energy expense. Toward this purpose, we introduce the following feedback law:

$$\xi_V(z) = (\mu_V(z), \nu_V(z)) = \operatorname*{arg\,inf}_{u\in\mathbb{R}^p, v\in\mathbb{M}} \left[V(A_v z + B_v u) + u^T R_v u\right], \tag{3}$$

where for each $v \in \mathbb{M}$, $R_v = R_v^T \succ 0$ characterizes the penalizing metric for the continuous control u. Since the quantity inside the bracket is bounded from below and grows to infinity as $\|u\| \to \infty$, the minimizer of (3) always exists in $\mathbb{R}^p \times \mathbb{M}$. Furthermore, if we have

$$V(z) - V(A_{\nu_V(z)} z + B_{\nu_V(z)} \mu_V(z)) \geq \kappa_3\|z\|^2, \tag{4}$$

for some constant $\kappa_3 > 0$, we know that system (1) is exponentially stabilizable by the stationary policy $\{\xi_V, \xi_V, \ldots\}$. The challenge is how to find an ECLF that satisfies (4).

In the rest of this paper, we will focus on a particular class of piecewise quadratic functions as candidates for the ECLFs of system (1). Each of these functions can be written as a pointwise minimum of a finite number of quadratic functions as follows:

$$V_{\mathcal{H}}(z) = \min_{P\in\mathcal{H}} z^T P z, \tag{5}$$

where $\mathcal{H}$ is a finite set of positive definite matrices, hereby referred to as the *FPD set*. The main reason that we focus on functions of the form (5) is that this form is sufficiently rich in terms of characterizing the ECLFs of system (1). It will be shown in Section 5 that the system is exponentially stabilizable if and only if there exists an ECLF of the form (5).

With the particular structure of the candidate ECLFs (5), the feedback law defined in (3) can be derived in closed form. Its expression is closely related to the Riccati equation and the Kalman gain of the classical LQR problem. To derive this expression, we first define a few notations. Let $\mathcal{A}$ be the *positive*

semidefinite cone, namely, the set of all symmetric positive semidefinite (p.s.d.) matrices. For each subsystem $i \in \mathbb{M}$, define a mapping $\rho_i^0 : \mathcal{A} \to \mathcal{A}$ as:

$$\rho_i^0(P) = A_i^T P A_i - A_i^T P B_i (R_i + B_i^T P B_i)^{-1} B_i^T P A_i. \tag{6}$$

It will become clear in Section 4 that the mapping ρ_i^0 is the difference Riccati equation of subsystem i with a zero state-weighting matrix. For each subsystem $i \in \mathbb{M}$ and each p.s.d. matrix P, the Kalman gain is defined as

$$K_i(P) \triangleq (R_i + B_i^T P B_i)^{-1} B_i^T P A_i. \tag{7}$$

Lemma 1. *Let $\mathcal{H}$ be an arbitrary FPD set. Let $V_{\mathcal{H}} : \mathbb{R}^n \to \mathbb{R}^+$ be defined by $\mathcal{H}$ through (5). Then the feedback law defined in (3) is given by*

$$\xi_{V_{\mathcal{H}}}(z) = \left(-K_{i_{\mathcal{H}}(z)}\left(P_{\mathcal{H}}(z)\right) z, i_{\mathcal{H}}(z)\right), \tag{8}$$

where $K_i(\cdot)$ is the Kalman gain defined in (7) and

$$\left(P_{\mathcal{H}}(z), i_{\mathcal{H}}(z)\right) = \operatorname*{arg\,min}_{P \in \mathcal{H}, i \in \mathbb{M}} z^T \rho_i^0(P) z. \tag{9}$$

Proof. By (3), to find ξ_V, we need to solve the following optimization problem:

$$\begin{aligned} f(z) &\triangleq \inf_{u \in \mathbb{R}^p, i \in \mathbb{M}} \left[\min_{P \in \mathcal{H}} (A_i z + B_i u)^T P (A_i z + B_i u) + u^T R_i u \right] \\ &= \min_{i \in \mathbb{M}, P \in \mathcal{H}} \left\{ \inf_{u \in \mathbb{R}^p} \left[(A_i z + B_i u)^T P (A_i z + B_i u) + u^T R_i u \right] \right\}. \end{aligned} \tag{10}$$

For each $i \in \mathbb{M}$ and $P \in \mathcal{H}$, the quantity inside the square bracket is quadratic in u. Thus, the optimal value of u can be easily computed as $u^* = -K_i(P)z$, where $K_i(P)$ is the Kalman gain defined in (7). Substituting u^* into (10) and simplifying the resulting expression yields $f(z) = z^T \rho_{i_{\mathcal{H}}(z)}^0(P_{\mathcal{H}}(z))z$, where $P_{\mathcal{H}}(z)$ and $i_{\mathcal{H}}(z)$ are defined in (9). □

To check whether a function $V_{\mathcal{H}}$ defined by a FPD set $\mathcal{H}$ is an ECLF, it is convenient to introduce another FPD set $\mathcal{F}_{\mathcal{H}}$ defined as:

$$\mathcal{F}_{\mathcal{H}} = \{\rho_i^0(P) : i \in \mathbb{M} \text{ and } P \in \mathcal{H}\}. \tag{11}$$

In other words, $\mathcal{F}_{\mathcal{H}}$ contains all the possible images of the mapping $\rho_i^0(P)$ as i ranges over $\mathbb{M}$ and P ranges over $\mathcal{H}$.

Theorem 3. *Let $\mathcal{H}$ be an arbitrary FPD set. Let $V_{\mathcal{H}} : \mathbb{R}^n \to \mathbb{R}^+$ and $V_{\mathcal{F}_{\mathcal{H}}} : \mathbb{R}^n \to \mathbb{R}^+$ be defined by $\mathcal{H}$ and $\mathcal{F}_{\mathcal{H}}$, respectively, by (5). Then the stationary policy $\pi_{V_{\mathcal{H}}} = \{\xi_{V_{\mathcal{H}}}, \xi_{V_{\mathcal{H}}}, \ldots\}$ is exponentially stabilizing if*

$$V_{\mathcal{H}}(z) - V_{\mathcal{F}_{\mathcal{H}}}(z) \geq \kappa_3 \|z\|^2, \tag{12}$$

for all $z \in \mathbb{R}^n$ and some constant $\kappa_3 > 0$.

Proof. Obviously, $V_{\mathcal{H}}$ satisfies the first condition of Definition 3. By (8), it can be easily verified that (12) implies (4). Thus, $V_{\mathcal{H}}$ is an ECLF satisfying (4) and the desired result follows. □

For a given function $V_{\mathcal{H}}$ of the form (5), to see whether it is an ECLF, we should check condition (12). Since both $V_{\mathcal{H}}$ and $V_{\mathcal{F}_{\mathcal{H}}}$ are homogeneous, we only need to consider the points on the unit sphere in $\mathbb{R}^n$ to verify (12). In $\mathbb{R}^2$, a practical way of checking (12) is to plot the functions $V_{\mathcal{H}}(z)$ and $V_{\mathcal{F}_{\mathcal{H}}}(z)$ along the unit circle to see whether $V_{\mathcal{H}}(z)$ is uniformly above $V_{\mathcal{F}_{\mathcal{H}}}(z)$. In higher dimensional state spaces, there is no general way to efficiently verify this condition. Nevertheless, a sufficient convex condition can be obtained using the S-procedure.

Theorem 4 (Convex Test). *With the same notations as in Theorem 3, the stationary policy $\pi_{V_{\mathcal{H}}} = \{\xi_{V_{\mathcal{H}}}, \xi_{V_{\mathcal{H}}}, \ldots\}$ is exponentially stabilizing if for each $P_{\mathcal{H}} \in \mathcal{H}$, there exists nonnegative constants α_j, $j = 1, \ldots, k$, such that*

$$\sum\nolimits_{j=1}^{k} \alpha_j = 1, \text{ and } P_{\mathcal{H}} \succ \sum\nolimits_{j=1}^{k} \alpha_j P_{\mathcal{F}_{\mathcal{H}}}^{(j)}, \tag{13}$$

where $k = |\mathcal{F}_{\mathcal{H}}|$ and $\{P_{\mathcal{F}_{\mathcal{H}}}^{(j)}\}_{j=1}^{k}$ is an enumeration of $\mathcal{F}_{\mathcal{H}}$.

Proof. See [17]. □

4 A Converse ECLF Theorem Using Dynamic Programming

By focusing on the ECLFs of the form (5) and the feedback laws of the form (3), the stabilization problem becomes a quadratic optimal control problem. The main purpose of this section is to prove that system (1) is exponentially stabilizable if and only if there exists an ECLF that satisfies (4). Our approach is based on the theory of the switched LQR problem recently developed in [13,15].

4.1 The Switched LQR Problem

Let $Q_i = Q_i^T \succ 0$ and $R_i = R_i^T \succ 0$ be the weighting matrices for the state and the control, respectively, for subsystem $i \in \mathbb{M}$. Define the running cost as

$$L(x, u, v) = x^T Q_v x + u^T R_v u, \quad \text{for } x \in \mathbb{R}^n, u \in \mathbb{R}^p, v \in \mathbb{M}. \tag{14}$$

Denote by $J_\pi(z)$ the total cost, possibly infinite, starting from $x(0) = z$ under policy π, i.e.,

$$J_\pi(z) = \sum\nolimits_{t=0}^{\infty} L(x(t), \mu_t(x(t)), \nu_t(x(t))). \tag{15}$$

Denote by $\mathit{\Pi}$ the set of all admissible policies, i.e., the set of all sequences of functions $\pi = \{\xi_0, \xi_1, \ldots\}$ with $\xi_t : \mathbb{R}^n \to \mathbb{R}^p \times \mathbb{M}$ for $t \in \mathbb{Z}^+$. Define $V^*(z) = \inf_{\pi \in \mathit{\Pi}} J_\pi(z)$. Since the running cost is always nonnegative, the infimum always

exists. The function $V^*(z)$ is usually called the *infinite-horizon value function*. It will be infinite if $J_\pi(z)$ is infinite for all the policies $\pi \in \Pi$. As a natural extension of the classical LQR problem, the *Discrete-time Switched LQR problem* (DSLQR) is defined as follows.

Problem 2 (DSLQR problem). For a given initial state $z \in \mathbb{R}^n$, find the infinite-horizon policy $\pi \in \Pi$ that minimizes $J_\pi(z)$ subject to equation (2).

4.2 The Value Functions of the DSLQR Problem

Dynamic programming solves the DSLQR problem by introducing a sequence of value functions. Define the N-horizon value function $V_N : \mathbb{R}^n \to \mathbb{R}$ as:

$$V_N(z) = \inf_{\substack{u(t)\in\mathbb{R}^p, v(t)\in\mathbb{M} \\ 0\le t\le N-1}} \left\{ \sum_{t=0}^{N-1} L(x(t), u(t), v(t)) \Big| x(0)=z \right\}. \tag{16}$$

For any function $V : \mathbb{R}^n \to \mathbb{R}^+$ and any feedback law $\xi = (\mu, \nu) : \mathbb{R}^n \to \mathbb{R}^p \times \mathbb{M}$, denote by $\mathcal{T}_\xi$ the operator that maps V to another function $\mathcal{T}_\xi[V]$ defined as:

$$\mathcal{T}_\xi[V](z) = L(z, \mu(z), \nu(z)) + V(A_{\nu(z)}z + B_{\nu(z)}\mu(z)),\ \forall z \in \mathbb{R}^n. \tag{17}$$

Similarly, for any function $V : \mathbb{R}^n \to \mathbb{R}^+$, define the operator $\mathcal{T}$ by

$$\mathcal{T}[V](z) = \inf_{u\in\mathbb{R}^p, v\in\mathbb{M}} \{L(z, u, v) + V(A_v z + B_v u)\},\ \forall z \in \mathbb{R}^n. \tag{18}$$

The equation defined above is called the *one-stage value iteration* of the DSLQR problem. We denote by $\mathcal{T}^k$ the composition of the mapping $\mathcal{T}$ with itself k times, i.e., $\mathcal{T}^k[V](z) = \mathcal{T}\left[\mathcal{T}^{k-1}[V]\right](z)$ for all $k \in \mathbb{Z}^+$ and $z \in \mathbb{R}^n$. Some standard results of Dynamic Programming are summarized in the following lemma.

Lemma 2 ([18]). *Let $V_0(z) = 0$ for all $z \in \mathbb{R}^n$. Then (i) $V_N(z) = \mathcal{T}^N[V_0](z)$ for all $N \in \mathbb{Z}^+$ and $z \in \mathbb{R}^n$; (ii) $V_N(z) \to V^*(z)$ pointwise in $\mathbb{R}^n$ as $N \to \infty$. (iii) The infinite-horizon value function satisfies the Bellman equation, i.e., $\mathcal{T}[V^*](z) = V^*(z)$ for all $z \in \mathbb{R}^n$.*

To derive the value function of the DSLQR problem, we introduce a few definitions. Denote by $\rho_i : \mathcal{A} \to \mathcal{A}$ the *Riccati Mapping* of subsystem $i \in \mathbb{M}$, i.e.,

$$\rho_i(P) = Q_i + A_i^T P A_i - A_i^T P B_i (R_i + B_i^T P B_i)^{-1} B_i^T P A_i. \tag{19}$$

Definition 4. *Let $2^{\mathcal{A}}$ be the power set of $\mathcal{A}$. The mapping $\rho_{\mathbb{M}} : 2^{\mathcal{A}} \to 2^{\mathcal{A}}$ defined by: $\rho_{\mathbb{M}}(\mathcal{H}) = \{\rho_i(P) : i \in \mathbb{M}$ and $P \in \mathcal{H}\}$ is called the* Switched Riccati Mapping *associated with Problem 2.*

Definition 5. *The sequence of sets $\{\mathcal{H}_k\}_{k=0}^N$ generated iteratively by $\mathcal{H}_{k+1} = \rho_{\mathbb{M}}(\mathcal{H}_k)$ with initial condition $\mathcal{H}_0 = \{0\}$ is called the* Switched Riccati Sets *of Problem 2.*

The switched Riccati sets always start from a singleton set $\{0\}$ and evolve according to the switched Riccati mapping. For any finite N, the set $\mathcal{H}_N$ consists of up to M^N p.s.d. matrices. An important fact about the DSLQR problem is that its value functions are completely characterized by the switched Riccati sets.

Theorem 5 ([13]). *The N-horizon value function for the DSLQR problem is given by*

$$V_N(z) = \min_{P \in \mathcal{H}_N} z^T P z. \tag{20}$$

4.3 A Converse ECLF Theorem

The main purpose of this subsection is to show that if system (1) is exponentially stabilizable, then an ECLF must exist and can be chosen to be the infinite-horizon value function V^* of the DSLQR problem. Denote by $\lambda_{\min}(\cdot)$ and $\lambda_{\max}(\cdot)$ the smallest and the largest eigenvalue of a p.s.d. matrix, respectively. Let

$$\sigma_A^+ = \max_{i \in \mathbb{M}} \left\{ \sqrt{\lambda_{\max}(A_i^T A_i)} \right\}, \ \lambda_Q^- = \min_{i \in \mathbb{M}} \{\lambda_{\min}(Q_i)\},$$
$$\lambda_Q^+ = \max_{i \in \mathbb{M}} \{\lambda_{\max}(Q_i)\}, \lambda_R^- = \min_{i \in \mathbb{M}} \{\lambda_{\min}(R_i)\} \text{ and } \lambda_R^+ = \max_{i \in \mathbb{M}} \{\lambda_{\max}(R_i)\}.$$

We first prove some important properties of V^*.

Lemma 3. *If system (1) is exponentially stabilizable, then (i) there exists a constant $\beta < \infty$ such that $\lambda_Q^- \|z\|^2 \leq V^*(z) \leq \beta \|z\|^2$; (ii) there exists a stationary optimal policy.*

Proof. (i) The proof of the first part is rather technical and is thus omitted here. Interested readers may refer to [17] for the detailed proof. (ii) By Lemma 2, $V^*(z)$ satisfies the Bellman equation, i.e.,

$$V^*(z) = \inf_{u \in \mathbb{R}^p, v \in \mathbb{M}} \{L(z,u,v) + V^*(A_v z + B_v u)\}, \ \forall z \in \mathbb{R}^n. \tag{21}$$

Let z be arbitrary and fixed. If $V^*(z)$ is infinite, then an arbitrary $\xi^*(z) \in \mathbb{R}^p \times \mathbb{M}$ achieves the infimum of (21) which is infinite. Now suppose $V^*(z)$ is finite. Then there exists a hybrid control (u, v) under which the quantity inside the bracket of (21) is finite. Denote by $\hat{V}$ this finite number. Since $R_v \succ 0$ for all $v \in \mathbb{M}$, there must exists a compact set $\mathcal{U}$ such that $L(z,u,v) \geq \hat{V}$ as long as $u \notin \mathcal{U}$. This implies that

$$V^*(z) = \inf_{u \in \mathcal{U}, v \in \mathbb{M}} \{L(z,u,v) + V^*(A_v z + B_v u)\}.$$

Since $\mathcal{U}$ is compact, there always exists a hybrid control that achieves the infimum of (21). Therefore, in any case, there must exist a feedback law $\xi^*(z) = (\mu^*(z), \nu^*(z))$ such that $\mathcal{T}_{\xi^*}[V^*](z) = V^*(z)$ for each $z \in \mathbb{R}^n$. □

The following theorem relates the exponential stabilizability with the infinite-horizon value function V^*.

Theorem 6 (Converse ECLF Theorem I). *System (1) is exponentially stabilizable if and only if $V^*(z)$ is an ECLF of system (1) that satisfies condition (4).*

Proof. The "only if" part follows directly from Theorem 2. Now suppose that system (1) is exponentially stabilizable. By part (i) of Lemma 3, $V^*(z)$ satisfies the first condition of Definition 3. Furthermore, by part (ii) of Lemma 3, there exists a feedback law $\xi^* = (\mu^*, \nu^*)$ such that $V^*(z) = \mathcal{T}_{\xi^*}[V^*](z)$. This implies that

$$V^*(z) - V^*(A_{\nu^*(z)}z + B_{\nu^*(z)}\mu^*(z)) - [\mu^*(z)]^T R_{\nu^*(z)}[\mu^*(z)] \geq \lambda_Q^- \|z\|^2.$$

Let $\xi_{V^*} = (\hat{\mu}, \hat{\nu})$ be defined as in (3) with V replaced by V^*. Then we have

$$\begin{aligned} &V^*(z) - V^*\left(A_{\hat{\nu}(z)}z + B_{\hat{\nu}(z)}\hat{\mu}(z)\right) \\ \geq &V^*(z) - V^*\left(A_{\hat{\nu}(z)}z + B_{\hat{\nu}(z)}\hat{\mu}(z)\right) - [\hat{\mu}(z)]^T R_{\hat{\nu}(z)}[\hat{\mu}(z)] \\ \geq &V^*(z) - V^*\left(A_{\nu^*(z)}z + B_{\nu^*(z)}\mu^*(z)\right) - [\mu^*(z)]^T R_{\nu^*(z)}[\mu^*(z)] \geq \lambda_Q^- \|z\|^2, \end{aligned}$$

where the last step follows from the definition of ξ_{V^*} in (3). Thus, V^* also satisfies condition (4). Hence, V^* is an ECLF satisfying (4). □

By this theorem, whenever system (1) is exponentially stabilizable, $V^*(z)$ can be used as an ECLF to construct an exponentially stabilizing feedback law ξ_{V^*}. However, from a design view point, such an existence result is not very useful as V^* can seldom be obtained exactly. In the next section, we will develop an efficient algorithm to compute an approximation of V^* which is also guaranteed to be an ECLF of system (1).

5 Efficient Computation of ECLFs

In this section, we will find an approximation of V^* which can be efficiently computed yet close enough to V^* so that it remains a valid ECLF of system (1). To find such an approximation, we need the following convergence result.

Theorem 7 ([14]). *If $V^*(z) \leq \beta\|z\|^2$ for some $\beta < \infty$, then*

$$|V_{N_1}(z) - V_N(z)| \leq \alpha\gamma^N \|z\|^2, \tag{22}$$

for any $N_1 \geq N \geq 1$, where $\gamma = \frac{1}{1+\lambda_Q^-/\beta} < 1$ and $\alpha = \max\{1, \frac{\sigma_A^+}{\gamma}\}$.

By this theorem, the N-horizon value function V_N approaches V^* exponentially fast as $N \to \infty$. Therefore, as we increase N, V_N will quickly become an ECLF of system (1).

Theorem 8 (Converse ECLF Theorem II). *If system (1) is exponentially stabilizable, then there exists an integer $N_0 < \infty$ such that $V_N(z)$ is an ECLF satisfying condition (4) for all $N \geq N_0$.*

Proof. Define

$$\xi_N^*(z) = (\mu_N^*, \nu_N^*) \triangleq \underset{u\in\mathbb{R}^p, v\in\mathbb{M}}{\arg\inf} \{L(z,u,v) + V_N(A_v z + B_v u)\}. \tag{23}$$

By Lemma 2 and equation (23), we know that

$$V_{N+1}(z) = \mathcal{T}[V_N](z) = \mathcal{T}_{\xi_N^*}(z)[V_N](z), \forall z \in \mathbb{R}^n.$$

We now fix an arbitrary $z \in \mathbb{R}^n$ and let $u^* = \mu_N^*(z)$, $v^* = \nu_N^*(z)$ and $x^*(1) = A_{v^*} z + B_{v^*} u^*$. Therefore, $V_{N+1}(z) - V_N(x^*(1)) - (u^*)^T R_{v^*}(u^*) \geq \lambda_Q^- \|z\|^2$. By Theorem 7, $V_{N+1}(z) \leq V_N(z) + \alpha\gamma^N \|z\|^2$. Hence,

$$V_N(z) - V_N(x^*(1)) - (u^*)^T R_{v^*}(u^*) \geq (\lambda_Q^- - \alpha\gamma^N)\|z\|^2.$$

Thus, there must exist an $N_0 \leq \infty$ such that $(\lambda_Q^- - \alpha\gamma^N) > \lambda_Q^-/2$ for all $N \geq N_0$. Then, by a similar argument as in the proof of Theorem 6, we can conclude that V_N is an ECLF satisfying (4) for all $N \geq N_0$. □

Theorem 8 implies that when the system is exponentially stabilizable, the ECLF not only exists but can also be chosen to be a piecewise quadratic function of the form (5). Furthermore, as N increases, the N-horizon value function V_N will eventually become an ECLF. Therefore, to solve the stabilization problem, we only need to compute the switched Riccati set $\mathcal{H}_N$. However, this method may not be computationally feasible as the size of $\mathcal{H}_N$ grows exponentially fast as N increases. Fortunately, if we allow a small numerical relaxation, an approximation of V_N can be efficiently computed [15].

Definition 6 (Numerical Redundancy). *A matrix $\hat{P} \in \mathcal{H}_N$ is called* (numerically) ϵ-redundant *with respect to $\mathcal{H}_N$ if*

$$\min_{P\in\mathcal{H}_N\setminus\hat{P}} z^T P z \leq \min_{P\in\mathcal{H}_N} z^T (P + \epsilon I_n) z, \text{ for any } z \in \mathbb{R}^n.$$

Definition 7 (ϵ-ES). *The set $\mathcal{H}_N^\epsilon$ is called an ϵ-Equivalent-Subset (ϵ-ES) of $\mathcal{H}_N$ if $\mathcal{H}_N^\epsilon \subset \mathcal{H}_N$ and for all $z \in \mathbb{R}^n$,*

$$\min_{P\in\mathcal{H}_N} z^T P z \leq \min_{P\in\mathcal{H}_N^\epsilon} z^T P z \leq \min_{P\in\mathcal{H}_N} z^T (P + \epsilon I_n) z.$$

Removing the ϵ-redundant matrices may introduce some error for the value function; but the error is no larger than ϵ for $\|z\| \leq 1$. To simplify the computation, for a given tolerance ϵ, we want to prune out as many ϵ-redundant matrices as possible. The following lemma provides a sufficient condition for testing the ϵ-redundancy for a given matrix.

Lemma 4 (Redundancy Test). *$\hat{P}$ is ϵ-redundant in $\mathcal{H}_N$ if there exist non-negative constants $\{\alpha_1\}_{i=1}^{k-1}$ such that $\sum_{i=1}^k \alpha_i = 1$ and $\hat{P} + \epsilon I_n \succeq \sum_{i=1}^k \alpha_i P^{(i)}$, where $k = |\mathcal{H}_N|$ and $\{P^{(i)}\}_{i=1}^{k-1}$ is an enumeration of $\mathcal{H}_N \setminus \{\hat{P}\}$.*

Algorithm 1

1. Denote by $P^{(i)}$ the i^{th} matrix in $\mathcal{H}_N$. Specify a tolerance ϵ and set $\mathcal{H}_N^{(1)} = \{P^{(1)}\}$.
2. For each $i = 2, \ldots, |\mathcal{H}_N|$, if $P^{(i)}$ satisfies the condition in Lemma 4 with respect to $\mathcal{H}_N$, then $\mathcal{H}_N^{(i)} = \mathcal{H}_N^{(i-1)}$; otherwise $\mathcal{H}_N^{(i)} = \mathcal{H}_N^{(i-1)} \cup \{P^{(i)}\}$.
3. Return $\mathcal{H}_N^{(|\mathcal{H}_N|)}$.

The condition in Lemma 4 can be easily verified using various existing convex optimization algorithms [19]. To compute an ϵ-ES of $\mathcal{H}_N$, we only need to remove the matrices in $\mathcal{H}_N$ that satisfy the condition in Lemma 4. The detailed procedure is summarized in Algorithm 1. Denote by $Algo_\epsilon(\mathcal{H}_N)$ the ϵ-ES of $\mathcal{H}_N$ returned by the algorithm. To further reduce the complexity, we can remove the ϵ-redundant matrices after every switched Riccati mapping. To this end, we define the *relaxed switched Riccati sets* $\{\mathcal{H}_k^\epsilon\}_{k=0}^N$ iteratively as:

$$\mathcal{H}_0^\epsilon = \mathcal{H}_0 \text{ and } \mathcal{H}_{k+1}^\epsilon = Algo_\epsilon(\rho_{\mathbb{M}}(\mathcal{H}_k^\epsilon)), \text{ for } k \leq N-1. \tag{24}$$

The function defined based on $\mathcal{H}_N^\epsilon$ is very close to V_N but much easier to compute as $\mathcal{H}_N^\epsilon$ usually contains much fewer matrices than $\mathcal{H}_N$. We now use the following example to demonstrate the simplicity of computing the set $\mathcal{H}_N^\epsilon$.

$$A_1 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}, A_2 = \begin{bmatrix} 1.5 & 1 \\ 0 & 1.5 \end{bmatrix}, B_1 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, B_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, Q_i = I_2, R_i = 1, i = 1, 2. \tag{25}$$

Clearly, neither subsystem is stabilizable. As shown in Fig. 1, a direct computation of $\{\mathcal{H}_k\}_{k=0}^N$ results in a combinatorial complexity of the order 10^9 for $N = 30$. However, if we use the relaxed iteration (24) with $\epsilon = 10^{-3}$, eventually $\mathcal{H}_N^\epsilon$ contains only 16 matrices. This example shows that the numerical

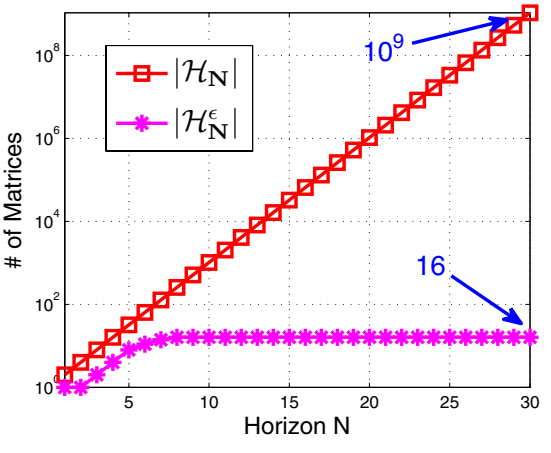


Fig. 1. Evolution of $|\mathcal{H}_N^\epsilon|$ with $\epsilon = 10^{-3}$

Algorithm 2 (Computation of ECLF)

Specify proper values for ϵ, ϵ_{min} and N_{max}.
while $\epsilon > \epsilon_{\min}$ **do**
  for $N = 0$ to $N_{\max}$ **do**
    $\mathcal{H}_{N+1} = Algo_\epsilon(\rho_{\mathbb{M}}(\mathcal{H}_N))$
    if $\mathcal{H}^\epsilon_{N+1}$ satisfies the condition of Theorem 4 **then**
      stop and return V^ϵ_N as an ECLF
    end if
  end for
  $\epsilon = \epsilon/2$
end while

relaxation can dramatically simplify the computation of $\mathcal{H}_N$. Our next task is to show that this relaxation does not change the value function too much. Define $V^\epsilon_N(z) = \min_{P\in\mathcal{H}^\epsilon_N} z^T P z$. It is proved in [15] that the total error between $V^\epsilon_N(z)$ and $V_N(z)$ can be bounded uniformly with respect to N.

Lemma 5 ([15]). *If $V^*(z) \le \beta\|z\|^2$ for some $\beta < \infty$, then*

$$V_N(z) \le V^\epsilon_N(z) \le V_N(z) + \epsilon\eta\|z\|^2, \tag{26}$$

where $\eta = \frac{1+(\beta/\lambda^-_Q - 1)\gamma}{1-\gamma}$.

The above lemma indicates that by choosing ϵ small enough, V^ϵ_N can approximate V_N with arbitrary accuracy. This warrants V^ϵ_N as an ECLF for large N and small ϵ.

Theorem 9 (Converse ECLF Theorem III). *If system (1) is exponentially stabilizable, then there exists an integer $N_0 < \infty$ and a real number $\epsilon_0 > 0$ such that $V^\epsilon_N(z)$ is an ECLF of system (2) satisfying condition (4) for all $N \ge N_0$ and all $\epsilon < \epsilon_0$.*

Proof. Similar to the proof of Theorem 8.

In summary, if the system is exponentially stabilizable, we can always find an ECLF of the form (5) defined by $\mathcal{H}^\epsilon_N$. To compute such an ECLF, we can start from a reasonable guess of ϵ and perform the relaxed switched Riccati mapping (24). After each iteration, we need to check whether the condition of Theorem 4 are met. If so, an ECLF is found; otherwise we should continue iteration (24). If the maximum iteration number N_{max} is reached, we should reduce ϵ and restart iteration (24). Since V^ϵ_N converges exponentially fast, N_{max} can usually be chosen rather small. The above procedure of constructing an ECLF is summarized in Algorithm 2. This algorithm is computationally efficient and guarantees to yield an ECLF provided that ϵ_{min} is sufficiently small and N_{max} is sufficiently large.

6 Numerical Example

Consider the same two-mode switched system as defined in (25). Neither of the subsystems is stabilizable by itself. However, this switched system is stabilizable

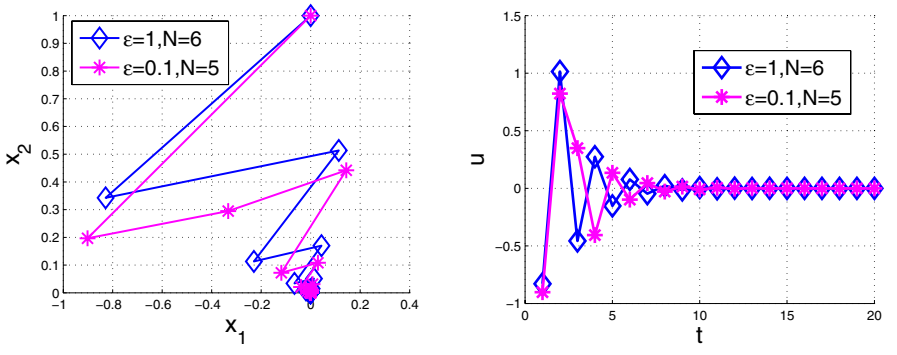


Fig. 2. Simulation Results. Left figure: phase-plane trajectories generated by the ECLFs V_6^1 and $V_5^{0.1}$ starting from the same initial condition $x_0 = [0, 1]^T$. Right figure: the corresponding continuous controls.

through a proper hybrid control. The stabilization problem can be easily solved using Algorithm 2. If we start from $\epsilon = 1$, then the algorithm terminates after 5 steps which results in an ECLF V_6^1 defined by the relaxed switched Riccati set $\mathcal{H}_6^1$. We have also tried a smaller relaxation $\epsilon = 0.1$. In this case, the algorithm stops after 4 steps resulting in an ECLF $V_5^{0.1}$ defined by the relaxed switched Riccati set $\mathcal{H}_5^{0.1}$. It is worth mentioning that $\mathcal{H}_6^1$ contains only two matrices and $\mathcal{H}_5^{0.1}$ contains 3 matrices. With these matrices, starting from any initial position x_0, the feedback laws corresponding to $\mathcal{H}_6^1$ and $\mathcal{H}_5^{0.1}$ can be easily computed using equation (3). The closed-loop trajectories generated by these two feedback laws starting from the same initial position $x_0 = [0, 1]^T$ are plotted on the left of Fig. 2. On the right of the same figure, the continuous control signals associated with the two trajectories are plotted. In both cases, the switching signals jump to the other mode at every time step and are not shown in the figure. It can also be seen that the ECLF $V_5^{0.1}$ stabilizes the system with a faster convergence speed and a smaller control energy than V_6^1. This is because it has a smaller relaxation ϵ which makes the resulting trajectory closer to the optimal trajectory of the DSLQR problem.

7 Conclusion

This paper studies the exponential stabilization problem for the discrete-time switched linear system. It has been proved that if the system is exponentially stabilizable, then there must exist a piecewise quadratic ECLF. More importantly, this ECLF can be chosen to be a finite-horizon value function of the switched LQR problem. An efficient algorithm has been developed to compute such an ECLF and the corresponding stabilizing policy whenever the system is exponentially stabilizable. Indicated by a numerical example, the ECLF and the stabilizing policy can usually be characterized by only a few p.s.d. matrices which

can be easily computed using the relaxed switched Riccati mapping. Future research will focus on extending the algorithm to solve the robust stabilization problem for uncertain switched linear systems.

References

1. Liberzon, D., Morse, A.S.: Basic problems in stability and design of switched systems. IEEE Control Systems Magazine 19(5), 59–70 (1999)
2. DeCarlo, R., Branicky, M., Pettersson, S., Lennartson, B.: Perspectives and results on the stability and stabilizability of hybrid systems. Proceedings of IEEE, Special Issue on Hybrid Systems 88(7), 1069–1082 (2000)
3. Branicky, M.S.: Multiple Lyapunov functions and other analysis tools for switched and hybrid systems. IEEE Transactions on Automatic Control 43(4), 475–482 (1998)
4. Skafidas, E., Evans, R.J., Savkin, A.V., Petersen, I.R.: Stability results for switched controller systems. Automatica 35(12), 553–564 (1999)
5. Pettersson, S.: Synthesis of switched linear systems. In: IEEE Conference on Decision and Control, Maui, HI, pp. 5283–5288 (December 2003)
6. Pettersson, S.: Controller design of switched linear systems. In: Proceedings of the American Control Conference, Boston, MA, pp. 3869–3874 (June 2004)
7. Hai, L., Antsaklis, P.J.: Switching stabilization and L_2 gain performance controller synthesis for discrete-time switched linear systems. In: IEEE Conference on Decision and Control, San Diego, CA, pp. 2673–2678 (December 2006)
8. Hai, L., Antsaklis, P.J.: Hybrid H_∞ state feedback control for discrete-time switched linear systems. In: IEEE 22nd International Symposium on Intelligent Control, Singapore, pp. 112–117 (October 2007)
9. Wicks, M.A., Peleties, P., DeCarlo, R.A.: Switched controller synthesis for the quadratic stabilization of a pair of unstable linear systems. European J. Control 4(2), 140–147 (1998)
10. Pettersson, S., Lennartson, B.: Stabilization of hybrid systems using a min-projection strategy. In: Proceedings of the American Control Conference, Arlington, VA, pp. 223–228 (June 2001)
11. Albertini, F., Sontag, E.D.: Continuous control-Lyapunov functions for asymptotically controllable time-varying systems. Internat. J. Control 72(18), 1630–1641 (1999)
12. Kellett, C.M., Teel, A.R.: Discrete-time asymptotic controllability implies smooth control-Lyapunov function. Systems & Control Letters 52(5), 349–359 (2004)
13. Zhang, W., Hu, J.: On Optimal Quadratic Regulation for Discrete-Time Switched Linear Systems. In: International Workshop on Hybrid Systems: Computation and Control, St Louis, MO, USA, pp. 584–597 (2008)
14. Zhang, W., Hu, J.: On the value functions of the optimal quadratic regulation problem for discrete-time switched linear systems. In: IEEE Conference on Decision and Control, Cancun, Mexico (December 2008)
15. Zhang, W., Hu, J., Abate, A.: Switched LQR problem in discrete time: Theory and algorithms. Submitted IEEE Transactions on Automatic Control (available upon request) (2008)
16. Khalil, H.K.: Nonlinear Systems. Prentice-Hall, Englewood Cliffs (2002)

17. Zhang, W., Hu, J.: Stabilization of switched linear systems with unstabilizable subsystems. Technical report, Purdue University, TR ECE 09-01 (2008)
18. Bertsekas, D.P.: Dynamic Programming and Optimal Control, 2nd edn., vol. 2. Athena Scientific (2001)
19. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, New York (2004)

Bounded and Unbounded Safety Verification Using Bisimulation Metrics*

Gang Zheng and Antoine Girard

Laboratoire Jean Kuntzmann, Université de Grenoble
{Gang.Zheng,Antoine.Girard}@imag.fr

Abstract. In this paper, we propose an algorithm for bounded safety verification for a class of hybrid systems described by metric transition systems. The algorithm combines exploration of the system trajectories and state space reduction using merging based on a bisimulation metric. The main novelty compared to an algorithm presented recently by Lerda et.al. lies in the introduction of a tuning parameter that makes it possible to increase the performances drastically. The second significant contribution of this work is a procedure that allows us to derive, in some cases, a proof of unbounded safety from a proof of bounded safety via a refinement step. We demonstrate the efficiency of the approach via experimental results.

1 Introduction

Formal methods are now well established in the domain of embedded systems, both for software and hardware design. Model checking [1] has been used successfully in various industrial settings. Still, algorithmic verification of computing systems embedded in a physical environment, involving interactions of discrete and continuous dynamics, remains a great challenge. Several approaches have emerged from hybrid systems research, ranging from approximate reachability analysis, to abstraction techniques and safety certificates. We refer the reader to the proceedings of the conference on *Hybrid Systems: Computation and Control* for a fair overview of the area. Recently, several safety verification techniques based on exploration of individual trajectories of a system have been proposed [2,3,4,5,6], arguing the moderate cost of the numerical simulation of individual trajectories relatively to the complex computations involved in the previously mentioned techniques.

In this paper, we first consider the problem of bounded safety verification for a class of hybrid systems described by metric transition systems. We propose an algorithm that determines if there exists a trajectory of bounded length that can reach a set of unsafe states. It combines exploration of the system trajectories and state space reduction using merging based on a bisimulation metric [3,7]. Intuitively, a bisimulation metric measures how far two states of the system are from being bisimilar. A similar approach has recently been proposed in [6]; there

* This work was supported by the ANR SETIN project VAL-AMS.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 426–440, 2009.

are significant differences though. The most important one lies in the introduction of a tuning parameter ρ that determines the opportunity of merging states. For $\rho = 0$, states are never merged and we make an exhaustive exploration of the trajectories of the system. For $\rho = 1$, states are merged whenever it is possible and we essentially get the algorithm presented in [6]. Interestingly, intermediate choices for the parameter ρ may increase drastically the performances of the algorithm. The second significant contribution is that we establish a procedure that makes it possible, in some cases, to derive a proof of unbounded safety (no trajectory of any length can reach the set of unsafe states) from a proof of bounded safety via a refinement step. Finally, we use our approach for the verification of a discrete-time switched linear system where the set of admissible switching sequences is specified by an automaton. We demonstrate the efficiency of the approach via experimental results.

2 Problem Formulation

We first introduce the class of metric transition systems and the associated bisimulation metrics. Then, we define the bounded and unbounded safety properties.

2.1 Metric Transition Systems

In this paper, we will use metric transition systems as an abstract formalism for describing hybrid systems. Essentially, metric transition systems are "classical" transition systems whose set of states is equipped with a pseudo-metric[1].

Definition 1. *A metric transition system (MTS) is a tuple* $T = (Q, \rightarrow, Q_0, d)$ *consisting of:*

- *A set of states* Q,
- *A transition relation* $\rightarrow \subseteq Q \times Q$,
- *A set of initial states* $Q \subseteq Q_0$,
- *A pseudo-metric* $d : Q \times Q \rightarrow \mathbb{R}^+ \cup \{+\infty\}$

A transition $(q, q') \in \rightarrow$ will be denoted $q \rightarrow q'$. Let us remark that the set of states can be discrete, continuous or hybrid. For all $q \in Q$, we will denote

$$\mathsf{succ}(q) = \{q' \in Q |\ q \rightarrow q'\}.$$

For simplicity, we shall assume that the system is *non-blocking* (for all $q \in Q$, $\mathsf{succ}(q)$ has at least one element) and possibly *non-deterministic* ($\mathsf{succ}(q)$ may have more than one element). We will further assume that the system is *finitely branching* ($\mathsf{succ}(q)$ have a finite number of elements).

A *trajectory* of a MTS is a finite sequence of states $s = q_0 \dots q_K$ such that $q_k \rightarrow q_{k+1}$, for all $0 \leq k < K - 1$. K is referred to as the *length* of s. In addition, we say that s is *initialized* if $q_0 \in Q_0$.

[1] A pseudo-metric over a set Q is a function $d : Q \times Q \rightarrow \mathbb{R}^+ \cup \{+\infty\}$ which satisfies the following properties: for all $q_1, q_2, q_3 \in Q$, (i) $d(q_1, q_1) = 0$, (ii) $d(q_1, q_2) = d(q_2, q_1)$, (iii) $d(q_1, q_3) \leq d(q_1, q_2) + d(q_2, q_3)$.

Example 1. Let us introduce a simple example that we will use for illustration throughout the paper. We consider a discrete-time switched linear system where the set of admissible switching sequences is specified by an automaton. Let $T = (Q, \rightarrow, Q_0, d)$ be a MTS where the hybrid set of states is $Q = \{1, 2, 3, 4\} \times \mathbb{R}^2$. For two states, $q = (\sigma, x)$ and $q' = (\sigma', x')$ there is a transition $q \rightarrow q'$ if and only if

$$\begin{cases} \sigma = 1 \text{ and } \sigma' = 2 & \text{and } x' = A_1 x + b_1 \\ \sigma = 2 \text{ and } \sigma' = 3 & \text{and } x' = A_2 x + b_2 \\ \sigma = 3 \text{ and } \sigma' \in \{1, 4\} & \text{and } x' = A_2 x + b_2 \\ \sigma = 4 \text{ and } \sigma' \in \{1, 3\} & \text{and } x' = A_2 x + b_2 \end{cases}$$

where A_1, A_2, b_1 and b_2 are matrices and vectors given by :

$$A_1 = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}, \; A_2 = \begin{pmatrix} 0 & 2 \\ 0.1 & 0 \end{pmatrix}, \; b_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \; b_2 = \begin{pmatrix} -1 \\ 0.9 \end{pmatrix}.$$

We can see that the system is non-deterministic. The set of initial states is $Q_0 = \{(1, x_0)\}$ with $x_0 = (0\ 0)^T$ and the pseudo-metric over the set of states is $d(q, q') = \|x - x'\|$ where $q = (\sigma, x)$ and $q' = (\sigma', x')$. The MTS T is represented in Figure 1. This system belongs to a class of models that has been studied recently in [8] in the context of control mode scheduling for switched systems.

2.2 Bisimulation Metrics

We first define the notion of bisimulation relation [9] in the framework of metric transition systems.

Definition 2. *A relation $\sim \subseteq Q \times Q$ is a bisimulation relation for the MTS $T = (Q, \rightarrow, Q_0, d)$ if for all $q_1 \sim q_2$ the following conditions hold:*

1. *$d(q_1, q_2) = 0$;*
2. *For all $q_1 \rightarrow q_1'$, there exists $q_2 \rightarrow q_2'$ such that $q_1' \sim q_2'$;*
3. *For all $q_2 \rightarrow q_2'$, there exists $q_1 \rightarrow q_1'$ such that $q_1' \sim q_2'$.*

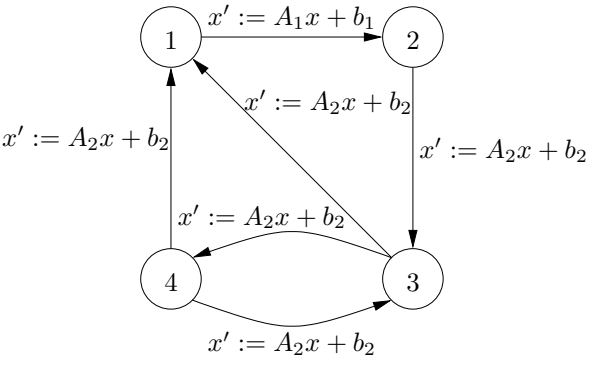


Fig. 1. Example of metric transition system

The notion of bisimulation function has been introduced in [7] as a quantitative generalization of the usual bisimulation relation. In this paper, we use bisimulation metrics which are both pseudo-metrics and bisimulation functions [3]:

Definition 3. *A function $V : Q \times Q \to \mathbb{R}^+ \cup \{+\infty\}$ is a bisimulation metric for the MTS $T = (Q, \to, Q_0, d)$ if the following conditions hold:*

1. *V is a pseudo-metric;*
2. *For all $q_1, q_2 \in Q$,*
$$V(q_1, q_2) \geq d(q_1, q_2); \tag{1}$$
3. *There exists $\lambda \in \mathbb{R}^+$ such that, for all $q_1, q_2 \in Q$,*
$$\lambda V(q_1, q_2) \geq \max_{q_1 \to q_1'} \min_{q_2 \to q_2'} V(q_1', q_2'). \tag{2}$$

If $\lambda < 1$, then the MTS T is said to be contractive with respect to bisimulation.

Remark 1. In the original definition of the bisimulation function, there is, in addition to equations (1) and (2), the symmetrical[2] of equation (2). It is not stated explicitly here because symmetry is implied by V being a pseudo-metric.

It is straightforward to check that the zero set of V is a bisimulation relation for the MTS T. Then, the intuitive meaning of a bisimulation metric is a measure of how far two states are from being bisimilar. Describing a systematic way to compute a bisimulation metric for arbitrary metric transition systems is out of the scope of this paper. However, for the system considered in Example 1, we show how a bisimulation metric can be obtained. A similar approach can be used for other systems of the same class.

Example 2. Let $T = (Q, \to, Q_0, d)$ be the MTS defined in Example 1. The relation given for $q_1 = (\sigma_1, x_1)$, $q_2 = (\sigma_2, x_2)$ by $q_1 \sim q_2$ if and only if

$$(x_1 = x_2) \wedge ([\sigma_1 = \sigma_2 = 1] \vee [\sigma_1 = \sigma_2 = 2] \vee [\sigma_1 \in \{3, 4\} \wedge \sigma_2 \in \{3, 4\}])$$

is a bisimulation relation for the MTS T. Then, let us search for a bisimulation metric of the form

$$V(q_1, q_2) = \begin{cases} \|x_1 - x_2\|_1 & \text{if } \sigma_1 = \sigma_2 = 1; \\ \|x_1 - x_2\|_2 & \text{if } \sigma_1 = \sigma_2 = 2; \\ \|x_1 - x_2\|_3 & \text{if } \sigma_1 \in \{3, 4\} \text{ and } \sigma_2 \in \{3, 4\}; \\ +\infty & \text{for all other cases.} \end{cases}$$

where the norms $\|.\|_i$ $(i = 1, 2, 3)$ are given by $\|x\|_i = \sqrt{x^T M_i x}$ with M_i positive definite symmetric matrices. Let us remark that the zero set of V is the bisimulation relation $\sim$. It is easy to check that V is a pseudo-metric. Equations (1) and (2) in Definition 3 translate to the following linear matrix inequalities :

$$M_i \succeq I, \ i = 1, 2, 3$$

and

[2] $\lambda V(q_1, q_2) \geq \max_{q_2 \to q_2'} \min_{q_1 \to q_1'} V(q_1', q_2')$.

$$\begin{cases} \lambda^2 M_1 \succeq A_1^T M_2 A_1, \\ \lambda^2 M_2 \succeq A_2^T M_3 A_2, \\ \lambda^2 M_3 \succeq A_2^T M_3 A_2 \text{ and } \lambda^2 M_3 \succeq A_2^T M_1 A_2. \end{cases}$$

This set of inequalities can be solved using semidefinite programming. For $\lambda = \sqrt{0.8}$, we find the following matrices

$$M_1 = \begin{pmatrix} 6.25 & 0 \\ 0 & 1.25 \end{pmatrix}, \; M_2 = \begin{pmatrix} 1 & 0 \\ 0 & 5 \end{pmatrix}, \; M_3 = \begin{pmatrix} 1 & 0 \\ 0 & 31.25 \end{pmatrix}.$$

Since $\lambda < 1$, the MTS T is contractive with respect to bisimulation.

2.3 Bounded/Unbounded Safety

In this paper, we will only consider safety verification whose objective is to determine whether there exists a trajectory reaching a predefined undesirable region of the set of states.

Definition 4. *Given a MTS $T = (Q, \to, Q_0, d)$, a set of unsafe states $Q_u \subseteq Q$ and $N \in \mathbb{N}$, we say that:*

- *A state $q \in Q$ is safe for bound N if for every trajectory $s = q_0 \dots q_K$ with $q_0 = q$ and of length $K \leq N$, we have $q_k \notin Q_u$, for all $0 \leq k \leq K$. The MTS T is safe for bound N, if all $q_0 \in Q_0$ are safe for bound N.*
- *A state $q \in Q$ satisfies the unbounded safety property if for every trajectory $s = q_0 \dots q_K$ with $q_0 = q$, we have $q_k \notin Q_u$, for all $0 \leq k \leq K$. The MTS T satisfies the unbounded safety property if all $q_0 \in Q_0$ satisfy the unbounded safety property.*

In the following, we will assume that the set of initial states Q_0 has only a finite number of elements. Since in addition, we consider finitely-branching systems, there exists only a finite number of trajectories of length N or less. Then, the bounded safety property can always be verified by complete exploration of the trajectories of the system. However, for non-deterministic systems, this approach may be computationally untractable as the number of trajectories to be explored generally grows exponentially with respect to the bound N. Further, since the set of states Q can be infinite, it is generally not possible to verify the unbounded safety property by exploring all reachable states. In the following section, we will show how the use of bisimulation metrics makes it possible to overcome these limitations.

3 Safety Verification Using Bisimulation Metrics

In this section, we propose an algorithm for bounded safety verification that does not explore completely the trajectories of the system. In some cases, it allows us to prove unbounded safety.

Definition 5. *Let V be a bisimulation metric, $q \in Q$, $\delta \in \mathbb{R}^+$, the neighborhood $[q]_\delta \subseteq Q$ is defined by*

$$[q]_\delta = \{r \in Q |\ V(q,r) < \delta\}.$$

We say that $[q]_\delta$ is safe for bound $N \in \mathbb{N}$, if all $r \in [q]_\delta$ are safe for bound N.

Our algorithm combines exploration of the system trajectories and state space reduction using merging. It works as follows. We start the exploration of the trajectories of T in a depth-first search manner; for each state q, whose safety is verified for some bound M, we compute a neighborhood $[q]_\delta$ safe for bound M. When reaching a new state q, if we previously determined a neighborhood $[p]_\gamma$ safe for a bound L such that $q \in [p]_\gamma$, the safety of q for a bound $M \leq L$ can be determined without exploring further the trajectories starting from q. This is the merging operation.

A similar approach, inspired by the classical explicit state model checking algorithm [1], combining depth-first search and merging based on proximity has recently been proposed in [6]. There are significant differences with our work though. Firstly, we use bisimulation metrics instead of bisimulation functions. The fact that we use metrics allows us to replace some geometrical considerations involving online numerical optimization problems by a simple use of the triangular inequality. Secondly and more importantly, we add a tuning parameter $\rho \in [0,1]$ to the algorithm that determines the opportunity to merge states. If there is a neighborhood $[p]_\gamma$ safe for a bound L, such that $V(p,q) < \rho\gamma$, then q is safe for all bounds $M \leq L$ and we do not explore the trajectories starting from q. Let us remark, that if $\rho\gamma \leq V(p,q) < \gamma$, then q is safe as well; however, in that case we shall explore the trajectories starting from q. For $\rho = 0$ (i.e. states are never merged), we make an exhaustive exploration of the trajectories of the system. For $\rho = 1$ (i.e. states are merged whenever it is possible), we essentially get, as a special case, the algorithm presented in [6]. Interestingly, intermediate choices for the parameter ρ may increase drastically the performances of the algorithm as will be shown later. Another significant contribution compared to [6] is that we establish a procedure that makes it possible, in some cases, to derive a proof of unbounded safety from a proof of bounded safety.

Bisimulation metrics have first been used for bounded safety verification in [3]. However, the algorithm presented in [3] is quite different as it uses breadth-first search and neighborhood splitting. Its performances are far behind those of the algorithm presented in this paper.

3.1 Bounded Safety Verification Algorithm

Algorithm 1 verifies whether a MTS $T = (Q, \rightarrow, Q_0, d)$ is safe for a given bound N. It returns "Safe" if T is safe for bound N. It returns ("Unsafe", s) if T is unsafe for bound N where $s = q_0 \dots q_K$ is a counterexample (i.e. an initialized trajectory of T of length $K \leq N$ and such that $q_K \in Q_u$).

The algorithm calls recursively the function check_safety which verifies if a state $q \in Q$ is safe for a bound $M \leq N$. If q is not safe for bound M, then

the function returns ("Unsafe", s) where s is a counterexample. Otherwise, it returns ("Safe", $[p]_\gamma, L$) where the pair $([p]_\gamma, L)$ is such that the neighborhood $[p]_\gamma$ is safe for bound $L \geq M$ and $q \in [p]_\gamma$.

We maintain a list of safe neighborhoods (with their safety bound) stored in the global variable $\mathcal{N}$. Initially, $\mathcal{N}$ is empty. Each time a safe neighborhood is determined, it is added to $\mathcal{N}$. We compute a transition relation over the set of safe neighborhoods $\mathcal{N}$, denoted $\rightsquigarrow$. Additionally, we compute the function $\mathsf{merge} : \mathcal{N} \to 2^{\mathcal{Q}}$ that keeps track of merging operations, and the function $\mathsf{num} : \mathcal{N} \to \mathbb{N}$ that records in which order the safe neighborhoods have been computed. Though the transition relation $\rightsquigarrow$ and functions merge, num are not technically necessary for bounded safety verification, they help for the proof of correctness and will be useful for the extension to unbounded safety verification.

The function check_safety works as follows. Given a state q and a bound M, there are four possible cases:

1. If $q \in Q_u$, then q is not safe. The trajectory of length 0 consisting of q is a counterexample. The function returns ("Unsafe", q).
2. If there exists $([p]_\gamma, L) \in \mathcal{N}$, such that $M \leq L$ and $V(p, q) < \rho\gamma$, then q is safe for bound M and we merge the states. We insert q in $\mathsf{merge}([p]_\gamma, L)$ and the function returns ("Safe", $[p]_\gamma, L$).
3. If $M = 0$ and $q \notin Q_u$, then q is safe for bound 0. Let δ be given by
$$\delta = d(q, Q_u) = \inf_{r \in Q_u} d(q, r). \tag{3}$$
We insert $([q]_\delta, 0)$ in $\mathcal{N}$, we set $\mathsf{merge}([q]_\delta, 0) = \{q\}$ and the function returns ("Safe", $[q]_\delta, 0$).
4. In the other cases, we need to check the safety for bound $M - 1$ of the successors of q:
   - If one of the successors q' is not safe for bound $M-1$, then the function $\mathsf{check\_safety}(q', M-1)$ returns a counterexample s. In that case, q is not safe for bound M and a counterexample is given by the trajectory qs. The function returns ("Unsafe", qs).
   - If all the successors $q'_1, \ldots, q'_n$ of q are safe for bound $M - 1$, then q is safe for bound M. Let δ be given by
$$\delta = \min\left(d(q, Q_u), \frac{\gamma'_1 - V(p'_1, q'_1)}{\lambda}, \ldots, \frac{\gamma'_n - V(p'_n, q'_n)}{\lambda}\right) \tag{4}$$
   where the pair $([p'_i]_{\gamma'_i}, L'_i) \in \mathcal{N}$ is returned by $\mathsf{check\_safety}(q'_i, M-1)$, for $i = 1, \ldots, n$. We insert $([q]_\delta, M)$ in $\mathcal{N}$, set $\mathsf{merge}([q]_\delta, M) = \{q\}$ and $([q]_\delta, M) \rightsquigarrow ([p'_i]_{\gamma'_i}, L'_i)$, for $i = 1, \ldots, n$. Then, the function returns ("Safe", $[q]_\delta, M$).

Remark 2. We want to point out the following simple but useful properties. Firstly, for all transitions $([q]_\delta, M) \rightsquigarrow ([p']_{\gamma'}, L')$, we have, by construction, that $L' \geq M - 1$ and $\mathsf{num}([q]_\delta, M) > \mathsf{num}([p']_{\gamma'}, L')$. Secondly, for all $([q]_\delta, M) \in \mathcal{N}$, with $M \geq 1$, for all $q \to q'$, there exists, by construction, $([q]_\delta, M) \rightsquigarrow ([p']_{\gamma'}, L')$, such that $q' \in \mathsf{merge}([p']_{\gamma'}, L')$.

Algorithm 1. Bounded safety verification algorithm

Input: MTS T, unsafe set Q_u, bisimulation metric V and bound N.
Output: "Safe" or ("Unsafe", s) where s is a counterexample.

Global $\mathcal{N} \leftarrow \emptyset$; $i \leftarrow 0$;
Global $\rightsquigarrow \subseteq \mathcal{N} \times \mathcal{N}$; $\mathsf{merge} : \mathcal{N} \rightarrow 2^Q$; $\mathsf{num} : \mathcal{N} \rightarrow \mathbb{N}$;

Main: ▷ Main procedure to check bounded safety of MTS T
for each $q \in Q_0$ **do**
  result $\leftarrow \mathsf{check\_safety}(q, N)$;
  if result=("Unsafe", s) **then**
    return result;
  end if
end for
return "Safe";

function $\mathsf{check\_safety}(q, M)$ ▷ Check whether q is safe for bound M
if $q \in Q_u$ **then** ▷ q is in the unsafe set
  return ("Unsafe", q);
else if $\exists([p]_\gamma, L) \in \mathcal{N}$ such that $M \leq L$ and $V(p,q) < \rho\gamma$ **then**
  $\mathsf{merge}([p]_\gamma, L) \leftarrow \mathsf{merge}([p]_\gamma, L) \cup \{q\}$; ▷ Merging
  return ("Safe", $[p]_\gamma, L$);
else if $M = 0$ **then** ▷ q is safe for bound 0
  $\delta \leftarrow d(q, Q_u)$;
  $\mathcal{N} \leftarrow \mathcal{N} \cup \{([q]_\delta, 0)\}$; $i \leftarrow i + 1$;
  $\mathsf{merge}([q]_\delta, 0) \leftarrow \{q\}$; $\mathsf{num}([q]_\delta, 0) \leftarrow i$;
  return ("Safe", $[q]_\delta, 0$);
else ▷ Need to explore the successors of q
  $\delta \leftarrow d(q, Q_u)$; $\mathcal{S} \leftarrow \emptyset$;
  for each $q' \in \mathsf{succ}(q)$ **do**
    result$\leftarrow \mathsf{check_safety}(q', M-1)$;
    if result=("Unsafe", s) **then**
      return("Unsafe", qs);
    else ▷ Then result=("Safe", $[p']_{\gamma'}, L'$)
      $\delta \leftarrow \min\left(\delta, \frac{\gamma' - V(p',q')}{\lambda}\right)$;
      $\mathcal{S} \leftarrow \mathcal{S} \cup \{([p']_{\gamma'}, L')\}$;
    end if
  end for
  $\mathcal{N} \leftarrow \mathcal{N} \cup \{([q]_\delta, M)\}$; $i \leftarrow i + 1$;
  $\mathsf{merge}([q]_\delta, M) \leftarrow \{q\}$; $\mathsf{num}([q]_\delta, M) \leftarrow i$;
  for each $([p']_{\gamma'}, L') \in \mathcal{S}$ **do**
    Set $([q]_\delta, M) \rightsquigarrow ([p']_{\gamma'}, L')$;
  end for
  return ("Safe", $[q]_\delta, M$);
end if

Lemma 1. *Let* $([q]_\delta, M) \in \mathcal{N}$ *and* $r \in Q$*, such that* $r \in [q]_\delta$*, then:*

- $r \notin Q_u$*;*
- *if* $M \geq 1$*, for all* $r \to r'$*, there is* $([q]_\delta, M) \rightsquigarrow ([p']_{\gamma'}, L')$ *such that* $r' \in [p']_{\gamma'}$*.*

Proof. From equations (3) and (4), we have that $\delta \leq d(q, Q_u)$. Then, from equation (1), $d(q,r) \leq V(q,r) < \delta \leq d(q, Q_u)$ which implies that $r \notin Q_u$. The first property holds. Let us assume that $M \geq 1$ and let $r \to r'$, from equation (2), there exists $q \to q'$, such that $V(q', r') \leq \lambda V(q,r) < \lambda\delta$. From equation (4), there exists $([q]_\delta, M) \rightsquigarrow ([p']_{\gamma'}, L')$ such that $\delta \leq (\gamma' - V(p', q'))/\lambda$. Then, it follows from the triangular inequality

$$V(p', r') \leq V(p', q') + V(q', r') < V(p', q') + \lambda\delta \leq \gamma'.$$

Hence, $r' \in [p']_{\gamma'}$ and the second property holds. ■

The correctness of algorithm 1 is a direct consequence of Lemma 1.

Theorem 1. *Algorithm 1 is correct: if it returns ("Unsafe", s) then T is not safe for bound N and the trajectory s is counterexample; if it returns "Safe", then the MTS T is safe for bound N.*

Proof. The proof of the first part of the theorem is straightforward. Let us assume that the algorithm returns "Safe". Let $r_0 \dots r_K$ be an initialized trajectory of T of length $K \leq N$. Since algorithm 1 returns "Safe", there exists $([q_0]_{\delta_0}, N) \in \mathcal{N}$, such that $r_0 \in [q_0]_{\delta_0}$. Let us prove, by induction, that for all $k \in \{0, \dots, K\}$, there exists $([q_k]_{\delta_k}, M_k) \in \mathcal{N}$ such that $r_k \in [q_k]_{\delta_k}$ and $M_k \geq N - k$. This is clearly true for $k = 0$. Let us assume this is true for some $k \in \{0, \dots, K-1\}$ and show that it is true for $k+1$. We have $r_k \to r_{k+1}$, then from Lemma 1, and since $M_k \geq N - k \geq 1$, it follows that there exists $([q_k]_{\delta_k}, M_k) \rightsquigarrow ([q_{k+1}]_{\delta_{k+1}}, M_{k+1})$ such that $r_{k+1} \in [q_{k+1}]_{\delta_{k+1}}$. In addition, by remark 2, we have $M_{k+1} \geq M_k - 1 \geq N - k - 1$. This completes the induction. Further, from Lemma 1, it follows that $r_k \notin Q_u$ for $k \in \{0, \dots, K\}$. Therefore, r_0 is safe for bound N and the MTS T is safe for bound N as well. ■

Remark 3. The termination of algorithm 1 is an obvious consequence of T being finitely branching with a finite set of initial states.

3.2 Unbounded Safety Verification Result

We now move to an interesting result that makes it possible, in some cases, to derive a proof of unbounded safety from the proof of bounded safety provided by algorithm 1. The main idea is the following. Let us assume that a MTS T has been proved safe for some bound N using algorithm 1 and that for all $([p]_\gamma, 0) \in \mathcal{N}$, there exists $([q]_\delta, M) \in \mathcal{N}$ with $M \geq 1$ such that $[p]_\gamma \subseteq [q]_\delta$. Thus, the neighborhoods $[p]_\gamma$ which are safe for bound 0, are included in neighborhoods $[q]_\delta$ safe for bound $M \geq 1$. This implies that the neighborhoods $[p]_\gamma$ are safe for bound 1 which in turn implies that the neighborhoods $[q]_\delta$ are safe for bound $M+1$. Then, by induction, it can be shown that the MTS T satisfies the unbounded

safety property. Unfortunately, the elements of $\mathcal{N}$ generally do not satisfy the previous condition. The reason is that for all $([p]_\gamma, 0) \in \mathcal{N}$, we have $\gamma = d(p, Q_u)$. Then, $[p]_\gamma$ is generally not included in another safe neighborhood $[q]_\delta$, since such a neighborhood would most probably intersect Q_u which contradicts the fact that it is safe.

Therefore, prior to apply the simple test described previously, we need to refine the neighborhoods in $\mathcal{N}$. In Algorithm 1, the safe neighborhoods are computed backward, the safe neighborhood around a state is determined from the safe neighborhoods around its successors. Our refinement step consists in reevaluating forward these safe neighborhoods. We keep the safe neighborhoods around the initial states unchanged. Then, the safe neighborhoods around the other states are updated according to the safe neighborhoods around their predecessors. More precisely, we use the function $\mathsf{refine} : \mathcal{N} \to \mathbb{R}^+$ defined recursively as follows. Let $([p']_{\gamma'}, L') \in \mathcal{N}$, there are two different cases:

1. If $L' = N$, then $([p']_{\gamma'}, L')$ corresponds to an initial state of T, we keep it unchanged and $\mathsf{refine}([p']_{\gamma'}, L') = \gamma'$.
2. If $L' \neq N$, we denote $([q_1]_{\delta_1}, M_1), \ldots, ([q_n]_{\delta_n}, M_n)$ the elements of $\mathcal{N}$ such that $([q_i]_{\delta_i}, M_i) \rightsquigarrow ([p']_{\gamma'}, L')$, $i = 1, \ldots, n$. Then,

$$\mathsf{refine}([p']_{\gamma'}, L') = \max_{i=1}^{n} \max_{j=1}^{m_i} \left(\lambda \mathsf{refine}([q_i]_{\delta_i}, M_i) + V(q'_{i,j}, p')\right). \tag{5}$$

   where $\{q'_{i,1}, \ldots, q'_{i,m_i}\} = \mathsf{succ}(q_i) \cap \mathsf{merge}([p']_{\gamma'}, L')$.

Lemma 2. *For all* $([p']_{\gamma'}, L') \in \mathcal{N}$, $\mathsf{refine}([p']_{\gamma'}, L')$ *is well defined and satisfies* $\mathsf{refine}([p']_{\gamma'}, L') \leq \gamma'$.

Proof. The proof is done by induction on $\mathsf{num}([p']_{\gamma'}, L')$. For $([p']_{\gamma'}, L') \in \mathcal{N}$, $\mathsf{num}([p']_{\gamma'}, L')$ ranges from 1 to $\mathrm{Card}(\mathcal{N})$. Further, it is easy to see that the last pair $([p']_{\gamma'}, L')$ added by algorithm 1, which verifies $\mathsf{num}([p']_{\gamma'}, L') = \mathrm{Card}(\mathcal{N})$ is such that $L' = N$. In that case, $\mathsf{refine}([p']_{\gamma'}, L') = \gamma'$. Now, let us assume that there exists $k \in \{2, \ldots, \mathrm{Card}(\mathcal{N})\}$ such that for all $([p']_{\gamma'}, L') \in \mathcal{N}$, satisfying $\mathsf{num}([p']_{\gamma'}, L') \geq k$, $\mathsf{refine}([p']_{\gamma'}, L')$ is well defined and $\mathsf{refine}([p']_{\gamma'}, L') \leq \gamma'$. Let us remark that this holds for $k = \mathrm{Card}(\mathcal{N})$. Let us prove that this true for $k-1$. Let $([p']_{\gamma'}, L') \in \mathcal{N}$, such that $\mathsf{num}([p']_{\gamma'}, L') = k - 1$, there are two possible cases. If $L' = N$, $\mathsf{refine}([p']_{\gamma'}, L') = \gamma'$ and the property holds. If $L' \neq N$, then for all $([q_i]_{\delta_i}, M_i) \rightsquigarrow ([p']_{\gamma'}, L')$, we have from remark 2, that $\mathsf{num}([q_i]_{\delta_i}, M_i) > \mathsf{num}([p']_{\gamma'}, L')$, hence by the induction assumption, we have that $\mathsf{refine}([q_i]_{\delta_i}, M_i)$ is well defined and $\mathsf{refine}([q_i]_{\delta_i}, M_i) \leq \delta_i$. Hence, $\mathsf{refine}([p']_{\gamma'}, L')$ is well defined. Further, from equation (4), we have that for all $q'_{i,j} \in \mathsf{succ}(q_i) \cap \mathsf{merge}([p']_{\gamma'}, L')$,

$$\gamma' \geq \lambda\delta_i + V(q'_{i,j}, p') \geq \lambda \mathsf{refine}([q_i]_{\delta_i}, M_i) + V(q'_{i,j}, p')$$

Since this holds for all $i \in \{1, \ldots, n\}$, for all $j \in \{1, \ldots, m_i\}$, it follows that $\mathsf{refine}([p']_{\gamma'}, L') \leq \gamma'$. This completes the induction. ∎

We define the set of refined neighborhoods:

$$\hat{\mathcal{N}} = \left\{ ([q]_{\hat{\delta}}, M) |\ \hat{\delta} = \mathsf{refine}([q]_\delta, M),\ ([q]_\delta, M) \in \mathcal{N} \right\}.$$

We lift the transition relation $\rightsquigarrow$ from the set $\mathcal{N}$ to the set $\hat{\mathcal{N}}$ by setting $([q]_{\hat{\delta}}, M) \rightsquigarrow ([p']_{\hat{\gamma}'}, L')$ if and only if $([q]_{\delta}, M) \rightsquigarrow ([p']_{\gamma'}, L')$.

Lemma 3. *Let $([q]_{\hat{\delta}}, M) \in \hat{\mathcal{N}}$ and $r \in Q$, such that $r \in [q]_{\hat{\delta}}$, then:*

- *$r \notin Q_u$;*
- *if $M \geq 1$, for all $r \to r'$, there is $([q]_{\hat{\delta}}, M) \rightsquigarrow ([p']_{\hat{\gamma}'}, L')$ such that $r' \in [p']_{\hat{\gamma}'}$.*

Proof. There exists $([q]_{\delta}, M) \in \mathcal{N}$ with $\hat{\delta} = \mathsf{refine}([q]_{\delta}, M)$. From Lemma 2, $\hat{\delta} \leq \delta$, it follows that $[q]_{\hat{\delta}} \subseteq [q]_{\delta}$. Then, from Lemma 1, the first property holds. Let us assume that $M \geq 1$ and let $r \to r'$, from equation (2), there exists $q \to q'$, such that $V(q', r') \leq \lambda V(q, r) < \lambda\hat{\delta}$. According to remark 2, there exists $([q]_{\hat{\delta}}, M) \rightsquigarrow ([p']_{\hat{\gamma}'}, L')$ such that $q' \in \mathsf{merge}([p']_{\gamma'}, L')$. Hence, from equation (5), we have that $\hat{\gamma}' \geq \lambda\hat{\delta} + V(q', p')$. Using the triangular inequality yields

$$V(p', r') \leq V(p', q') + V(q', r') < V(p', q') + \lambda\hat{\delta} \leq \hat{\gamma}'.$$

Hence, $r' \in [p']_{\hat{\gamma}'}$ and the second property holds. ■

We can now state the following unbounded safety verification result:

Theorem 2. *Let us assume that*

$$\forall([p]_{\hat{\gamma}}, 0) \in \hat{\mathcal{N}},\ \exists([q]_{\hat{\delta}}, M) \in \hat{\mathcal{N}} \text{ with } M \geq 1 \text{ such that } [p]_{\hat{\gamma}} \subseteq [q]_{\hat{\delta}}. \tag{6}$$

Then, the MTS T satisfies the unbounded safety property.

Proof. Let $r_0 \dots r_K$ (with $K \in \mathbb{N}$) be an initialized trajectory of T. Since the MTS T has been proved safe for some bound $N \geq 1$ using Algorithm 1, there exists $([q_0]_{\delta_0}, N) \in \mathcal{N}$ such that $r_0 \in [q_0]_{\delta_0}$. We have $\hat{\delta}_0 = \mathsf{refine}([q_0]_{\delta_0}, N) = \delta_0$, therefore $([q_0]_{\hat{\delta}_0}, N) \in \hat{\mathcal{N}}$ and $r_0 \in [q_0]_{\hat{\delta}_0}$. Let us prove, by induction, that for all $k \in \{0, \dots, K\}$, there exists $([q_k]_{\delta_k}, M_k) \in \mathcal{N}$ such that $r_k \in [q_k]_{\delta_k}$ and $M_k \geq 1$. This is clearly true for $k = 0$. Let us assume this is true for some $k \in \{0, \dots, K-1\}$ and show that it is true for $k+1$. We have $r_k \to r_{k+1}$, then from Lemma 3, it follows that there exists $([q_k]_{\hat{\delta}_k}, M_k) \rightsquigarrow ([p_{k+1}]_{\hat{\gamma}_{k+1}}, L_{k+1})$ such that $r_{k+1} \in [p_{k+1}]_{\hat{\gamma}_{k+1}}$. If $L_{k+1} \geq 1$, then the induction hypothesis holds for $k+1$. If $L_{k+1} = 0$, we have from equation (6) that there exists $([q_{k+1}]_{\hat{\delta}_{k+1}}, M_{k+1}) \in \hat{\mathcal{N}}$ with $M_{k+1} \geq 1$ and such that $[p_{k+1}]_{\hat{\gamma}_{k+1}} \subseteq [q_{k+1}]_{\hat{\delta}_{k+1}}$. It follows that $r_{k+1} \in [q_{k+1}]_{\hat{\delta}_{k+1}}$. This completes the induction. Then, from Lemma 3, we have for all $k = 0 \dots K$, $r_k \notin Q_u$. This completes the proof. ■

The inclusion test $[p]_{\hat{\gamma}} \subseteq [q]_{\hat{\delta}}$ in equation (6) can be replaced conservatively by the easily checkable inequality $V(p, q) + \hat{\gamma} \leq \hat{\delta}$.

Remark 4. From equation (5), we can show that there exists $([p]_{\hat{\gamma}}, 0) \in \hat{\mathcal{N}}$ such that for all $([q]_{\hat{\delta}}, M) \in \hat{\mathcal{N}}$, $\hat{\gamma} \geq \lambda^M \hat{\delta}$. This implies that the unbounded safety verification result given by Theorem 2 cannot be applied if T is not contractive with respect to bisimulation.

4 Experimental Results

In this section, we evaluate the performances of our approach by verifying safety properties of the MTS defined in Example 1. The set of unsafe states is parameterized by $\theta \in \mathbb{R}$: $Q_u = \{(\sigma, x) \in Q |\ (0\ 1)x \geq \theta\}$. It can be verified that by choosing the admissible sequence of discrete states $1, 2, 3, 4, 3, 4, 3, 4 \ldots$, the second component of the continuous state approaches asymptotically 1. Hence, we shall choose $\theta \geq 1$.

4.1 Bounded Safety Verification

We start with bounded safety verification. Algorithm 1 has been implemented in Matlab. It was applied for several values of the parameters ρ, N and θ. The results are presented in figures 2, 3 and 4.

We first discuss the importance of the choice of the tuning parameter ρ. Let us recall that for $\rho = 0$, Algorithm 1 explores exhaustively all the trajectories of T of length N. For $\rho = 1$, Algorithm 1 merges states whenever it is possible and is similar to the algorithm presented in [6]. In figure 2, we can see the performances of the algorithm (CPU time and number of neighborhoods in $\mathcal{N}$) for several values of ρ and $N \in \{20, 30, 40\}$, $\theta = 1.1$. As expected, the larger the bound N, the more expensive the verification. What is more surprising is the influence of the tuning parameter ρ. It can be seen that the optimal value lies somewhere around $\rho = 0.2$ and not at $\rho = 1$ as one may intuitively think. This means that sometimes, it is better not to merge states, even though it is be possible.

We shall try to give an explanation by looking at the distribution of the pairs $([q]_\delta, M) \in \mathcal{N}$ as a function of the safety bound M (see figure 3). For larger values of ρ, e.g. 0.6, we can see that the number of pairs with a small value of M is much less than the number of pairs with an intermediate value for M (bell-shaped distribution). The interpretation is the following. With $\rho = 0.6$, Algorithm 1 often uses the opportunity to merge states. This results, for small values of M, in a lot of merging operations and few computed neighborhoods. However, a lot of merging operations for small values of M result, because of

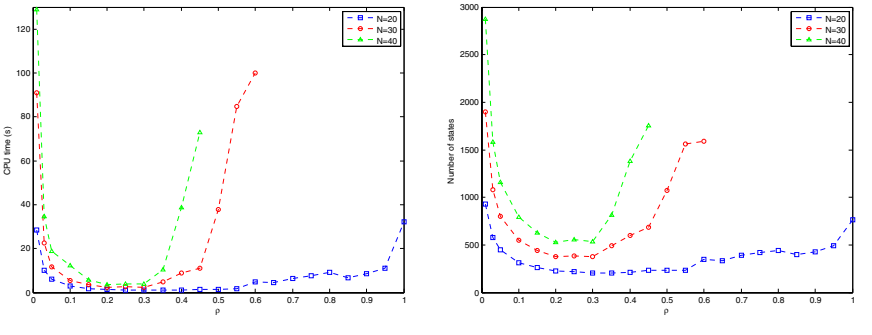


Fig. 2. CPU time and number of elements in $\mathcal{N}$ for several values of ρ and N ($\theta = 1.1$)

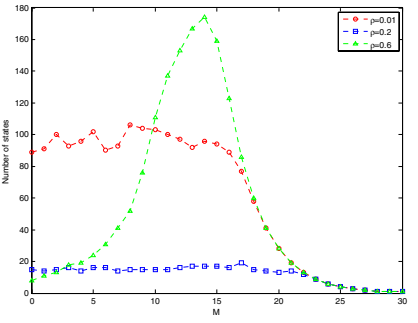


Fig. 3. Distribution of the pairs $([q]_\delta, M) \in \mathcal{N}$ as a function of the safety bound M, for several values of the tuning parameter ρ, $N = 30$, $\theta = 1.1$

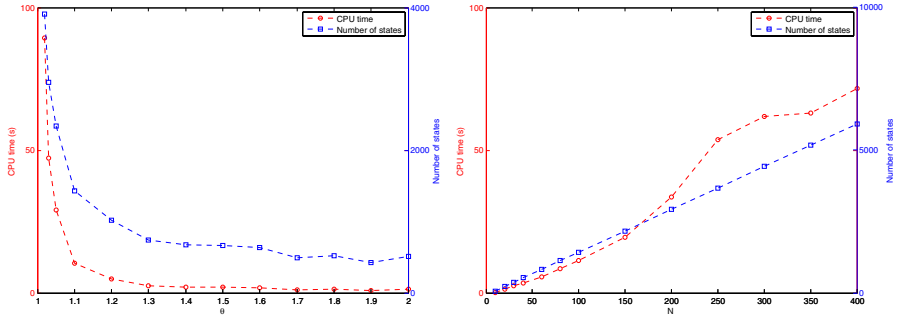


Fig. 4. CPU time and number of elements in $\mathcal{N}$ for several values of θ (left, $N = 100$, $\rho = 0.2$) and N (right, $\theta = 1.1$, $\rho = 0.2$)

backward computations, in safe neighborhoods of smaller size for larger values of M. As the safe neighborhoods become smaller, Algorithm 1 has much less opportunities to merge states and it needs to compute a lot of neighborhoods for intermediate values of M to verify the safety of the system. On the contrary, for smaller values of ρ, e.g. 0.2 and 0.01, the distribution of the pairs looks quite uniform. This means that, by merging states less often when it gets the opportunity, Algorithm 1 receives this opportunity more often. The optimal balance seems to be obtained for $\rho = 0.2$.

For the sake of comparison, we also implemented the exhaustive exploration without testing for merging, which results in much faster computations than Algorithm 1 with $\rho = 0$. It takes 0.4 seconds (4022 explored states) for $N = 20$ and 18.3 seconds (183915 explored states) for $N = 30$, we stopped it before completion for $N = 40$. We can see that even for $N = 30$, our algorithm with $\rho = 0.2$ is already much faster (2.3 seconds) than the exhaustive exploration of the trajectories.

We now discuss the influence of the parameters θ and N (ρ is set to 0.2). In figure 4, we represented the performances of the algorithm (CPU time and number of neighborhoods in $\mathcal{N}$) for several values of θ and N. We can see that as θ approaches 1, Algorithm 1 needs more time to verify the safety of the

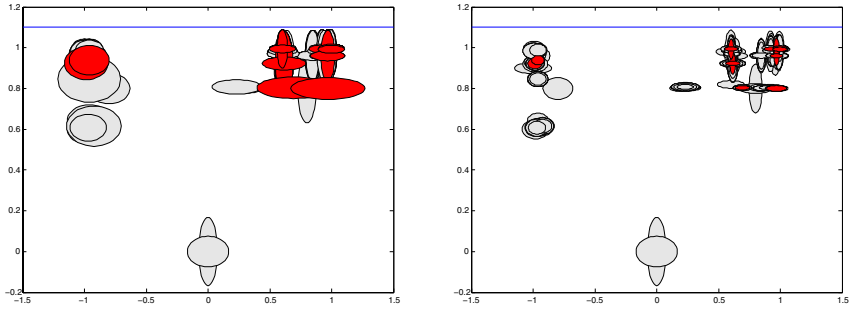


Fig. 5. Set of safe neighborhoods obtained after bounded safety verification (left) and set of refined safe neighborhoods allowing to derive a proof of unbounded safety (right). The darker (red) safe neighborhoods are those corresponding to safety bound $M = 0$. Region above the line is the unsafe set.

system: as θ becomes close to 1, some trajectories of the system are very close to the unsafe set resulting in safe neighborhoods of small size and less merging opportunities. This corroborates the fact pointed out in [3] that robust safety properties are easier to verify. The effect of the bound N is quite surprising. When using exhaustive exploration of the trajectories, the time needed for bounded safety verification grows exponentially with parameter N. With our approach and for the optimal value $\rho = 0.2$, it seems that the time needed for bounded safety verification grows only linearly with N. This is a huge improvement. For comparison, we were able to verify, with our approach, bounded safety for $N = 400$ in a minute whereas we interrupted the exhaustive exploration for $N = 40$ since it was taking too much time.

4.2 Unbounded Safety Verification

We move to unbounded safety verification, the parameter θ is set to 1.1. The bounded safety verification had previously been verified for $N = 30$ with $\rho = 0.2$. Using the result presented in Theorem 2, a proof of unbounded safety could be derived. Most of the computational effort was spent on bounded safety verification so the overall process took less than 3 seconds.

In figure 5, we represented the set of safe neighborhoods obtained after bounded safety verification and the set of safe neighborhoods after refinement. The neighborhoods corresponding to safety bound $M = 0$ are represented in dark (red). We can check that, after refinement, these are included in other safe neighborhoods, thus allowing us to conclude that the system satisfies the unbounded safety property.

5 Conclusion

In this paper, we first presented an algorithm for verifying bounded safety of metric transition systems. In some cases, proofs of unbounded safety can be

derived from the results of our algorithm. We provided experiments that show the efficiency of the approach when the tuning parameter ρ of the algorithm is well chosen. Future work will focus on better understanding the influence of this parameter and extending our approach to infinitely branching transition systems.

References

1. Clarke, E., Grumberg, O., Peled, D.: Model Checking. MIT Press, Cambridge (2000)
2. Kapinski, J., Krogh, B., Maler, O., Stursberg, O.: On systematic simulation of open continuous systems. In: Maler, O., Pnueli, A. (eds.) HSCC 2003. LNCS, vol. 2623, pp. 283–297. Springer, Heidelberg (2003)
3. Girard, A., Pappas, G.J.: Verification using simulation. In: Hespanha, J.P., Tiwari, A. (eds.) HSCC 2006. LNCS, vol. 3927, pp. 272–286. Springer, Heidelberg (2006)
4. Donzé, A., Maler, O.: Systematic simulation using sensitivity analysis. In: Bemporad, A., Bicchi, A., Buttazzo, G. (eds.) HSCC 2007. LNCS, vol. 4416, pp. 174–189. Springer, Heidelberg (2007)
5. Julius, A., Fainekos, G., Anand, M., Lee, I., Pappas, G.: Robust test generation and coverage for hybrid systems. In: Bemporad, A., Bicchi, A., Buttazzo, G. (eds.) HSCC 2007. LNCS, vol. 4416, pp. 329–342. Springer, Heidelberg (2007)
6. Lerda, F., Kapinski, J., Clarke, E., Krogh, B.: Verification of supervisory control software using state proximity and merging. In: Egerstedt, M., Mishra, B. (eds.) HSCC 2008. LNCS, vol. 4981, pp. 344–357. Springer, Heidelberg (2008)
7. Girard, A., Pappas, G.: Approximation metrics for discrete and continuous systems. IEEE Trans. Automatic Control 52(5), 782–798 (2007)
8. Weiss, G., Alur, R.: Automata based interfaces for control and scheduling. In: Bemporad, A., Bicchi, A., Buttazzo, G. (eds.) HSCC 2007. LNCS, vol. 4416, pp. 601–613. Springer, Heidelberg (2007)
9. Milner, R.: Communication and Concurrency. Prentice-Hall, Englewood Cliffs (1989)

The Optimal Boundary and Regulator Design Problem for Event-Driven Controllers*

Pau Martí[1], Manel Velasco[1], and Enrico Bini[2]

[1] Automatic Control Department, Technical University of Catalonia,
Pau Gargallo 5, 08028 Barcelona, Spain
pau.marti@upc.edu
[2] Scuola Superiore Sant'Anna
Pisa, Italy

Abstract. Event-driven control systems provide interesting benefits such as reducing resource utilization. This paper formulates the optimal boundary and regulator design problem that minimizes the resource utilization of an event-driven controller that achieves a cost equal to the case of periodic controllers.

1 Event-Driven Control System Model

We consider the control system

$$\begin{aligned} \dot{x}(t) &= A\,x(t) + B\,u(t) \\ y(t) &= C\,x(t) \end{aligned} \tag{1}$$

with $x \in \mathbb{R}^{n\times 1}$, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $u \in \mathbb{R}^{m\times 1}$, and $C \in \mathbb{R}^{1\times n}$. Let

$$u(t) = u_k = L\,x(a_k) = L\,x_k \qquad \forall t \in [a_k, a_{k+1}[\tag{2}$$

be the control updates given by a linear feedback controller designed in the continuous-time domain but using only samples of the state at discrete instants $a_0, a_1, \ldots, a_k, \ldots$ Between two consecutive control updates, $u(t)$ is held constant. In periodic sampling we have $a_{k+1} = a_k + h$, where h is the period of the controller.

Let $e_k(t) = x(t) - x_k$ be the error evolution between consecutive samples with $t \in [a_k, a_{k+1}[$. For several types of event-driven control approaches [1,2], event conditions can be generalized by introducing a function $f(\cdot,\cdot,\Upsilon) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ that defines a boundary measuring the tolerated error with to respect the sampled state [3]. The condition that must be ensured is

$$f(e_k(t), x_k, \Upsilon) \leq \eta \tag{3}$$

* This work was supported in part by ArtistDesign NoE IST-2008-214373, and by CICYT DIP-2007-61527.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 441–444, 2009.

where η is the error tolerance and $\Upsilon = \{\upsilon_1, \upsilon_2, \ldots, \upsilon_p\}$, $\upsilon_i \in \mathbb{R}$ is a set of free parameters. Hence, we can define the complete dynamics of the event-driven system by the $n+1$ order non linear discrete-time system

$$\begin{aligned} a_{k+1} &= a_k + \Lambda(x_k, \Upsilon, \eta) \\ x_{k+1} &= (\Phi(\Lambda(x_k, \Upsilon, \eta)) + \Gamma(\Lambda(x_k, \Upsilon, \eta))L)x_k \end{aligned} \tag{4}$$

where $\Lambda(x_k, \Upsilon, \eta)$ denotes the time separation between two consecutive activations a_{k+1} and a_k, that solves (1), (2), and (3), assuming that $x_k = x(a_k)$ is the state sampled at a_k, Υ is the set of free parameters of f, and η is the tolerance to the error. We also define $\Phi(t) = e^{At}$ and $\Gamma(t) = \int_0^t e^{As}\mathrm{d}sB$. We highlight that we have been able to find an expression for $\Lambda(x_k, \Upsilon, \eta)$ only by approximating Φ and Γ by Taylor expansion [3]. In all the other cases Λ can only be computed numerically.

2 Optimal Problem Formulation

The optimal problem for event-driven controllers can be formulated in two complementary ways: to minimize the cost while using the same amount of resources than the periodic controller, or to minimize the computational demand while achieving the same cost as in the case of the periodic controller. Here we describe the resource usage minimization given a cost constraint. The other formulation simply requires to exchange the goal function and one constraint, as it will be indicated later.

Let be a standard quadratic cost function in continuous time defined as

$$J(L, \Upsilon, \eta) = \int_0^{a_\ell} x(t)^T Q_c x(t) + u(t)^T R_c u(t)\mathrm{d}t + x(a_\ell)^T N_c x(a_\ell) \tag{5}$$

The optimal boundary and regulator design problem for resource minimization can be formulated as

$$\text{maximize} \quad \frac{\sum_{k=0}^{\ell-1} \Lambda(x_k, \Upsilon, \eta)}{k} \quad \text{w.r.t. } L, \Upsilon, \eta \tag{6}$$

$$\text{subject to} \quad x_{k+1} = (\Phi(\Lambda(x_k, \Upsilon, \eta)) + \Gamma(\Lambda(x_k, \Upsilon, \eta))L)x_k \tag{7}$$

$$a_{k+1} = a_k + \Lambda(x_k, \Upsilon, \eta) \tag{8}$$

$$J(L, \Upsilon, \eta) \leq J_h \tag{9}$$

where (6) sets the maximization goal equal to the average of the first ℓ sampling intervals, (7) enforces the relationship between two consecutive sampled states, (8) describes the constraint among the activations, and J_h is the cost of an optimal h-periodic controller.

Notice that by exchanging (6) with (9) we obtain the complementary problem that minimizes the cost given an upper bound on the period.

The problem (6)–(9) can be numerically solved by constrained minimization techniques such as Lagrange multipliers, or by standards procedures for time varying discrete-time systems.

3 Example

Consider the double integrator system

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \quad , \quad y = \begin{bmatrix} 1 & 0 \end{bmatrix} x.$$

A first closed loop system using a periodic optimal regulator designed using standard methods to minimize (5) with $h = 0.6$s gives a cost of 27.3648 with $L = \begin{bmatrix} -0.6115 & -1.2637 \end{bmatrix}$, where

$$x_0 = \begin{bmatrix} 0.54 \\ 0.84 \end{bmatrix}, Q_c = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}, R_c = \begin{bmatrix} 10 \end{bmatrix}, N_c = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, a_n = 100s.$$

Alternatively, for an event-driven controller, let

$$\dot{x}_{k^+}^T M_1 \dot{x}_{k^+} (a_{k+1} - a_k)^2 = \eta x_k^T M_2 x_k \tag{10}$$

be an execution rule as in (3) that intuitively mandates to trigger more frequent control updates when the state moves fast. In (10) we set

$$\dot{x}_{k^+} = \lim_{t \to a_k^+} \dot{x}(t) = (A + BL) x_k \tag{11}$$

to denote the state derivative once the controller has been applied to the sampled state. From (10), it follows that

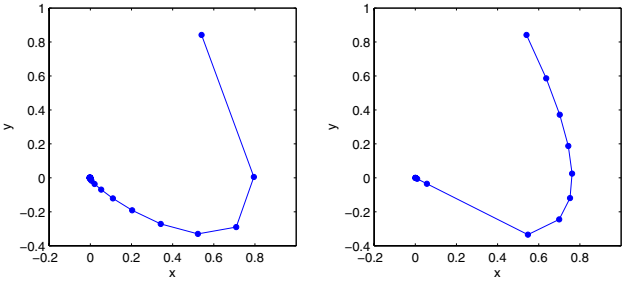
$$a_{k+1} - a_k = \Lambda(x_k, \Upsilon, \eta) = \sqrt{\eta \frac{x_k^T M_2 x_k}{x_k^T (A + BL)^T M_1 (A + BL) x_k}}. \tag{12}$$

The optimal problem (6)–(9) is completely defined except for (9). Note that for each optimization problem, 9 free parameters have been defined (6 for both positive semidefinite $M_{1,2}$, 1 for η, and 2 for L). Considering for example problem (6)–(9), the optimal solution achieves a slightly better cost than the optimal periodic controller, 26.6005, with

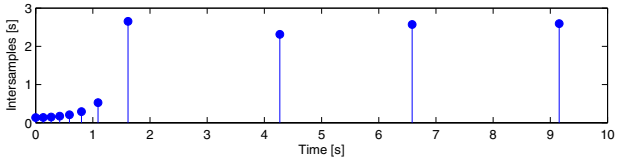
$$L = \begin{bmatrix} -0.847 & -1.723 \end{bmatrix}, M_1 = \begin{bmatrix} 0.028 & 0.091 \\ 0.091 & 0.336 \end{bmatrix}, M_2 = \begin{bmatrix} 0.054 & 0.017 \\ 0.017 & 0.069 \end{bmatrix}, \eta = 0.0.212,$$

but drastically reducing resource utilization.

Figure 1 a) shows the closed loops dynamics of the periodic optimal controller and event-driven controller respectively, where circles mark control updates. Both trajectories exhibit similar dynamics. Focusing on the dynamics given by the periodic controller, we can observe that from the first to the second control update, the state moves fast because it covers a long trajectory. And as control updates progress, the covered trajectories become shorter (the state moves slow). Looking at the dynamics given by the event-driven controller, we can observe the opposite behavior. When the state moves fast, we have more frequent control updates than when the state moves slow.



(a) Discrete and event-driven closed loop dynamics



(b) Event-driven controller activation times

Fig. 1. Numerical example

Figure 1 b) shows the activation pattern of control updates for the event-driven controller. The x-axis is simulation time, and each control update is represented by a vertical line, whose height indicates the time (in seconds) elapsed to the next control update. It shows that activation times occur within a range ([0.1 2.6]s, approximately) that in average is 1.84s, three times slower than the periodic controller!! Only the first 10s of simulation time are shown in this subfigure. By looking at the rest of simulation time, we would observe that sampling intervals oscillate within 2.51s and 2.58s.

4 Conclusions

This paper has formalized two optimal control design problems for event-driven controllers with limited resource utilization. The formalization includes a restriction on the amount of resources to be spent or on the cost to be achieved. Future work will look for closed solutions to the problem.

References

1. Tabuada, P.: Event-triggered real-time scheduling of stabilizing control tasks. IEEE Transactions on Automatic Control 52(9), 1680–1685 (2007)
2. Wang, X., Lemmon, M.: Self-triggered Feedback Control Systems with Finite-Gain L2 Stability. IEEE Transactions on Automatic Control (accepted) (July 2008)
3. Velasco, M., Martí, P., Bini, E.: Control-driven Tasks: Modeling and Analysis. In: 29th IEEE Real-Time Systems Symposium, Barcelona, Spain (2008)

Morphisms for Non-trivial Non-linear Invariant Generation for Algebraic Hybrid Systems

Nadir Matringe[2], Arnaldo Vieira Moura[3,⋆], and Rachid Rebiha[1,3,⋆⋆]

[1] Faculty of Informatics, University of Lugano, Switzerland.
rachid.rebiha@lu.unisi.ch
[2] Institue de Mathematiques de Jussieu Université Paris 7-Denis Diderot, France
matringe@math.jussieu.fr
[3] Institute of Computing, University of Campinas, SP.Brasil
arnaldo@ic.unicamp.br

Abstract. We present a new method that addresses various deficiencies of the state-of-the-art non-linear invariant generation methods for hybrid systems. By identifying suitable endomorphisms for each algebraic consecution condition, we reduce the problem to the computation of specific eigenspaces and their intersections.

1 Introduction

Consider a computational model of a hybrid system. An invariant at a location is an assertion true of any reachable system states associated to this location. In order to automate the generation of *non-trivial* multivariate polynomial invariants, one needs to handle *initiation* and *discrete consecution* conditions. We need to discover inductive algebraic assertions that hold at the initialization, and are induced by the structure of the discrete transitions at each state. Moreover, one needs to handle *continuous consecution* conditions: *differential consecution* and *local* conditions. This requires inductive algebraic assertions that hold at each state satisfying the local state conditions and obeying the local differential rules. Invariant generation for hybrid systems have seen tremendous progress [1, 2] in recent years. But they are often based on highly complex computations or are limited to linear or constant differential and discrete systems, or abstract the *local* and *initial* conditions. First, we extend our previous work on discrete systems [3] and we identify suitable morphisms for each consecution condition of non-linear systems. The problem is then reduced to linear algebra and can be solved using known techniques from algebraic geometry. All proofs and examples are completely described in our associated technical report [4].

2 Hybrid Systems and Inductive Assertions

Definition 1. *A* hybrid system *is given by* $\langle V, V_t, L, \mathcal{T}, \mathcal{C}, \mathcal{D}, l_0, \Theta\rangle$*, where* V *is a set of variables,* $V_t = \{X_1, .., X_n\}$ *where* $X_i(t)$ *is a function of* t*,* L *is a set of*

⋆ Supported in part by CNPq grant 472504/2007-0.
⋆⋆ Authors in alphabetic order, supported in part by CNPq grant 142170/2007-0.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 445–449, 2009.

locations and l_0 is the initial location. A state is an interpretation of the variables in $V \cup V_t$. A transition $\tau \in \mathcal{T}$ is given by a tuple $\langle l_{pre}, l_{post}, \rho_\tau \rangle$, where l_{pre} and l_{post} name the pre- and post- locations of τ. The transition relation ρ_τ is a first-order assertion over $V \cup V_t \cup V' \cup V_t'$, where V and V_t correspond to current-state variables and functions, while V' and V_t' correspond to the next-state variables and functions. Θ is the initial condition, given as a first-order assertion over $V \cup V_t$. Also, $\mathcal{C}$ associates each location $l \in L$ to a local condition $\mathcal{C}(l)$ *denoting an assertion over $V \cup V_t$. Finally, $\mathcal{D}$ associates each location $l \in L$ to a* differential rule $\mathcal{D}(l)$ *corresponding to an assertion over $V \cup \{dX_i/dt | X_i \in V_t\}$.*

The differential rules describe the local evolution of variables and functions in V_t during an interval. A run of a hybrid automaton is an infinite sequence $\langle l_0, \kappa_0 \rangle \xrightarrow{\mu_0} \cdots \xrightarrow{\mu_{i-1}} \langle l_i, \kappa_i \rangle \xrightarrow{\mu_i} \cdots$ of states $\langle l_i, \kappa_i \rangle \in L \times \mathbb{R}^{|V \cup V_t|}$ where l_0 is the *initial* location and $\kappa_0 \models \Theta$. Given two consecutive states $\langle l_i, \kappa_i \rangle$ and $\langle l_{i+1}, \kappa_{i+1} \rangle$, each condition μ_i describes a *discrete consecution* if there exists a transition $\langle q, p, \rho_i \rangle \in \mathcal{T}$ such that $q = l_i$, $p = l_{i+1}$ and $\langle \kappa_i, \kappa_{i+1} \rangle \models \rho_i$. Otherwise μ_i is a *continuous consecution* condition and there exists $q \in L$, $\varepsilon \in \mathbb{R}$ and a differentiable and continuous function $\phi : [0, \varepsilon] \to \mathbb{R}^{|V \cup V_t|}$ such that the following three conditions hold: (i) $l_i = l_{i+1} = q$, (ii) $\phi(0) = \kappa_i$, $\phi(\varepsilon) = \kappa_{i+1}$, (iii) During the time interval $[0, \varepsilon]$, ϕ satisfies the local condition $\mathcal{C}(q)$ (i.e. $\forall t \in [0, \varepsilon]$, $\phi(t) \models \mathcal{C}(q)$) according to the local differential rule $\mathcal{D}(q)$ (in other words $\forall t \in [0, \varepsilon]$, $\langle \phi(t), d\phi(t)/dt \rangle \models \mathcal{D}(q)$).

Definition 2. *Let W be a hybrid system. An assertion φ over $V \cup V_t$ is an* invariant *at $l \in L$ if $\kappa \models \varphi$ when $\langle l, \kappa \rangle$ is a reachable state of W.*

So, an invariant holds on all states that reach location l.

Definition 3. *Let D be an assertion domain. An assertion map for W is a map $\gamma : L \to D$. We say that γ is* inductive *if and only if the* Initiation *and* Consecution *conditions hold: (Initiation) $\Theta \models \gamma(l_0)$, (Discrete Consecution) for all $\tau \in \mathcal{T}$ s.t $\tau = \langle l_i, l_j, \rho_\tau \rangle$ we have $\gamma(l_i) \wedge \rho_\tau \models \gamma(l_j)'$, (Continuous Consecution) for all $l \in L$, and two consecutive reachable states $\langle l, \kappa_i \rangle$ and $\langle l, \kappa_{i+1} \rangle$ in a possible run of W such that κ_{i+1} is obtained from κ_i according to the local differential rule $\mathcal{D}(l)$, if $\kappa_i \models \gamma(l)$ then $\kappa_{i+1} \models \gamma(l)$.*

Note that if $\gamma(l) \equiv (Q(X_1, .., X_n) = 0)$ where Q is a multivariate polynomial in $K[X_1, .., X_n]$ then $\mathcal{C}(l) \wedge (Q(X_1, .., X_n) = 0) \models (d(Q(X_1, .., X_n)/dt = 0)$. Hence, if γ is an inductive assertion map then $\gamma(l)$ is an invariant at l for W.

3 New Continuous Consecution Conditions

Now we show how we can encode differential continuous consecution conditions. Consider W as the hybrid automaton just as above. Let $l \in L$ (which could eventually be in a circuit) and $\eta(l)$ be a polynomial with unknown coefficients (that is, a candidate invariant) of the form $\eta(l) = P(X_1, .., X_n)$. Hence we have $d\eta(l)/dt = \partial P(X_1, .., X_n)/\partial X_1 \; dX_1(t)/dt + \cdots + \partial P(X_1, .., X_n)/\partial X_n \; dX_n(t)/dt$.

Definition 4. *For $P(X_1,..,X_n) \in \mathbb{R}[X_1,..,X_n]$, we define the polynomial D_P of $\mathbb{R}[Y_1,..,Y_n,X_1,..,X_n]$: $D_P(Y_1,..,Y_n,X_1,..,X_n) = \partial P(X_1,..,X_n)/\partial X_1\ Y_1 + ... + \partial P(X_1,..,X_n)/\partial X_n\ Y_n$.*

Hence, $d\eta(l)/dt = D_P(\dot{X}_1,..,\dot{X}_n,X_1,..,X_n)$. From now on, let $\dot{F}$ denote dF/dt. Let $\langle l,\kappa_i\rangle$ and $\langle l,\kappa_{i+1}\rangle$ be two consecutive configurations in a run. Then we can express local state continuous consecutions as $\mathcal{C}(l) \wedge (\eta(l)=0) \models (\dot{\eta}(l)=0)$.

Definition 5. *Let W be a hybrid system and let η be an algebraic inductive map. We identify the following notions to encode* continuous consecution conditions*: (i) η satisfies a* Constant*-scale local consecution at l if and only if there exists a constant $\lambda \in K$ such that $\mathcal{C}(l) \models (d\eta(l)/dt - \lambda\eta(l_j) = 0)$, (ii) η satisfies a* Strong*-scale local consecution at l if and only if $\mathcal{C}(l) \models (d\eta(l)/dt = 0)$.*

4 Morphisms for *strong* Differential Invariant Generation

We first consider a non linear differential system without initial conditions of the form $[\ \dot{X}_1 = P_1(X_1,\ldots,X_n),\ldots,\dot{X}_n = P_n(X_1,\ldots,X_n)\ ]$.

Theorem 1. *A polynomial $Q \in K[X_1,..,X_n]$ is a* strong invariant *for the differential rules if and only if $D_Q(P_1(X_1,..,X_n),..,P_n(X_1,..,X_n),X_1,..,X_n) = 0$.*

If Q has degree r, and d is the maximal degree of the P_i's, then we must have that $D_Q(P_1..P_n,X_1..X_n)$ has degree at most $r+d-1$. We reduce the problem by considering the endomorphism D from $\mathbb{R}_r[X_1,..,X_n]$ to $\mathbb{R}_{r+d-1}[X_1,..,X_n]$ given by $P(X_1,..,X_n) \mapsto D_P(P_1(X_1,..,X_n),..,P_n(X_1,..X_n),X_1,..,X_n)$ and we denote by M_D its matrix in the canonical basis of $\mathbb{R}_r[X_1,..,X_n]$ and $\mathbb{R}_{r+d-1}[X_1,..,X_n]$.

Theorem 2. *A polynomial Q of $\mathbb{R}_r[X_1,..,X_n]$ is a strong differential invariant for the preceding differential system if and only if it lies in the kernel of M_D.*

Example 1. (M_D for 2 variables, a degree 2 differential rule, and degree 2 invariants) The polynomials P_1 and P_2 are of the form $P_1(x,y) = a_1x^2+a_2xy+a_3y^2+a_4x+a_5y+a_6$, and $P_2(x,y) = a_7x^2+a_8xy+a_9y^2+a_{10}x+a_{11}y+a_{12}$. Using the basis $(x^2,xy,y^2,x,y,1)$ of $\mathbb{R}_2[x,y]$ and the basis $(x^3,x^2y,xy^2,y^3,x^2,xy,y^2,x,y,1)$ of $\mathbb{R}_3[x,y]$, the matrix M_D becomes:

$$\begin{pmatrix}
2a_1 & a_7 & 0 & 0 & 0 & 0\\
2a_2 & a_1+a_8 & 2a_7 & 0 & 0 & 0\\
2a_3 & a_2+a_9 & 2a_8 & 0 & 0 & 0\\
0 & a_3 & 2a_9 & 0 & 0 & 0\\
2a_4 & a_{10} & 0 & a_1 & a_7 & 0\\
2a_5 & a_4+a_{11} & 2a_{10} & a_2 & a_8 & 0\\
0 & a_5 & 2a_{11} & a_3 & a_9 & 0\\
2a_6 & a_{12} & 0 & a_4 & a_{10} & 0\\
0 & a_6 & 2a_{12} & a_5 & a_{11} & 0\\
0 & 0 & 0 & a_6 & a_{12} & 0
\end{pmatrix}.$$

If we add initial conditions of the form $(x_1(0)=u_1,..,x_n(0)=u_n)$, we are looking for an invariant in $\mathbb{R}_r[x_1,..,x_n]$ that belongs to the hyperplane $P(u_1,..,u_n) = 0$, *i.e.*, we are looking for Q in $ker(M_D) \cap \{P/P(u_1,..,u_n)=0\}$. As the intersection of the hyperplane $\{P|P(u_1,..,u_n)=0\}$ with constant polynomials is always reduced to zero, and as the intersection of any hyperplane with a subspace of $\mathbb{R}_r[x_1,..,x_n]$ has dimension at least one, we deduce the following theorem.

Theorem 3. *There exists a strong invariant of degree r for the differential system with initial conditions (any initial conditions, actually), if and only if the kernel of M_D is of dimension at least two.*

Lemma 1. *Let $Q_1, \ldots, Q_n$ be n polynomials in $\mathbb{R}[x_1, \ldots, x_n]$. Then there exists a polynomial Q such that $\partial_1 Q = Q_1, \ldots, \partial_n Q = Q_n$ if and only if for any $i \neq j$, $1 \leq i, j \leq n$, one has $\partial_i Q_j = \partial_j Q_i$.*

Let $Syz(P_1, .., P_n)$ denote the Syzygy bases [5] of $(P_1, \ldots, P_n)$.

Theorem 4. *There exists a strong invariant for a differential system if and only if there exists $(Q_1, .., Q_n)$ in $Syz(P_1, .., P_n)$, such that for any i, j with $i \neq j$ and $1 \leq i, j, \leq n$, one has $\partial_i Q_j = \partial_j Q_i$.*

For example, when $n = 2$, we get the following class of systems for which one can always find a strong invariant: $[\ \dot{x_1} = P_1(x_1, x_2),\ \dot{x_2} = P_1(x_1, x_2)\ ]$ with $\partial_2 P_2 = -\partial_1 P_1$. Indeed, $(P_2 - P_1)$ always belongs to $Syz(P_1, P_2)$ (it is actually a basis when P_1 and P_2 are relatively prime). Consider the following differential rules: $[\ \dot{x} = xy,\ \dot{y} = -y^2/2\ ]$. Here, we indeed have $\partial_2 P_2 = -\partial_1 P_1 = -y$. The corresponding invariant is $Q(x, y) = xy^2/2 = 0$. Consider a generalization to dimension n of the *rotational motion of a rigid body* as an other example: $[\ \dot{x_1} = a_1 x_2 \ldots x_n,\ \ldots\ ,\ \dot{x_n} = a_n x_1 \ldots x_{n-1}\ ]$. We treat the case when the a_i's are non zero, other cases being easier. Indeed, the vector $(Q_1 = x_1/a_1, Q_2 = -x_2/(n-1)a_2, \ldots, Q_n = -x_n/(n-1)a_n)$ belongs to $Syz(P_1, .., P_n)$, where $P_i = a_i x_1 \ldots x_{i-1} x_{i+1} \ldots x_n$ belongs to the set of polynomials defining the differential rule. Now if $i \neq j$, one has $\partial_i Q_j = \partial_j Q_i = 0$, and applying Theorem 4 we deduce that the system admits a strong invariant. We just have to solve $\partial_1 Q = x_1/a_1; Q_2 = -x_2/(n-1)a_2; ...; Q_n = -x_n/(n-1)a_n$. A trivial solution is $Q(x_1, .., x_n) = x_1^2/2a_1 - x_2^2/2(n-1)a_2 \cdots - x_n^2/2(n-1)a_n$. Hence, the system admits $x_1^2/2a_1 - x_2^2/2(n-1)a_2 \cdots - x_n^2/2(n-1)a_n = 0$ as a strong invariant.

5 Morphisms for Constant-Scale Consecution

Definition 6. *Let $Q \in K[X_1, ..., X_n]$. Then Q is a* λ-invariant for constant-scale continuous consecution *for the differential rules if $\dot{Q}(X_1, .., X_n) = \lambda Q(X_1, .., X_n)$, that is, $D_Q(P_1(X_1, .., X_n), .., P_n(X_1, .., X_n), X_1, .., X_n) = \lambda Q(X_1, .., X_n)$.*

If Q has degree r, we reduce the problem by considering the endomorphism D of $\mathbb{R}_r[X_1, .., X_n]$ given by $P(X_1, .., X_n) \mapsto D_P(P_1, .., P_n, X_1, .., X_n)$. By the definition of invariant for constant-scale consecution, Q will be a λ-invariant for constant-scale consecution of degree at most r if and only if λ is an eigenvalue of D, and Q is an eigenvector for λ. By letting M_D be the matrix of D in the canonical basis of $\mathbb{R}_r[X_1, .., X_n]$ we can state the following theorem.

Theorem 5. *A polynomial Q of $\mathbb{R}_r[X_1, .., X_n]$ is a λ-invariant for differential scale consecution of the differential system if and only if there exists an eigenvalue λ of M_D such that Q belongs to the eigenspace of M_D corresponding to λ.*

Example 2. (General case for 2 variables and degree 2) Consider the differential rules $[\ \dot{x} = a_1x + b_1y + c_1\ \ \dot{y} = a_2x + b_2y + c_2\ ]$, the matrix M_D in the basis $(x^2, xy, y^2, x, y, 1)$ is: $\begin{pmatrix} 2a_1 & a_2 & 2b_2 & 0 & 0 & 0 \\ 2b_1 & a_1+b_2 & 2a_2 & 0 & 0 & 0 \\ 0 & b_1 & 0 & 0 & 0 & 0 \\ 2c_1 & c_2 & 0 & a_1 & 0 & 0 \\ 0 & c_1 & 2c_2 & b_1 & b_2 & 0 \\ 0 & 0 & 0 & c_1 & c_2 & 0 \end{pmatrix}$. Roots of such associated characteristic polynomial can be calculated by Cardan's method. Thus, one will always be able to find non-trivial λ invariants in this case.

Consider the following differential system with initial conditions (where strong invariant can not be generated): $[\ \dot{x} = x\ \wedge\ \dot{y} = ny\ \wedge\ (x(0), y(0)) = (\lambda, \mu)\ ]$ has associated endomorphism $L : Q(x, y) \mapsto \partial_x Q(x, y)x + n\partial_y Q(x, y)y$. Writing its matrix in the basis $(x^n, x^{n-1}y, .., xy^{n-1}, y^n, .., x, y, 1)$ gives: $\begin{pmatrix} n & \dots & 0 & 0 \\ 0 & M & 0 & 0 \\ 0 & \dots & n & 0 \\ 0 & \dots & 0 & 0 \end{pmatrix}$. The corresponding eigenspace has at least dimension 2, and it contains $Vect(x^n, y)$. Using the theorem on the existence on solutions for any initial conditions, we deduce that there exists an invariant of the form $ax^n + by$, and which must verify $a\lambda^n + b\mu = 0$. If λ and μ are non zero, which is the interesting case, one can take $a = \lambda^{-n}$ and $b = \mu^{-1}$, which gives the invariant $x^n/\lambda^n + y/\mu = 0$.

6 Conclusions

Our non-trivial non-linear invariant generation methods do not require (doubly) exponential computations from the use of Grobner bases, quantifier eliminations, cylindrical algebraic decompositions, direct resolution of non-linear systems, or any abstraction operators. Moreover, we succeeded in reducing the problems to linear algebra and we presented necessary and sufficient conditions for the existence of *non-trivial* non-linear invariants.

References

[1] Sankaranarayanan, S., Sipma, H., Manna, Z.: Constructing invariants for hybrid system. In: Alur, R., Pappas, G.J. (eds.) HSCC 2004. LNCS, vol. 2993, pp. 539–554. Springer, Heidelberg (2004)

[2] Tiwari, A.: Generating box invariants. In: Egerstedt, M., Mishra, B. (eds.) HSCC 2008. LNCS, vol. 4981, pp. 658–661. Springer, Heidelberg (2008)

[3] Rebiha, R., Matringe, N., Vieira Moura, A.: Endomorphisms for non-trivial non-linear loop invariant generation. In: Fitzgerald, J.S., Haxthausen, A.E., Yenigun, H. (eds.) ICTAC 2008. LNCS, vol. 5160, pp. 425–439. Springer, Heidelberg (2008)

[4] Matringe, N., Vieira-Moura, A., Rebiha, R.: Morphisms for non-trivial non-linear invariant generation for algebraic hybrid systems. Technical Report TR-IC-08-32, Institute of Computing, University of Campinas (November 2008)

[5] Kreuzed, A., Robbiano, L.: Computational commutative algebra. Springer, Heidelberg (2005)

An Analysis of the Fuller Phenomenon on Transfinite Hybrid Automata

Katsunori Nakamura[1,2] and Akira Fusaoka[2]

[1] Department of Computer Science, Ritsumeikan University
Nojihigashi, Kusatsu-city, SIGA, JAPAN 525-8577
[2] Department of English Communication, Heian Jogakuin University
Nanpeidai, Takatsuki-city, OSAKA, JAPAN 569-1092
fusaoka@cs.ritsumei.ac.jp

Abstract. In this paper, we introduce the hybrid automaton on the discrete time structure of a countable scattered linear-order set and present the reachability analysis for the Zeno and reversed-Zeno behaviors in the Fuller's phenomenon by using the extended automaton.

1 Introduction

A hybrid system is the system in which discrete and continuous dynamics interact each other. In the most general hybrid system, the set of time points at which the discrete change occurs consists of a countable scattered linear-order set, which contain no dense sub-ordering. By Hausdorff's theorem [4], a countable scattered linear order set is formed from the order-type $\boldsymbol{n}$, $\mathbb{N}$ and $-\mathbb{N}$, where $\boldsymbol{n}$ denotes the order-type of the finite set $\{0, 1, 2, \cdots, n-1\}$, $\mathbb{N}$ denotes the order-type of the set of natural number $1, 2, \cdots$, and $-\mathbb{N}$ means the order-type of the negative part of integer number $\{\cdots, -2, -1\}$. For example, the discrete time structure of Zeno is given by $\mathbb{N}$, and the reverse Zeno is corresponding to $-\mathbb{N}$. Therefore, Hausdorff's theorem means that the order-type of discrete time structure of hybrid systems is restricted in the order-type of finite, Zeno, reversed-Zeno or its superposition. In this paper, we extend the discrete structure of the transfinite hybrid automaton (hereafter THA) from the ordinal to the countable scattered linear-order set, and we gives the reachability analysis for the Zeno and reversed-Zeno behaviors in the Fuller's phenomenon which has the order structure of $\mathbb{N} + \boldsymbol{n} + -\mathbb{N}$.

2 Transfinite Hybrid Automaton

[Hyper-real number $^*\mathbb{R}$]. $^*\mathbb{R}$ is an enlargement of the field of real number $\mathbb{R}$ including infinitesimal numbers and infinite numbers. Let $\mathcal{F}$ is Frećhet filter such that $\mathcal{F} = \{A \subseteq \mathbb{N} | \mathbb{N} - A \text{ is finite } \}$. Let W denote a set of sequences of real numbers $(a_1, a_2, \cdots)$. The set of hyperreal numbers $^*\mathbb{R}$ is defined by introducing the following equivalence relation into W.
$(a_1, a_2, \cdots) \sim (b_1, b_2, \cdots) \Leftrightarrow \{k \mid a_k = b_k\} \in \mathcal{F}$

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 450–454, 2009.

Namely, ${}^*\mathbb{R} = W/\sim$. We denote the equivalence class of $(a_1, a_2, \cdots)$ by $[(a_1, a_2, \cdots)]$. Intuitively, $[(a_1, a_2, \cdots)]$ is formed from the sequences by ignoring the difference of the finite parts. The usual real number a is treated as $[(a, a, a, \cdots)]$, so that ${}^*\mathbb{R}$ contains $\mathbb{R}$ itself. An element of $\mathbb{R}$ in ${}^*\mathbb{R}$ is called a standard number. We define a relation $u \approx v$ if the distance from u to v is infinitesimal. It is known that for any finite number $a \in {}^*\mathbb{R}$, there is only one standard number $b \in \mathbb{R}$ such that $a \approx b$. b is called a shadow of a and denoted by $b = {}^\circ a$. We use an infinite number $\omega = [(1, 2, \cdots)]$, and an infinitesimal number ε ($\varepsilon = \frac{1}{\omega}$) in the following section.

[Transfinite Hybrid Automaton THA]. A **THA** is a combination of the Büchi's transfinite automaton on ordinals and the nonstandard analysis which has been proposed in [3]. We extend the original THA to the automata on the countable scattered linear-order set by adding the left-limit transition [2]. In this paper, we use the (nonstandard) differential equation to describe the continuous dynamics of the system instead of the infinite iteration of infinitesimal action in the original THA.

Definition 1. A THA $\mathcal{A}$ is a 4 tuple, $\mathcal{A} = (X, \dot{X}, Q, E)$ where
(1) $X = \{x_1, x_2, \cdots, x_m\}, \dot{X} = \{\dot{x}_1, \dot{x}_2, \cdots, \dot{x}_m\}$: Sets of continuous variables and its derivatives. $(x, \dot{x})$ is called 'situation'.
(2) $Q = \{q_1, q_2, \cdots, q_n\}$: The finite set of states.
(3) E: The set of transition rules. $E \subseteq Q \times Q \cup \mathcal{P}(Q) \times Q \cup Q \times \mathcal{P}(Q)$.
(4) $\dot{X} = f_q(x, \dot{x})$ which gives the continuous dynamics for each $q \in Q$.
(5) The discrete action $H_i(x, \dot{x}) \to x := g_i(x, \dot{x})$ which is a label for E. $H_i(x, \dot{x})$ is a switching manifold on which discrete changes happen.

Definition 2. Let
$Lim_L(\{q_0, q_1, \cdots, q_i\}) = \{q \in Q | \forall k < i \exists j [k < j < i \text{ and } q = q_j]\}$,
$Lim_R(\{q_{i+1}, q_{i+2}, \cdots\}) = \{q \in Q | \forall k > i \exists j [i < j < k \text{ and } q = q_j]\}$.
A scattered linear-order set $\mathcal{T}$=$(t_0, t_1, \cdots, t_k) :$, where k is nonstandard integer, has a transition of the automaton $\mathcal{A} = (X, \dot{X}, Q, E)$ if and only if there exists $\varphi : \mathcal{T} \to Q$ such that $\varphi(t_i) = q_i$ and each q_i satisfies one of the following conditions for the continuation.

(1) a next transition: There exists $(q_i, q_{i+1}) \in E$ such that $H_i(x(t_i), \dot{x}(t_i))$, and $x(t_{i+1}) = g_i(x(t_i), \dot{x}(t_i))$.
(2) a left-limit transition: There exist $\{q_{i1}, q_{i2}, \cdots, q_{ir}\} \subseteq Lim_L(\{q_0, q_1, \cdots, q_i\})$ such that $(\{q_{i1}, q_{i2}, \cdots, q_{ir}\}, q_{i+1}) \in E$, and $x(t_{i1}) \approx x(t_{i2}) \approx \cdots \approx x(t_{ir})$ then $x(t_{i+1}) \approx x(t_{i1})$.
(2) a right-limit transition: There exist $\{q_{i1}, q_{i2}, \cdots, q_{ir}\} \subseteq Lim_R(\{q_0, q_1, \cdots, q_i\})$ such that $(q_i, \{q_{i1}, q_{i2}, \cdots, q_{ir}\}) \in E$, and $x(t_{ik}) \approx x(t_i)$ for $1 \leq k \leq r$.

3 A Verification of Zeno Phenomenon

[Fuller's Phenomenon]. Fuller's problem is to minimize $\int_{t_0}^{t_f} x^2 dx$ under the condition of $\dot{x} = y, \dot{y} = u, u \in [-1, 1]$ with the initial condition of

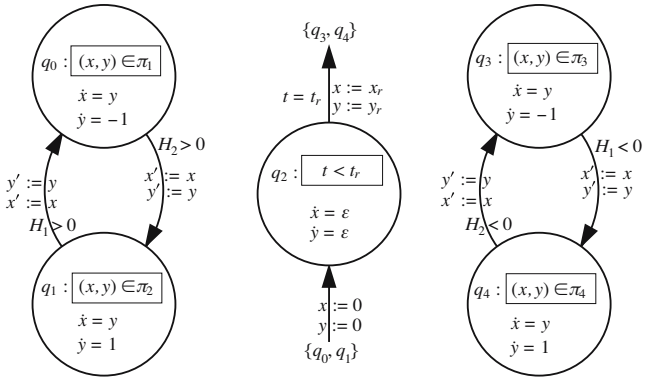


Fig. 1. THA of Fuller's dynamics

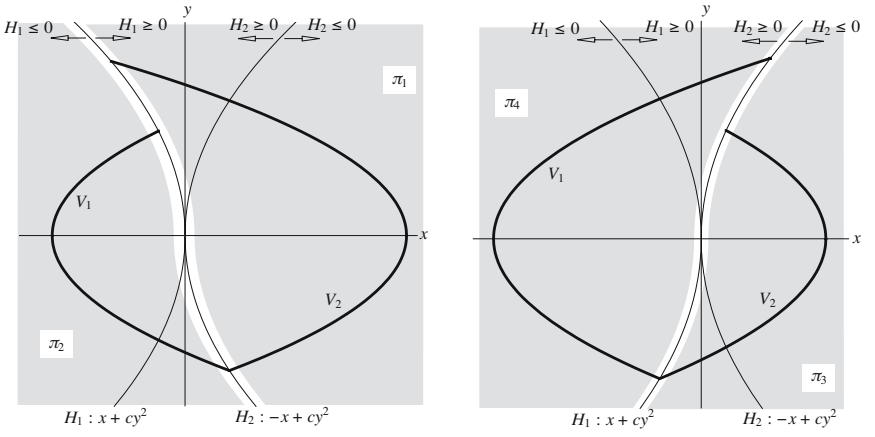
$x(t_0) = x_0, y(t_0) = y_0, x(t_f) = x_f, y(t_f) = y_f$. This is a classical problem of the optimal control theory. By using Pontryagin's maximal principle, Fuller had given the solution that contains infinite repetitions of discrete changes in 1961 [5]. Actually, Fuller's solution approaches to the point $(x, y) = (0, 0)$ through Zeno trajectory, then stay there for a finite period, and goes out through reversed-Zeno trajectory (Fig.2). Namely, the Zeno and reversed-Zeno give the optimal path to enter the stable state and to escape from the state, respectively. The THA for Fuller's phenomenon is given in Fig. 1.

[The existence of Zeno in Fuller's dynamics]. We analyze the part of trajectory toward the stable point $(x, y) = (0, 0)$ from the initial state of which discrete time structure is denoted by $\mathcal{T} = (t_0, t_1, \cdots, t_\omega)$ (Fig.2(a)). Let assume that $H_1(x, y) = x + cy^2, H_2(x, y) = -x + cy^2$ and the areas π_1, π_2 is defined as:
$\pi_1 = \{(x, y) | H_1(x, y) \geq 0 \wedge y \geq 0 \vee H_2(x, y) \leq 0 \wedge y < 0\}$
$\pi_2 = \{(x, y) | H_1(x, y) \leq 0 \wedge y \geq 0 \vee H_2(x, y) \geq 0 \wedge y < 0\}$.
The areas π_1, π_2 divides the space (x, y) into two parts. The part of $y > 0$ of $H_1(x, y) = 0$ and the part of $y < 0$ of $H_2(x, y) = 0$ are switching manifold (Fig.2).

Under the condition $0 < c < \frac{1}{4}$, the THA for Fuller dynamics satisfies:

$$\dot{x} = y \quad , \quad \dot{y} = u, (\text{inside of all states})$$
$$H_1(x, y) = 0 \wedge y > 0 \rightarrow u := -1, (\text{at the state transition from } q_1 \text{ to } q_0)$$
$$H_2(x, y) = 0 \wedge y < 0 \rightarrow u := 1 (\text{at the state transition from } q_0 \text{ to } q_1).$$

Without loss of generality, we can assume that the initial situation (x_0, y_0) satisfies $I \equiv x_0 < 0 \wedge 0 < y_0 \wedge H_1(x_0, y_0) = 0$.
We introduce the predicate Ψ such that
$\Psi(t_i) \equiv \exists a, b[\forall t_j \leq t_i \in \mathcal{T}(|x(t_j)| \leq a \wedge |y(t_j)| \leq b)]$ for standard $a, b > 0$.
Also we introduce the following two functions,
$V_1(x, y) = x + \frac{1}{2}y^2, V_2(x, y) = -x + \frac{1}{2}y^2$.



(a) Zeno (b) reversed-Zeno

Fig. 2. Fuller's dynamics (x-y phase portrait)

Because $\dot{V}_1(x, y) = \dot{x} + y\dot{y} = y - y = 0$ in π_1, we get
$V_1(x, y) = x + \frac{1}{2}y^2 = x_0 + \frac{1}{2}y_0^2 = (\frac{1}{2} - c)y_0^2$.
And because $x_0 \le x \le (\frac{1}{2} - c)y_0^2$, we get $a = (\frac{1}{2} - c)y_0^2$. Therefore, $|x| \le a$ at π_1. Also, from $\frac{1}{2}y^2 = (\frac{1}{2} - c)y_0^2 - x_0$, we have $|y| \le y_0$. Thus, we get $|y| \le b$ at π_1 by putting $b = y_0$. Considering that $\dot{y} = -1$ (y decreases monotonically) in this case, and x, y is bounded, we can see that the trajectory of the system behavior reaches the other manifold $H_2(x, y) = 0 \wedge y < 0$ within a finite time duration without being trapped into any equilibrium nor going out to infinity. Let assume that this point on the switching manifold is $(x, y) = (x_1, y_1)$, then we have $H_2(x_1, y_1) = 0 \wedge V_1(x_1, y_1) = 0$. By putting $\rho = \dfrac{1 - 2c}{1 + 2c}$, we get $x_1 = \rho c y_0^2, y_1 = -\sqrt{\rho} y_0$.
And the time to reach to the next switching manifold is
$t_1 = t_0 + \displaystyle\int_{y_0}^{y_1} -1 dy = y_0 - y_1 = (1 + \sqrt{\rho})y_0$.
Thus we have that $I \supset \varphi(t_1)$.

In the same way, we have $V_2(x, y) = -x + \frac{1}{2}y^2 = -x_1 + \frac{1}{2}y_1^2 = (\frac{1}{2} - c)y_1^2$ from $\dot{V}_2(x, y) = -\dot{x} + y\dot{y} = -y + y = 0$ in the area π_2. And from $(\frac{1}{2} - c)y_0^2 \le x \le x_0$, $\frac{1}{2}y^2 = (\frac{1}{2} - c)y_1^2 + x_1$, we get $|x| \le \rho a, |y| \le \rho b$. Thus, we have $|x| \le \rho a \wedge |y| \le \rho b$ at π_2.

Therefore, the trajectory of the dynamics reaches to the switching manifold $H_1(x, y) = 0 \wedge y > 0$ again. We put this point $(x, y) = (x_2, y_2)$, then we get $x_2 = -\rho x_1 = \rho^2 x_0, y_2 = -\sqrt{\rho} y_1 = \rho y_0$ because $H_1(x_2, y_2) = 0 \wedge V_2(x_2, y_2) = 0$. And the time to reach to the switching manifold is $t_2 - t_1 = y_2 - y_1 = \sqrt{\rho}(1 + \sqrt{\rho})y_0$. Therefore, $\Psi(t_1) \supset \Psi(t_2)$ becomes true because $\rho < 1$.

Similarly we can prove $\Psi(t_i) \supset \Psi(t_{i+1})$ for any $t_i \in \mathcal{T}$. Note that the mathematical induction holds even for the nonstandard natural number so that we get

$x(t_n) = (-\rho)^n x_0, y(t_n) = (-\sqrt{\rho})^n y_0, t_n = t_{n-1} + \sqrt{\rho}^n(1+\sqrt{\rho})$.
Since $\rho < 1$, we have $x(t_{i1}) \approx x(t_{i2}) \approx \cdots \approx x(t_{in}) \approx 0$ for the infinite integers $i1, i2, \cdots, in$. By the continuation condition, $x(t_\omega) \approx x(t_{i1}) \approx 0$. .Also,

$$t_z = t_1 + t_2 + \cdots = (1 + \sqrt{\rho} + \cdots)t_1 \approx \frac{1+\sqrt{\rho}}{1-\sqrt{\rho}} y_0$$

Thus we can conclude that the twisted chattering arc of Zeno approaches to the stable point $(0, 0)$ after the infinite switching within the finite time duration.

[Trajectory of the reverse Zeno]. We analyze the behavior the untwisted chattering of reverse Zeno on the time structure $\mathcal{T} = (t_{-\omega}, \cdots t_{-i}, t_{-i+1}, \cdots, t_f)$. However, the reverse Zeno contains the inherent uncertainty such that an infinitesimal difference in the initial condition causes observable effect after the infinite switching. Therefore, we can only argue about the weak reachability that there exists a trajectory such that $x(t_f) = x_F$ and there is no infinite integer ij such that $x(t_{-\omega}) \not\approx x_{-ij}$.
We define the areas π_3, π_4 as following:
$\pi_3 = \{(x, y) | H_1(x, y) \geq 0 \wedge y < 0 \vee H_2(x, y) \leq 0 \wedge y \geq 0\}$
$\pi_4 = \{(x, y) | H_1(x, y) \leq 0 \wedge y < 0 \vee H_2(x, y) \geq 0 \wedge y \geq 0\}$.
The part $y \leq 0$ of $H_1(x, y) = 0$ and the part $y > 0$ of $H_2(x, y) = 0$ are switching manifold this time.

Since the behavior of this part is completely symmetric to Zeno though time is reverse, we have $x_{n-k} = (-\rho)^k x_f, \quad y_{n-k} = (-\sqrt{\rho})^k y_f$ for any k.

Therefore, we can find the starting time and the position $(x, y) = (x_r, y_r)$ for the reverse Zeno.

$t_r = t_n + t_{n-1} + \cdots = (1 + \sqrt{\rho})(1 + \sqrt{\rho} + \cdots)y_b \approx \frac{1+\sqrt{\rho}}{1-\sqrt{\rho}} y_0$,
$x_r = \delta x_f, \quad y_r = \sqrt{\delta} y_f$ where $\delta = \rho^\omega \approx 0$

Finally, we can conclude that the system reaches to the Zeno point from the initial situation (x_0, y_0, t_0), then it remains at stable state for sometime, and there exists a trajectory which starts from the neighbor of $(0, 0)$ and arrives at the final state (x_f, y_f, t_f) after the untwisted chattering (reversed-Zeno).

References

1. Ames, A.D., Abate, A., Sastry, S.: Sufficient condition for the existence of Zeno behavior. In: 44th IEEE Conference on Decision and Control and Europian Control Conference (2005)
2. Bruyeŕe, V., Carton, O.: Automata on linear orderings. Journal of Computer and System Sciences 73(1), 1–24 (2007)
3. Nakamura, K., Fusaoka, A.: On Transfinite Hybrid Automata. In: Morari, M., Thiele, L. (eds.) HSCC 2005. LNCS, vol. 3414, pp. 495–510. Springer, Heidelberg (2005)
4. Rosenstein, J.G.: Linear Orderings. Academic Press, London (1982) (A Subsidiary of Harcourt Brace Jovanovich, Publishers)
5. Zelikin, M.I., Borisov, V.F.: Theory of Chattering Control with applications to Astronautics, Robotics, Economics, and Engineering. Birkhaüser, Boston (1994)

Stochastic Optimal Tracking with Preview for Linear Discrete-Time Markovian Jump Systems (Extended Abstract)

Gou Nakura

Osaka University, Department of Engineering
2-1, Yamadaoka, Suita, Osaka, 565-0871, Japan
nakura@watt.mech.eng.osaka-u.ac.jp

Abstract. In this paper we study the stochastic optimal tracking problems with preview for a class of linear discrete-time Markovian jump systems. Our systems are described by the discrete-time switching systems with Markovian mode transitions. The necessary and sufficient conditions for the solvability of our optimal tracking problem is given by coupled Riccati difference equations with terminal conditions. Correspondingly feedforward compensators introducing future information are given by coupled difference equations with terminal conditions. We consider both of the cases by state feedback and output feedback.

Keywords: Markovian jump systems; Stochastic optimization theory; Tracking control with preview; Coupled Riccati difference equations; Coupled feedforward compensators.

Notations: Throughout this paper the superscript "$'$" stands for the matrix transposition, $\|\cdot\|$ denotes the Euclidian vector norm and $\|v\|_R^2$ also denotes the weighted norm $v'Rv$.

1 Problem Formulation

Let $(\Omega, \mathcal{F}, \mathcal{P})$ be a probability space and, on this space, consider the following linear discrete-time time-varying system with reference signal and Markovian mode transitions.

$$\begin{aligned} x(k+1) &= A_{d,m(k)}(k)x(k) + G_{d,m(k)}(k)\omega_d(k) \\ &\quad + B_{2d,m(k)}(k)u_d(k) + B_{3d,m(k)}(k)r_d(k),\ x(0) = x_0,\ m(0) = i_0 \quad (1) \\ z_d(k) &= C_{1d,m(k)}(k)x(k) + D_{12d,m(k)}(k)u_d(k) + D_{13d,m(k)}(k)r_d(k) \\ y(k) &= C_{2d,m(k)}(k)x(k) + H_{d,m(k)}(k)w_d(k) \end{aligned}$$

where $x \in \mathbf{R^n}$ is the state, $\omega_d \in \mathbf{R^{p_d}}$ is the exogenous random noise, $u_d \in \mathbf{R^m}$ is the control input, $z_d \in \mathbf{R^{k_d}}$ is the controlled output, $r_d(\cdot) \in \mathbf{R^{r_d}}$ is known or measurable reference signal and $y \in \mathbf{R^k}$ is the measured output. x_0 is an unknown initial state with given distribution and i_0 is a given initial mode.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 455–459, 2009.

$\{m(k)\}$ is a homogeneous Markov process taking values on the finite set $\phi = \{1, 2, \cdots, N^*\}$ with the following transition probabilities:

$$\mathcal{P}\{m(k+1) = j | m(k) = i\} =: p_{d,ij}(k)$$

where $p_{d,ij}(k) \geq 0$ is also the transition rate at the jump instant from the state i to j, $i \neq j$, and $\sum_{j=1}^{N^*} p_{d,ij}(k) = 1$. We assume that all these matrices are of compatible dimensions. Throughout this paper the dependence of the matrices on k will be omitted for the sake of notation simplification.

For this system (1), we assume the following conditions.

A1: $D_{12d,m(k)}(k)$ is of full column rank.
A2: $D'_{12d,m(k)}(k)C_{1d,m(k)}(k) = O,\ D'_{12d,m(k)}(k)D_{13d,m(k)}(k) = O$
A3: $\mathbf{E}\{x(0)\} = \mu_0,\ \mathbf{E}\{\omega_d(k)\} = 0,$
$\mathbf{E}\{\omega_d(k)\omega'_d(k)\mathbf{1}_{\{m(k)=i\}}\} = \Xi_i(k),\ \mathbf{E}\{x(0)x'(0)\mathbf{1}_{\{m(0)=i_0\}}\} = Q_{i_0}(0)$
A4: $G_{d,m(k)}(k)H'_{m(k)}(k) = O,\ H_{m(k)}(k)H'_{m(k)}(k) = O$

where $\mathbf{1}_{\{m(k)=i\}} := 1$ if $m(k) = i$, and $\mathbf{1}_{\{m(k)=i\}} := 0$ if $m(k) \neq i$.

For the given initial mode i_0 and the given distribution of x_0, considering the stochastic mode transitions and the average of the performance indices over the statistics of the unknown part of r_d, we define the following performance index.

$$J_{dT}(x_0, u_d, r_d) := \sum_{k=0}^{N} \mathbf{E}_{\bar{R}_k}\{\|C_{1d,m(k)}x(k) + D_{13d,m(k)}r_d(k)\|^2\} + \sum_{k=0}^{N-1} \mathbf{E}_{\bar{R}_k}\{\|D_{12d,m(k)}u_d(k)\|^2\} \quad (2)$$

$\mathbf{E}_{\bar{R}_k}$ means the expectation over $\bar{R}_{k+h}$, h is the preview length of $r_d(k)$, and $\bar{R}_k$ denotes the future information on r_d at the time k, i.e.,$\bar{R}_k := \{r_d(l); k < l \leq N\}$.

Now we formulate the following optimal fixed-preview tracking problems for (1) and (2). In these problems, it is assumed that, at the current time k, $r_d(l)$ is known for $l \leq \min(N, k+h)$.

The Stochastic Optimal Tracking Problem
Consider the system (1) and the performance index (2), and assume the conditions **A1**, **A2** and **A3**. Then, find $\{u_d^*\}$ minimizing the performance index (2).

State feedback Case. The control strategy $u_d^*(k)$, $0 \leq k \leq N-1$, is based on the information $R_{k+h} := \{r_d(l); 0 \leq l \leq k+h\}$ with $0 \leq h \leq N$ and the state information $x(k)$ *at the current time* k.

Output Feedback Case. The control strategy $u_d^*(k)$, $0 \leq k \leq N-1$, is based on the information $R_{k+h} := \{r_d(l); 0 \leq l \leq k+h\}$ with $0 \leq h \leq N$ and the observed information $\mathcal{Y}_k := \{y(l); 0 \leq l \leq k\}$.

2 Design of Tracking Controllers by State Feedback

Now we consider the coupled Riccati difference equations ([1],[4])

$$X_i(k) = A'_{d,i}\mathcal{E}_i(X(k+1),k)A_{d,i} + C'_{1d,i}C_{1d,i} - F'_{2,i}T_{2,i}F_{2,i}(k),\ k = 0,1,\cdots \quad (3)$$

where $\mathcal{E}_i(X(k+1),k) = \sum_{j=1}^{N^*} p_{d,ij}(k)X_j(k+1)$, $X(k) = (X_1(k),\cdots,X_{N^*}(k))$

$$T_{2,i}(k) = D'_{12d,i}D_{12d,i} + B'_{2d,i}\mathcal{E}_i(X(k+1),k)B_{2d,i},$$
$$R_{2,i}(k) = B'_{2d,i}\mathcal{E}_i(X(k+1),k)A_{d,i},\ F_{2,i}(k) = -T_{2,i}^{-1}R_{2,i}(k)$$

and the following scalar coupled difference equations.

$$\alpha_i(k) = \mathcal{E}_i(\alpha(k+1),k) + tr\{G_{d,i}\Xi_i(k)G'_{d,i}\mathcal{E}_i(X(k+1),k)\} \quad (4)$$

Then we obtain the following necessary and sufficient conditions for the solvability of our stochastic optimal tracking problem and an optimal control strategy by state feedback for this problem.

Theorem 1. *Consider the system (1) and the performance index (2). Suppose* **A1**, **A2** *and* **A3**. *Then* **the Stochastic Optimal Tracking Problem by State Feedback** *for (1) and (2) is solvable if and only if there exist matrices* $X_i(k) > O$ *and scalar functions* $\alpha_i(k)$, $i = 1,\cdots,N^*$ *satisfying the conditions* $X_i(N) = C'_{1d,i}(N)C_{1d,i}(N)$ *and* $\alpha_i(N) = 0$ *such that the coupled Riccati equations (3) and and the coupled scalar equations (4) hold over* $[0,N]$. *Moreover an optimal control strategy for our tracking problem (1) and (2) is given by*

$$u_d^*(k) = F_{2,i}(k)x(k\tau) + \mathbf{D}_{u,i}(k)r_d(k) + \mathbf{D}_{\theta u,i}\mathcal{E}_i(\theta_c(k+1),k)@$$
$$for\ i = 1,\cdots,N^*$$

$\mathbf{D}_{u,i}(k) = -T_{2,i}^{-1}(k)B'_{2d,i}\mathcal{E}_i(X(k+1),k)B_{3d,i}$ *and* $\mathbf{D}_{\theta u,i}(k) = -T_{2,i}^{-1}(k)B'_{2d,i}$. $\theta_i(k)$, $i = 1,\cdots,N$, $k \in [0,N]$ *satisfies*

$$\theta_i(k) = \bar{A}'_{d,i}(k)\mathcal{E}_i(\theta(k+1),k) + \bar{B}_{d,i}(k)r_d(k),\ \theta_i(N) = C'_{1,i}D_{13,i}r_d(N) \quad (5)$$

where $\bar{A}_{d,i}(k) = A_{d,i} - \mathbf{D}'_{\theta u,i}T_{2,i}F_{2,i}(k)$,

$$\bar{B}_{d,i}(k) = A'_{d,i}\mathcal{E}_i(X(k+1),k)B_{3d,i} - F'_{2,i}T_{2,i}\mathbf{D}_{u,i}(k) + C'_{1d,i}D_{13d,i}$$

and $\theta_{c,i}(k)$ *is the 'causal' part of* $\theta_i(\cdot)$ *at time* k. *This* $\theta_{c,i}$ *is the expected value of* θ_i *over* $\bar{R}_k$ *and given by*

$$\begin{cases} \theta_{c,i}(l) = \bar{A}'_{d,i}(l)\mathcal{E}_i(\theta_c(l+1),l) + \bar{B}_{d,i}(l)r_d(l),\ k+1 \le l \le k+h, \\ \theta_{c,i}(k+h+1) = 0\ if\ k+h \le N-1 \\ \theta_{c,i}(k+h) = C'_{1,i}D_{13,i}r_d(N)\ if\ k+h = N \end{cases} \quad (6)$$

$\mathcal{E}_i(\theta_c(k+1),k) = \sum_{j=1}^{N^*} p_{d,ij}(k)\theta_{c,j}(k+1)$ *and* $\theta_c(k) = (\theta_{c,1}(k),\cdots,\theta_{c,N^*}(k))$. *Moreover, the optimal value of the performance index is*

$$J_{dT}(x_0^*,u_d^*,r_d) = tr\{Q_{i_0}X_{i_0}\} + \alpha_{i_0}(0) + \mathbf{E}_{\bar{R}_0}\{2\theta'_{i_0}x_0\}$$
$$+ \sum_{k=0}^{N-1}\mathbf{E}_{\bar{R}_k}\{\|T_{2,m(k)}^{1/2}\mathbf{D}_{\theta u,m(k)}(k)\mathcal{E}_{m(k)}(\theta_c^-(k+1),k)\|^2\} + \bar{J}_d(r_d) \quad (7)$$

where $\theta^-_{c,m(k)}(k) = \theta_{m(k)}(k) - \theta_{c,m(k)}(k),\ k \in [0, N]$,
$\mathcal{E}_i(\theta^-_c(k+1), k) = \sum_{j=1}^{N^*} p_{d,ij}(k)\theta^-_{c,j}(k+1)$, *and* $\theta^-_c(k) = (\theta^-_{c,1}(k), \cdots, \theta^-_{c,N^*}(k))$
$\bar{J}_d(r_d)$ *means the tracking error terms including the future information* θ_i *and not depending on* x_0 *and* u_d.

3 Output Feedback Case

For the plant dynamics (1), consider the controller

$$\begin{aligned}\hat{x}_e(k+1) &= A_{d,m(k)}(k)\hat{x}_e(k) + B_{2d,m(k)}(k)\bar{u}_{d,c}(k) + \bar{r}_{d,c}(k) \\ &\quad -M_{m(k)}(k)[y(k) - C_{2d,m(k)}\hat{x}_e(k)] \quad (8)\\ \bar{u}_{d,c}(k) &= F_{2,m(k)}(k)\hat{x}_e(k),\ \hat{x}_e(0) = \mathbf{E}_{\bar{R}_0}\{x_0\} = \mu_0\end{aligned}$$

where $M_{m(k)}$ is the controller gain to decide later, using the solution of another coupled Riccati equations introduced below, and

$$\begin{aligned}\bar{u}_{d,c}(k) &:= u_d(k) - \mathbf{D}_{u,i}(k)r_d(k) - \mathbf{D}_{\theta u,i}(k)\mathcal{E}_i(\theta_c(k+1), k)\\ \bar{r}_{d,c}(k) &:= B_{2d,m(k)}(k)\{\mathbf{D}_{u,m(k)}(k)r_d(k) + \mathbf{D}_{\theta u,m(k)}(k)\mathcal{E}_{m(k)}(\theta_c(k+1), k)\}\\ &\quad +B_{3d,m(k)}(k)r_d(k)\end{aligned}$$

Define the error variable $e(k) := x(k) - \hat{x}_e(k)$ and the error dynamics is as follows:

$$\begin{aligned}e(k+1) &= A_{d,m(k)}(k)e(k) + G_{d,m(k)}(k)w_d(k)\\ &\quad +M_{m(k)}(k)[y(k) - C_{2d,m(k)}(k)\hat{x}_e(k)]\\ &= [A_{d,m(k)} + M_{m(k)}C_{2d,m(k)}](k)e(k) + [G_{d,m(k)}(k) + M_{m(k)}H_{m(k)}](k)w_d(k)\end{aligned}$$

Note that this error dynamics does not depend on the exogenous inputs u_d nor r_d. Our objective is to design the controller gain $M_{m(k)}$ which minimizes

$$\begin{aligned}&J_{dT}(x_0, \bar{u}_{d,c*}, r_d) = tr\{Q_{i_0}X_{i_0}\} + \alpha_{i_0}(0) + \mathbf{E}_{\bar{R}_0}\{2\theta'_{i_0}x_0\}\\ &+\sum_{k=0}^{N-1}\mathbf{E}_{\bar{R}_k}\{\|F_{2,m(k)}e(k) + \mathbf{D}_{\theta u,m(k)}(k)\mathcal{E}_{m(k)}(\theta^-_c(k+1), k)\|^2\}_{T_{2,m(k)}(k)} + \bar{J}_d(r_d)\end{aligned}$$

Now we consider the following coupled Riccati difference equations and the initial conditions.

$$\begin{aligned}Y_j(k+1) &= \sum_{i\in\mathbf{J}(k)} p_{d,ij}\left[A_{d,i}Y_i(k)A'_{d,i}\right.\\ &\quad -A_{d,i}Y_i(k)C'_{2d,i}(H_{d,i}H'_{d,i}\pi_i(k) + C_{2d,i}Y_i(k)C'_{2d,i})^{-1}C_{2d,i}Y_i(k)A'_{d,i} \quad (9)\\ &\quad \left.+\pi_i(k)G_{d,i}G'_{d,i}\right],\ Y_i(0) = \pi_i(0)(\mathbf{Q}_0 - \mu_0\mu_0^*)\end{aligned}$$

where $\pi_i(k) := Prob\{\theta(k) = i\}$, $\sum_{i=1}^{N^*} p_{d,ij}\pi_i = \pi_j$, $\sum_{i=1}^{N^*}\pi_i = 1$, $\mathbf{J}(k) := \{i \in \mathbf{N}; \pi_i(k) > 0\}$

Since $\mathbf{E}\{e(k)\} = 0$ for $k \in [0, N]$ and $\bar{r}_{d,c}(k)$ is deterministic if $r_d(l)$ is known at all $l \in [0, k+h]$, we can show, for each $k \in [0, N]$,

$$\mathbf{E}_{\bar{R}_k}\{e(k)\bar{r}'_{d,c}(k)\mathbf{1}_{\{m(k)=i\}}\} = \pi_i(k)\mathbf{E}_{\bar{R}_k}\{e(k)\}\bar{r}'_{d,c}(k) = O.$$

Namely there exist no couplings between $e(\cdot)$ and $\bar{r}_{d,c}(\cdot)$. The development of $e(\cdot)$ on time k is independent of the development of $\bar{r}_{d,c}(\cdot)$ on time k. Then we can show the orthogonal property $\mathbf{E}_{\bar{R}_k}\{e(k)\hat{x}'_e(k)\mathbf{1}_{\{m(k)=i\}}\} = O$ as [Theorem 5.3 in [1] or Theorem 2 in [3]] by induction on k. Moreover define

$$\bar{Y}_i(k) = \mathbf{E}\{e(k)e'(k)\mathbf{1}_{\{m(k)=i\}}\}$$

and then we can show $Y_i(k) = \bar{Y}_i(k)$. From all these (orthogonality) results, as the case of $r_d(\cdot) \equiv 0$, using the solutions of the coupled difference Riccati equations, it can be shown that the gain $M_{m(k)}$ minimizing J_{dT} is decided as follows (cf.[1,3]):

$$M_i(k) = \begin{cases} -A_{d,i}Y_i(k)C'_{2d,i}(H_{d,i}H'_{d,i}\pi_i(k) \\ \qquad +C_{2d,i}Y_i(k)C'_{2d,i})^{-1} \text{ for } i \in \mathbf{J}(k) \\ 0 \text{ for } i \notin \mathbf{J}(k) \end{cases} \quad (10)$$

Finally the following theorem, which gives the solution of the output feedback problem, holds.

Theorem 2. *Consider the system (1) and the performance index (2). Suppose* **A1**, **A2**, **A3** *and* **A4**. *Then an optimal control strategy which, gives the solution of* **the Stochastic Optimal Tracking Problem by Output Feedback** *for (1) and (2) is given by the dynamic controller (8) with the gain (10) using the solutions of the two types of the coupled Riccati difference equations (3) with* $X_i(N) = C'_{1d,i}(N)C_{1d,i}(N)$ *and (9).*

References

1. Costa, O.L.V., Fragoso, M.D., Marques, R.P.: Discrete-Time Markov Jump Linear Systems. Springer, London (2005)
2. Cohen, A., Shaked, U.: Linear Discrete-Time H_∞-Optimal Tracking with Preview. IEEE Trans. Automat. Contr. 42(2), 270–276 (1997)
3. Costa, O.L.V., Tuesta, E.F.: Finite Horizon Quadratic Optimal Control and a Separation Principle for Markovian Jump Linear Systems. IEEE Trans. Automat. Contr. 48(10), 1836–1842 (2003)
4. Fragoso, M.D.: Discrete-Time Jump LQG Problem. Int. J. Systems Science 20(12), 2539–2545 (1989)
5. Nakura, G.: Stochastic Optimal Tracking with Preview for Linear Continuous-Time Markovian Jump Systems. In: Proc. SICE Annual Conference 2008, 2A09-2 (CD-ROM), Chofu, Tokyo, Japan (2008)
6. Nakura, G.: H_∞ Tracking with Preview for Linear Discrete-Time Markovian Jump Systems. In: Proc. 37th Symposium on Control Theory (in Japan), pp. 171–176 (2008)

Reachability Analysis for Stochastic Hybrid Systems Using Multilevel Splitting

Derek Riley[1], Xenofon Koutsoukos[1], and Kasandra Riley[2]

[1] ISIS/EECS,
Vanderbilt University, Nashville, TN 37235, USA
Derek.Riley,Xenofon.Koutsoukos@vanderbilt.edu
[2] Howard Hughes Medical Institute
Yale University, New Haven, CT, USA
Kasandra.Riley@yale.edu

1 Introduction

Biomedical research is increasingly using formal modeling and analysis methods to improve the understanding of complex systems. Verification methods for Stochastic Hybrid Systems (SHSs) are burdened with the curse of dimensionality; however, probabilistic analysis methods such as Monte Carlo (MC) methods can be used to analyze larger systems. MC methods are useful for estimating probabilities of event occurrences in SHS, but large and complex systems may require prohibitively large computation time to generate sufficient accuracy. In this work we present the multilevel splitting (MLS) variance reduction technique that has the potential to reduce the variance of MC methods by an order of magnitude significantly improving both their efficiency and accuracy [1].

This work presents an implementation of MLS methods for safety analysis of SHS. We apply the approach for safety analysis of the glycolysis process, which we model with a SHS model with two discrete states and 22 continuous variables. We also present experimental data along with accuracy and efficiency analysis. Further, the technique is parallelized to increase the efficiency, and we present the scalability of the parallelization.

2 SHS Model of Glycolysis

Glycolysis is a series of biochemical reactions that converts carbohydrates into chemicals and energy in a currency useful to cells. As it is a fundamental process to all living cells, it has been studied and modeled extensively in many organisms. Although the individual steps of glycolysis have been thoroughly examined, the interaction of glycolytic enzymes, substrates, and products with the intracellular environment is not fully understood. Modeling and simulating glycolysis using SHS can further our understanding of contextual cellular respiration.

Twenty-two chemical species and 37 chemical reactions have been identified which play an important role in glycolysis. The reaction rates for the system have been developed in previous work and can be found in [2]. The model presented

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 460–464, 2009.

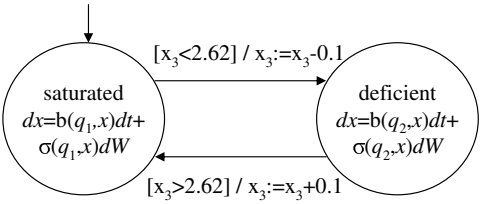


Fig. 1. SHS model of glycolysis

in [2] is a deterministic model, but the chemical reactions in the real system actually behave in a probabilistic manner due to the uncertainty of molecular motion, so we have developed a stochastic model in a similar manner as was done in [3].

We have added discrete dynamics to the original glycolysis model to capture the concept of 'feeding' the yeast. Glucose must be added to the system to continue production of the energy molecules, and when the concentration of glucose diminishes, the amount of energy molecules that the system can produce decreases. In many organisms, this reduction in energy output triggers mechanisms which encourage the introduction of more glucose (i.e. feeding). Therefore, we have modeled this behavior using a SHS with two states: saturated and deficient. In the saturated state, the glucose intake is fairly low, and in the deficient state, the glucose intake is much higher. Switching between the states is regulated by the concentration of ATP (x_3). A probabilistic reset map is used on the transition to avoid Zeno behavior. Figure 1 depicts the graphical version of the SHS model.

It is important for the cell performing glycolysis to maintain a certain concentration of Glucose x_1 to maintain cell health. Therefore, we want to determine if the state trajectories will avoid the set $U = \{(q, x) : x_1 < 2.5\}$.

3 Multilevel Splitting

We denote $s(t)$ the SHS trajectory, τ_{max} the maximum simulation time, and we define the stopping time $\tau_U = inf\{t > 0 : s(t) \in U\}$. We want to compute the probability that a trajectory will hit the unsafe set $P_{hit} = \mathbb{P}[\tau_U < \tau_{max}]$. P_{hit} can be estimated using MC methods; however, they are often computationally too expensive to generate estimators with small variance. MLS is an adaptation of MC methods that reduces the overall variance of the estimator by increasing the density of simulation trajectories near U [1].

MLS trajectories use importance values v_i to represent the amount of influence a trajectory has on P_{hit}. Initially $v_i = 1/n$ where n is the total number of trajectories. We define splitting levels using proper supersets of the unsafe set U: $U \subset U_1 \subset U_2 \subset ... \subset U_g$. When a trajectory crosses from a bigger set U_k into a smaller set U_{k-1}, the trajectory is split into j new trajectories, the importance value of the current trajectory is split between the new forked trajectories, and the total number of trajectories n_m is incremented by $j - 1$.

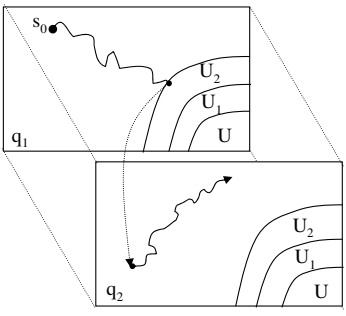


Fig. 2. MLS problem in a hybrid state space

The probability of hitting the unsafe set U is estimated for MLS methods by $\widehat{P_{hit}} = \sum_{i=1}^{n_m} H_i v_i$ where $H_i = 1$ if the trajectory eventually reaches U and $H_i = 0$ otherwise. Even though the resulting split trajectories are not completely independent, $\widehat{P_{hit}}$ is an unbiased estimator. The efficiency and accuracy of the estimator are dictated by the boundary placement, splitting policy, and dynamics of the model. The efficiency can be evaluated using $Eff\left[\widehat{P_{hit}}\right] = \frac{1}{Var \cdot C}$ where C is the expected execution time to compute the estimator, and Var is the variance of the estimator [1].

The discrete boundaries and reset maps present in SHS can create discontinuities which can cause accuracy and efficiency challenges for variance reduction methods. Figure 2 demonstrates a MLS scenario for a SHS where the trajectory crosses a splitting and hybrid boundary simultaneously. A hybrid trajectory starts at state $s_0 = (q_1, x_0)$, and evolves until it reaches the boundary for U_2 or the guards for a hybrid transition are satisfied. In this scenario, both the hybrid transition is fired and the splitting level is crossed, and the reset of the hybrid transition updates the state of the trajectory to $s = (q_2, x_t)$. Because the new state is not in the splitting region U_2, splitting the trajectory before applying the reset will not necessarily reduce the variance, and will decrease the efficiency, so it should be avoided. This problem is further exacerbated if the number of splits at a level is large because poor splitting choices decrease efficiency without increasing accuracy.

We have implemented the MLS algorithm for SHS using the simulation methods described in [4] to generate accurate and efficient SHS trajectories. Our MLS implementation ensures that discrete transitions are fired before testing splitting boundaries to avoid the potential efficiency loss of the boundary problem shown in Figure 2. The number of required simulations to achieve a sufficiently small variance may still be quite large even when using our algorithm, so we use parallel methods to improve overall efficiency. There are no dependencies between MLS simulations, so trajectories can be parallelized by running simulations concurrently on multiple processors. This type of parallelization has been used previously with MC methods [5], and care must be taken to ensure that the random number generators do not introduce bias.

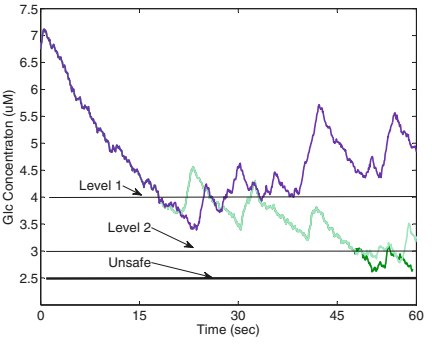


Fig. 3. Trajectory of the glycolysis model with MLS

Table 1. Performance and variance results

Simulation	Estimator	Var	$Time(C)$	Eff	P_{hit}
$sim1$	MC	623	398	0.0000040	0.12
$sim2$	MC	625	500	0.0000032	0.0625
$sim1$	MLS	711	128	0.0000110	0.0625
$sim2$	MLS	691	123	0.0000118	0.0625

Table 2. Parallel performance results

Processors	Time to Execute (m)
16	8.6
8	8.5
4	8.2
2	8.3
1	8.5

4 Experimental Results

Single trajectories of a model can be used to collect specific information about the system. In Figure 3, we show a single trajectory of the glycolysis model using MLS with two levels and two splits at each level.

To evaluate the efficiency of our methods for the glycolysis model, we used 16 trajectories for the MC methods and 2 MLS trajectories. This allows both methods to reach the same potential accuracy because the MLS scenario used three levels with two splits at each level yielding $2^4 = 16$ potential forked trajectories. In Table 1 we compare the variance and execution times using an order 0.5 simulation method ($sim1$) and an order 1.0 method with probabilistic boundary detection ($sim2$) from [4] with both MC methods and MLS methods. The data shows a significant efficiency improvement for the MLS estimator without a significant decrease in variance.

We also performed experiments to test the parallel scalability of our algorithm. We found that the MLS algorithm took virtually the same amount of

time regardless of the number of processors it was run on as seen in Table 2. The Advanced Computing Center for Research and Education (ACCRE) at Vanderbilt University provides the parallel computing resources for our experiments (www.accre.vanderbilt.edu).

5 Conclusions and Future Work

Analysis of SHS using Monte Carlo methods with variance reduction is an important technique which has the potential to expose insights into complex models efficiently. The SHS analysis method we present in this work demonstrates an efficient, accurate variance reduction method with parallelization, but it requires significant domain knowledge to determine appropriate splitting parameters. The method holds promise to provide further analysis capability for SHS simulation methods as well. In the future we will be investigating methods for selecting boundary placement and splitting policies based on methods presented for other splitting techniques. Our goal is to find an optimal policy for selecting the MLS parameters.

Acknowledgements. This research is partially supported by the National Science Foundation (NSF) CAREER grant CNS-0347440.

References

1. L'Ecuyer, P., Tuffin, B.: Splitting for rare-event simulation. In: Winter Simulation Conference, pp. 137–148 (2006)
2. Hynne, F., Dano, S., Sorensen, P.: Full-scale model of glycolysis in Saccharomyces cerevisiae. Biophysical Chemistry 94, 121–163 (2001)
3. Riley, D., Koutsoukos, X., Riley, K.: Modeling and simulation of biochemical processes using stochastic hybrid systems: The sugar cataract development process. In: Egerstedt, M., Mishra, B. (eds.) HSCC 2008. LNCS, vol. 4981, pp. 429–442. Springer, Heidelberg (2008)
4. Riley, D., Koutsoukos, X., Riley, K.: Simulation of stochastic hybrid systems with switching and reflective boundaries. In: Winter Simulation Conference, pp. 804–812 (2008)
5. Troyer, M., Ammon, B., Heeb, E.: Parallel object oriented monte carlo simulations. In: Caromel, D., Oldehoeft, R.R., Tholburn, M. (eds.) ISCOPE 1998. LNCS, vol. 1505, pp. 191–198. Springer, Heidelberg (1998)

Orbital Control for a Class of Planar Impulsive Hybrid Systems with Controllable Resets

Axel Schild[1], Magnus Egerstedt[2], and Jan Lunze[1]

[1] Institute of Automation and Computer Control, Ruhr-Universitaet Bochum, Universitaetsstrasse 150, 44780 Bochum, Germany
{Schild,Lunze}@atp.rub.de
[2] School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, 30332 Georgia, USA
magnus@ece.gatech.edu

Abstract. This paper addresses the orbital stabilization of controlled polygonal billiard systems. Such systems form a subclass of so-called planar impulsive hybrid systems that feature controllable guards and state resets. While the structure of the guards and the reset map is completely determined by the system set-up, both mappings are jointly adjustable through exogenous control inputs. As a central feature, control actions cause *simultaneous*, inseparable changes in the *reset time* and in the *reset action*. The paper proposes a hybrid control approach for the stabilization of an admissible stationary orbit.

1 Introduction

This paper proposes a model-based hybrid control strategy for the orbital stabilization of controlled *polygonal billiard systems* [1]. Such systems consist of a ball, which moves along straight line sequences on a closed polygonal table (see Fig. 1(a)). Each collision between the ball and one of the E walls $\mathcal{W}_\sigma$ is elastic and results in a reflection of the ball. By rotating the pivot-mounted walls ν degrees away from their nominal orientation $\boldsymbol{n}_\sigma^\star$, the reflection angle γ can be adjusted at run-time to control the ball's evolution as desired. Commanded wall rotations are assumed to occur instantaneously. Uncontrolled polygonal billiards are known to behave chaotically.

From a system theoretic perspective, controlled billiard systems belong to a class of periodically operated, planar impulsive hybrid systems (IHS) [2,3] with externally manipulable *autonomous state resets*. In the realm of mechanical systems with impacts, they are referred to as *juggling systems* [4,5] and have received considerable attention due to their relevance for robotics and locomotion. In particular, the task of stabilizing periodic stationary operations by controlled impacts has been extensively studied. Due to the inherent complexity of the problem, published results are limited to either simple 1-D dynamics, small control actions and/or low order periodic orbits [6,7].

This paper addresses the model-based design of an *event-surface controller* for controlled *planar* polygonal billiard systems. The controller processes sampled state measurements into an input sequence that *simultaneously* alters parameters of the reset map as well as the instants $\bar{t}(k)$, at which such resets occur. Due to the inseparable coupling of the event-time and the reset effect, every control action compromises between the

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 465–469, 2009.

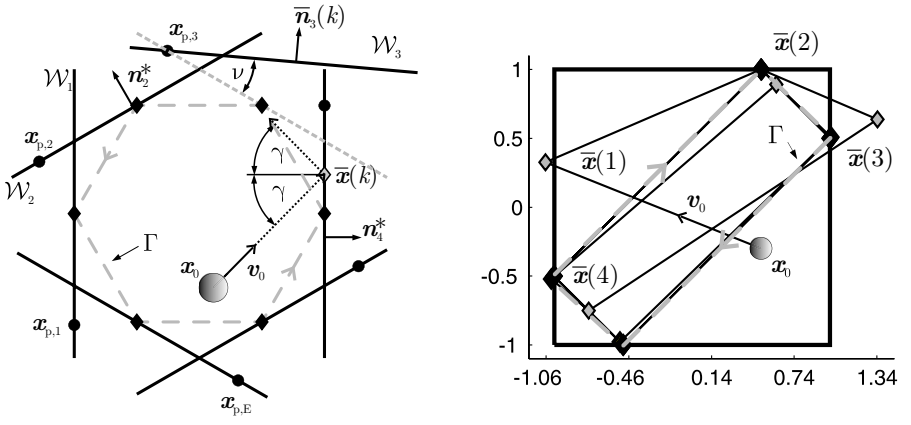


Fig. 1. Controlled polygonal billiard system: (a) Physical set-up, (b) stabilized ball motion on a rectangular table

"best" reset action and the "best" event time. A more detailed description of the results summarized here can be found in [8].

2 Continuous-Time Hybrid Model of Polygonal Billiards

The controlled billiard system evolves according to a hybrid model similar to [7]:

Impulsive hybrid system with controlled resets (IHSCR):

Autonomous continuous dynamics:

$$\dot{\boldsymbol{\zeta}}(t) = \left(\begin{array}{c|c} \boldsymbol{0} & \boldsymbol{0} \\ \hline \boldsymbol{I} & \boldsymbol{0} \end{array}\right) \boldsymbol{\zeta}(t)\,, \quad \boldsymbol{\zeta}(0) = \left(v_{x,0}\ v_{y,0} \,\middle|\, x_{x,0}\ x_{y,0}\right)^{\mathrm{T}} \tag{1}$$

Controlled event generator: (control input $\boldsymbol{u}(t) = \left(\bar{\boldsymbol{n}}_1(t)\ \ldots\ \bar{\boldsymbol{n}}_E(t)\right)^{\mathrm{T}}$)

$$\Phi(\boldsymbol{\zeta}(t)\,,\,\boldsymbol{u}(t)\,,\,\sigma) = \begin{cases} \bar{\boldsymbol{n}}_1^{\mathrm{T}}(t)\,(\boldsymbol{x}_{\mathrm{p},1} - \boldsymbol{x}(t))\,, \text{if } \sigma = 1 \\ \vdots \\ \bar{\boldsymbol{n}}_E^{\mathrm{T}}(t)\,(\boldsymbol{x}_{\mathrm{p},E} - \boldsymbol{x}(t))\,, \text{if } \sigma = E \end{cases} \tag{2}$$

$$e(t) = \arg\min_{\sigma \in \Sigma} |\Phi(\boldsymbol{\zeta}(t)\,,\,\boldsymbol{u}(t)\,,\,\sigma)| \tag{3}$$

$$\bar{t}(k) = \arg\min_{t > \bar{t}(k-1)} t \ :\ |\Phi(\boldsymbol{\zeta}(t)\,,\,\boldsymbol{u}(t)\,,\,e(t))| = 0 \tag{4}$$

Controlled reset map (assuming completely elastic impacts):

$$\boldsymbol{\zeta}\left(\bar{t}(k)^+\right) = \left(\begin{array}{c|c} \boldsymbol{I} - 2\frac{\bar{\boldsymbol{n}}_{\bar{e}(k)}\bar{\boldsymbol{n}}_{\bar{e}(k)}^{\mathrm{T}}}{\bar{\boldsymbol{n}}_{\bar{e}(k)}^{\mathrm{T}}\bar{\boldsymbol{n}}_{\bar{e}(k)}} & \boldsymbol{0} \\ \hline \boldsymbol{0} & \boldsymbol{I} \end{array}\right) \boldsymbol{\zeta}\left(\bar{t}(k)^-\right), \quad \bar{\boldsymbol{n}}_{\bar{e}(k)} = \boldsymbol{n}_{e\left(\bar{t}(k)^-\right)}\left(\bar{t}(k)^-\right) \tag{5}$$

Here, $\boldsymbol{\zeta}(t) = (\boldsymbol{v}(t), \boldsymbol{x}(t))^{\mathrm{T}} \in \mathbb{R}^4$ is the ball's state (heading and position), which evolves according to (1). An elastic collision between the ball and the σ-th wall occurs, whenever the controllable piecewise-affine event function (2) evaluates to zero for a tupel $(\boldsymbol{\zeta}(\bar{t}(k)^-), \boldsymbol{u}(\bar{t}(k)^-), \sigma)$. The states $\boldsymbol{\zeta}$, for which this condition is satisfied, form the *controlled event surfaces* $\mathcal{S}(\sigma, \boldsymbol{u}) = \{\boldsymbol{\zeta} : \Phi(\boldsymbol{\zeta}, \boldsymbol{u}, \sigma) = 0\}$ in state-space. At each intersection of $\boldsymbol{\zeta}(t)$ with a surface $\mathcal{S}(\sigma, \boldsymbol{u})$, the ball's state is discontinuously updated according to the *controlled reset map* (5), i.e. it is reflected at $\mathcal{S}(\sigma, \boldsymbol{u})$. The exogenous input $\boldsymbol{u}(t) \in \mathbb{R}^{2E}$, which defines the current wall orientations $\boldsymbol{n}_{\bar{e}(k)}$, allows to control the event time and the piecewise constant event sequence $e(t)$, despite the autonomous motion of the ball. Event-sampled signal values are denoted by $\bar{\boldsymbol{\zeta}}(k), \bar{e}(k)$, etc.

A trajectory $\boldsymbol{\zeta}^\star(t)$ is called *periodic* of *order* p, iff $\bar{\boldsymbol{\zeta}}^\star(k+p) = \bar{\boldsymbol{\zeta}}^\star(k)$ for all k. It traces out a closed orbit Γ, which is called the *limit cycle* corresponding to the stationary input $\boldsymbol{u}(t) = \boldsymbol{u}^\star$. Let $\bar{\boldsymbol{\zeta}}^\star(\sigma_k^\star) \in \Gamma$ denote the *impact points* of Γ associated with the event $\sigma_k^\star$ of the periodic event sequence $\bar{e}^\star(k) = \left(\sigma_0^\star \ldots \sigma_{p-1}^\star \sigma_0^\star \ldots\right)$. Moreover, define $\bar{\tau}^\star(\sigma_k^\star) = \bar{t}^\star(k+1) - \bar{t}^\star(k)$ and $\Gamma_k = \left\{\boldsymbol{y} : \exists \tau \in \mathbb{R} \text{ s.t. } \boldsymbol{y} = \boldsymbol{\zeta}\left(\tau, \bar{\boldsymbol{\zeta}}^\star(\sigma_k^\star)\right)\right\}$. $\boldsymbol{\zeta}_m^\star = 1/m \sum_{k=0}^{p-1} \bar{\boldsymbol{\zeta}}^\star(\sigma_k^\star)$ is the centroid and $\mathcal{I}(\Gamma) = \left\{\boldsymbol{\zeta} : \boldsymbol{n}_{\Gamma_k}^{\mathrm{T}}\left(\boldsymbol{\zeta} - \bar{\boldsymbol{\zeta}}^\star(\sigma_k^\star)\right) \leq 0, \forall k = 0 \ldots (p-1)\right\}$ is the interior of Γ. The normal $\boldsymbol{n}_{\Gamma_k}$ of Γ_k satisfies $\boldsymbol{n}_{\Gamma_k}^{\mathrm{T}}\left(\bar{\boldsymbol{\zeta}}^\star(\sigma_{k+1}^\star) - \bar{\boldsymbol{\zeta}}^\star(\sigma_k^\star)\right) = 0$ and $\boldsymbol{n}_{\Gamma_k}^{\mathrm{T}}\left(\boldsymbol{\zeta}_m^\star - \bar{\boldsymbol{\zeta}}^\star(\sigma_k^\star)\right) < 0$. Similarly, the interior of the *nominal table set-up* $\mathcal{W}^\star = \{\boldsymbol{n}_{\sigma_0^\star}^\star, \ldots \boldsymbol{n}_{\sigma_E^\star}^\star\}$ is $\mathcal{I}(\mathcal{W}^\star) = \{\boldsymbol{\zeta} : \Phi(\boldsymbol{\zeta}, \boldsymbol{u}^\star, \sigma) \geq 0, \forall \sigma = 1 \ldots E\}$.

3 Problem Formulation for the Billiards Problem

Problem 1. Given the hybrid model (1)-(5) of the billiards problem and an admissible desired stationary orbit Γ, the objective is to *determine* stabilizing pivot locations $\boldsymbol{x}_{\mathrm{p},\sigma}$, $\sigma = 1 \ldots E$ and an event surface control law $\boldsymbol{u}(t) = \boldsymbol{f}_c(\boldsymbol{\zeta}(t), e(t))$ in state-feedback form, which ensures global asymptotic orbital stability of Γ.

Definition 1 (Global asymptotic orbital stability). *[9] A periodic trajectory $\boldsymbol{\zeta}^\star(t)$ is called asymptotically orbitally stable, iff for any $\epsilon > 0$ there exists a $\delta > 0$, such that any $\boldsymbol{\zeta}(t)$ starting at $\mathrm{dist}(\boldsymbol{\zeta}(0), \Gamma) < \delta$ asymptotically converges towards $\boldsymbol{\zeta}^\star(t)$, which implies that $\mathrm{dist}(\boldsymbol{\zeta}(t), \Gamma) < \epsilon$, $\forall t > 0$ and $\lim_{t\to\infty} \mathrm{dist}(\boldsymbol{\zeta}(t), \Gamma) = 0$.*

A trajectory $\boldsymbol{\zeta}^\star(t)$ is called globally asymptotically orbitally stable, iff convergence is given for arbitrary initial conditions $\boldsymbol{\zeta}(0) \in \mathcal{I}(\mathcal{W}^\star)$.

4 Controlled Embedded Map for the Controlled Billiards System

A central step in the solution to problem 1 is to derive a sampled abstraction of the hybrid model (1)-(5) at the unknown time instants $\bar{t}(k)$. This *controlled embedded map* [10] describes the state evolution from one impact $\bar{\boldsymbol{\zeta}}(k)$ to the next $\bar{\boldsymbol{\zeta}}(k+1)$. It establishes an explicit relation between the design parameters and the control output and provides the foundation for the model-based design. Given a desired orbit Γ and assuming its event sequence $\bar{e}^\star(k)$ can be enforced by dedicated controls $\bar{\boldsymbol{u}}(k)$, the controlled embedded error map is:

Controlled embedded error map of the billiard system with respect to Γ:

$$\Delta\bar{\boldsymbol{v}}(k+1) = \Delta\bar{\boldsymbol{v}}(k) - \frac{2\bar{\boldsymbol{n}}_{\sigma_k^\star}}{\bar{\boldsymbol{n}}_{\sigma_k^\star}^{\mathrm{T}}\bar{\boldsymbol{n}}_{\sigma_k^\star}} + \frac{2\boldsymbol{n}_{\sigma_k^\star}^\star}{\boldsymbol{n}_{\sigma_k^\star}^{\star\mathrm{T}}\boldsymbol{n}_{\sigma_k^\star}^\star}, \quad \Delta\bar{\boldsymbol{\zeta}}(0) = \boldsymbol{\zeta}_0 - \bar{\boldsymbol{\zeta}}^\star(\sigma_0^\star) \tag{6}$$

$$\Delta\bar{\boldsymbol{x}}(k+1) = \Delta\bar{\boldsymbol{x}}(k) + \bar{\tau}^\star(\sigma_k^\star)\,\Delta\bar{\boldsymbol{v}}(k) - \bar{\boldsymbol{v}}(k)\left\{\Delta\boldsymbol{x}_p^{\mathrm{T}}(k)\,\bar{\boldsymbol{n}}_{\sigma_k^\star} + \bar{\tau}^\star(\sigma_k^\star)\right\} \tag{7}$$

Here, the errors are defined as $\Delta\bar{\boldsymbol{x}}(k) = \bar{\boldsymbol{x}}(k) - \bar{\boldsymbol{x}}^\star(\sigma_k^\star)$, $\Delta\boldsymbol{x}_p(k) = \bar{\boldsymbol{x}}(k) - \boldsymbol{x}_{\mathrm{p},\sigma_{k+1}^\star}$ and $\Delta\bar{\boldsymbol{v}}(k) = \bar{\boldsymbol{v}}(k) - \bar{\boldsymbol{v}}^\star(\sigma_k^\star)$. Moreover, the k-th input must satisfy the length constraint $\bar{\boldsymbol{v}}^{\mathrm{T}}(k)\,\bar{\boldsymbol{n}}_{\bar{c}(k)} = 1$. Equation (6), (7) reveal: 1. Only by controlling the wall orientations at operation, the heading error can be compensated, 2. only by deviating from the nominal heading, the position error can be reduced, and, 3. a vanishing position error $\Delta\bar{\boldsymbol{x}}(k) = 0,\ \forall k > K$ implies $\Delta\bar{\boldsymbol{v}}(k) = 0,\ \forall k > K$.

5 Stabilizing Hybrid Control Strategy and Simulation Results

The key ingredient for achieving orbital stabilization of Γ is to enforce a monotonous decay of the heading error $\Delta\bar{\boldsymbol{v}}(k)$ at each impact.

Theorem 1. *For a given nominal table set-up $\mathcal{W}^\star$, there exist pivot locations $\boldsymbol{x}_{\mathrm{p},\sigma}$ for each wall $\mathcal{W}_\sigma$, which are independent of the initial condition $\boldsymbol{\zeta}_0 \in \mathcal{I}(\mathcal{W}^\star)$ and enable a monotonous reduction of the heading error $\Delta\boldsymbol{v}(k)$ with increasing k.*

Being allowed to freely place the pivots $\boldsymbol{x}_{\mathrm{p},\sigma}$ along the nominally oriented walls, the following can be shown [8]:

Theorem 2. *Under the assumption of appropriate pivot locations $\boldsymbol{x}_{\mathrm{p},\sigma}$, it is possible to asymptotically drive any ball trajectory starting inside $\mathcal{I}(\mathcal{W}^\star)$ onto the orbit Γ by composing an appropriate input sequence from two basic control maneuvers*

- *(I) enforce next impact $\bar{\boldsymbol{x}}(k+1) \in \Gamma$ on orbit*
- *(II) command next impact $\bar{\boldsymbol{x}}(k+1) \notin \Gamma$, such that the ball is tranferred to a point $\bar{\boldsymbol{x}}(k+2) = \left(\bar{\boldsymbol{x}}^\star\left(\sigma_{k+2}^\star\right) + \bar{\boldsymbol{x}}^\star\left(\sigma_{k+3}^\star\right)\right)/2$.*

For both control maneuvers (I) and (II) closed-form state-feedback expression were obtained. The hybrid control strategy, which achieves global asymptotic stability of a given Γ, is summarized in Algorithm 1. For details, please refer to [8].

A successful application of the control strategy to a billiard system with a nominal rectangular table set-up is illustrated in Figure 1(b)). The depicted simulated execution starts at an arbitrarily chosen initial condition and is driven to the desired unstable orbit Γ within eight impacts. At the beginning, an initial transition onto Γ is executed. The third impact $\bar{\boldsymbol{x}}(3)$ is commanded to occur not on the orbit, as $\bar{\boldsymbol{x}}(4)$ would otherwise be once again located outside of the orbit. Instead, another transfer to a point on the inside of Γ is performed. All subsequent impacts then alternate between the inside and the outside of the orbit, which is crucial for the stabilization of Γ.

Acknowledgements. The work by A. Schild and J. Lunze is supported by the German Research Foundation (LU462/21-3) and the German Academic Exchange Service

(D/08/45420). The work by M. Egerstedt was funded by the US national science foundation (0509064).

Algorithm 1. *Globally stabilizing hybrid control strategy for controlled billiards.*

Given: *Billiard system (1)-(5), admissible orbit Γ and nominal wall set-up $\mathcal{W}^\star$*

Initialization:

1. *Determine stabilizing pivot locations $\boldsymbol{x}_{\mathrm{p},\sigma}$ for each wall.*
2. *Run the system uncontrolled with nominal wall orientations $\mathcal{W}^\star$, until $\exists\, \bar{\boldsymbol{n}}_{\sigma^\star_{k+1}}$, such that $\bar{\boldsymbol{x}}(k+2) = \bar{\boldsymbol{x}}^\star(\sigma^\star_2)$.*
3. *Given $\bar{\boldsymbol{\zeta}}(k)$, adjust $\mathcal{W}_{\sigma^\star_{k+1}}$, such that a transfer from $\bar{\boldsymbol{x}}(k)$ to $\bar{\boldsymbol{x}}^\star(\sigma^\star_2)$ is achieved.*

During operation *($k \geq 2$)**:***

1. *If impact $\bar{\boldsymbol{x}}(k)$ occurs on Γ_k, but not in $\mathcal{I}(\Gamma)$, execute maneuver (I).*
2. *Else, compute the actuations $\bar{\boldsymbol{n}}^{(1)}_{\sigma^\star_{k+1}}$ and $\bar{\boldsymbol{n}}^{(2)}_{\sigma^\star_{k+1}}$ for both control maneuvers (I) and (II) and apply the input $\bar{\boldsymbol{n}}_{\sigma^\star_{k+1}} = \arg\min_i (\Delta \boldsymbol{x}_p^{\mathrm{T}}(k)\, \bar{\boldsymbol{n}}^{(i)}_{\sigma^\star_{k+1}}(k))$.*

Result: *Global orbital stability of Γ for arbitrary $(\boldsymbol{x}(0), \boldsymbol{v}(0))$.*

References

1. Gutkin, E.: Billiards in polygons: Survey of recent results. Jour. of Stat. Phys. 83, 7–26 (1996)
2. Guan, Z., Hill, D., Shen, X.: On hybrid impulsive and switching systems and application to nonlinear control. IEEE Trans. Autom. Contr. 50, 1058–1062 (2005)
3. Lakshmikantham, V., Liu, X.: Impulsive hybrid systems and stability theory. Dynam. System Appl. 7, 1–9 (1998)
4. Buehler, M., Koditschek, D., Kindlmann, P.: Planning and control of robotic juggling and catching tasks. The International Journal of Robotics Research 13, 101–118 (1994)
5. Brogliato, B., Mabrouk, M., Rio, A.: On the controllability of linear juggling mechanical systems. Systems and control letters 55, 350–367 (2005)
6. Sepulchre, R., Gerard, M.: Stabilization of periodic orbits in a wedge billiard. In: Proc. 42nd IEEE Conference on Decision and Control, December 9-12, 2003, pp. 1568–1573 (2003)
7. Sanfelice, R.G., Teel, A.R., Sepulchre, R.: A hybrid systems approach to trajectory tracking control for juggling systems. In: Proc. 46th IEEE Conference on Decision and Control, December 12–14, 2007, pp. 5282–5287 (2007)
8. Schild, A., Egerstedt, M.: Orbital control for a class of planar impulsive hybrid systems with controllable resets. Technical report, GRITS Lab, Georgia Institute of Technology (2008), http://www.ece.gatech.edu/~magnus/HSCC09IHSCR.pdf
9. Fradkov, A., Progromsky, A.: Introduction to control of oscillations and chaos. World Scientific, Singapore (1998)
10. Schild, A., Lunze, J.: Stabilization of limit cycles of discretely controlled continuous systems by controlling switching surfaces. In: Bemporad, A., Bicchi, A., Buttazzo, G. (eds.) HSCC 2007. LNCS, vol. 4416, pp. 515–528. Springer, Heidelberg (2007)

Distributed Tree Rearrangements for Reachability and Robust Connectivity

Michael Schuresko[1] and Jorge Cortés[2]

[1] Department of Applied Mathematics and Statistics
University of California, Santa Cruz
mds@soe.ucsc.edu
[2] Department of Mechanical and Aerospace Engineering
University of California, San Diego
cortes@ucsd.edu

Abstract. We study maintenance of network connectivity in robotic swarms with discrete-time communications and continuous-time motion capabilities. Assuming a network topology induced by spatial proximity, we propose a coordination scheme which guarantees connectivity of the network by maintaining a spanning tree at all times. Our algorithm is capable of repairing the spanning tree in the event of link failure, and of transitioning from any initial tree to any other tree which is a subgraph of the communications graph.

1 Introduction

Given a group of robots with processing, motion, and communication capabilities executing a motion coordination algorithm, we address the following problem: how can we guarantee that the interaction graph induced by the inter-agent communication remains connected?

One strategy is to make custom modifications to each motion control algorithm to guarantee connectivity. It is desirable, however, to synthesize a general methodology that goes beyond a case by case study, and can be used in conjunction with any motion coordination algorithm. In this paper we take on this aim and propose an approach based on the preservation of a spanning tree of the underlying communication graph. The idea is to synthesize a distributed algorithm to agree upon "safe" re-arrangements of the spanning tree (i.e., re-arrangements that do not break connectivity) based on preferences specified by the motion coordination algorithm. Space constraints prevent us from presenting more than a rough sketch of our algorithm design and analysis results. A complete version of our discussion here can be found in [13].

Literature review. The fundamental importance of spanning trees to distributed algorithms motivate a vast collection of literature, see e.g., [7,9], which explores their properties and designs algorithms to construct them. Spanning trees are especially crucial for the specific case of distributed computation over

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 470–474, 2009.

ad-hoc networks. For a survey of spanning tree repair algorithms for ad-hoc networks, see [4]. In cooperative control and robotics, several works have studied how to constrain the motion of the agents to preserve connectivity. For brevity, we only mention a few here. [1,6,5,10] maintain the connectivity of the network by constraining robot motion to maintain a fixed set of links. The centralized solution proposed in [15] allows for a general range of agent motions. Many works [16,3,14,12] control the algebraic connectivity of a robotic network.

2 Connectivity Maintenance Algorithm

The Connectivity Maintenance (CM) Algorithm is an algorithm to maintain a spanning tree of the communication graph. The intent is that if robot motion is constrained to not break any links of the spanning tree, the underlying graph will remain connected as well. An informal algorithm description follows.

> *[Informal description:]* Each robot maintains a reference to its parent in the spanning tree. At pre-arranged times, each robot is allowed to change its parent. Connectivity is preserved in the following way. Each robot keeps an estimate of its depth, i.e., distance from the root in the spanning tree. If no robot picks a robot of greater depth than its parent's, then no robot will pick one of its current descendants as a parent node. To allow robots to attach to potential parents of the same depth estimate, a tie-breaking algorithm based on UIDs is used to prevent potential formation of cycles, thus maintaining the spanning tree property.

The CM Algorithm should be coupled with two other algorithms:

- The first is a modification of the underlying motion coordination algorithm, modified to preserve the links of the spanning tree maintained by CM Algorithm. We refer to an algorithm which satisfies the constraints required for this role as one which is *motion compatible with* CM Algorithm. This algorithm also specifies which neighbors each agent would prefer to be connected to as an order relation. The relation we use in the simulations presented in Section 4 is roughly "each agent prefers to attach to agents which are closer to its position in physical space."
- The second algorithm is one which tells the robots to artificially increase their depth estimates at particular times. Doing so allows a robot of a lower actual depth to attach to a robot of a higher actual depth, at the expense of making the "depth estimates" diverge from the actual depth (hopefully only for short periods of time). Care must be taken to ensure that such an algorithm does not cause robot depth estimates to grow in an unbounded fashion. We specify a series of constraints on such an algorithm so that it still guarantees correctness of CM Algorithm. We refer to an algorithm which satisfies these constraints as one which is *depth compatible with* CM Algorithm. The simplest such algorithm, called Null Depth Increment Algorithm, never tells an agent to artificially increase its depth estimate.

We show in [13] that CM Algorithm can recover from a wide variety of states (positions and topologies) resulting from link failures of the form "a link between two agents disappears who are instantly made aware of the link failure."

3 Reachability Analysis: Cycle-Detecting Depth Increment Algorithm

We introduce an algorithm which is *depth compatible with* CM Algorithm called Cycle-detecting Depth Increment Algorithm. When this algorithm is combined with CM Algorithm, the resulting strategy satisfies a very nice property: the combined algorithm can induce the constraint tree to match any tree, T_2, which is a subgraph of the communication graph. Specifically, if a tree, T_2, is a subgraph of the current graph, and every edge of T_2 is preferred by its parent, i, in T_2 to each of i's neighbors, then the tree stored by CM Algorithm will eventually become T_2.

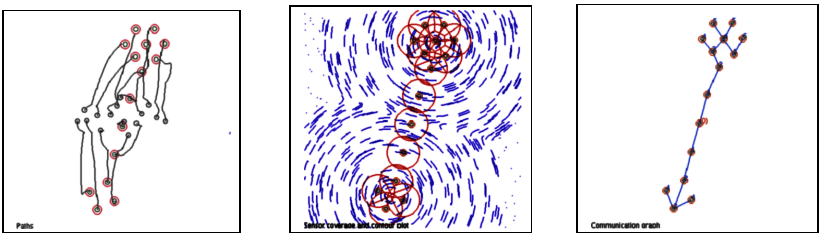
An informal description of Cycle-detecting Depth Increment Algorithm is as follows.

> *[Informal description:]* Each robot stores a "start number", a "number of descendants" and a "mapping from child UID to child start number." At each round, in addition to the tree constraint info, each node sends the following info to each neighbor. If the neighbor is a child, it sends the appropriate entry in its mapping, or, if the child is not in the mapping, it sends its own start number. It always sends its "number of descendants." If the neighbor is not a child, it sends its own start number. With the messages received, each node updates its numbers in the following way. Its "number of descendants" is the sum of the "number of descendants" info received from each child, plus one (for itself). Its "start number" is the number its parent sends it. For each child it receives a message for, it adds an entry to its map that is indexed by that child's UID and has a value of "the sum, over all children with lesser UID, of the number of descendants of those children, plus one plus its own start number."

4 Simulations

Here we illustrate the performance of the Connectivity Maintenance Algorithm in several simulations. We combine the algorithm with Cycle-detecting Depth Increment Algorithm and the deployment algorithm presented in [2]. The proximity graph of the robotic network is the r-disk proximity graph. The deployment algorithm assumes that each robot has a sensor coverage disk. It moves the robots to maximize sensor coverage of a "region of interest" represented by a density function $\rho : \mathbb{R}^2 \mapsto \mathbb{R}$ given sensors which cover disks of radius r_d. We assume the robots have a maximum velocity of v_{speed}.

Links of the constraint tree are preserved via a modification of the procedure described in [1]. To preserve a link between two robots, we constrain the motion



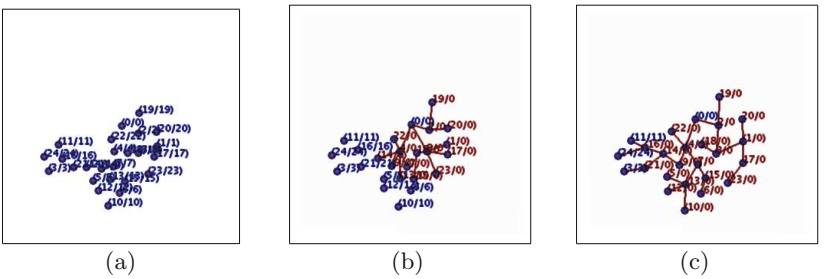
(a) (b) (c)

Fig. 1. The plots show an execution of Connectivity Maintenance Algorithm, showing (a) the paths taken by the robots, (b) a contour plot of the density field and the sensor coverage regions of the robots, (c) the final network constraint tree

of the two robots to a circle of radius $\frac{r}{2}$ centered at the midpoint of the line between their positions. Because each robot has a "target" it moves towards, we can find the closest point to the target in the intersection of the circles generated by the constraint edges.

The algorithm resulting from the combination of the deployment algorithm with the Connectivity Maintenance Algorithm is executed in our Java simulation platform [11]. This platform provides a software implementation of the modeling framework introduced in [8]. Our results are shown in Figure 1.

Figure 2 shows the evolution of the algorithm when repairing an initially disconnected tree.



(a) (b) (c)

Fig. 2. Progress of repair starting with an initially disconnected constraint tree. Agents are labelled by "agent id/root id" : those in blue have not yet completed repair.

5 Conclusions and Future Work

We have designed a tree-rearrangement algorithm for connectivity with reachability and repair capabilities. The algorithm can be shown to be provably correct, and is easily composable with other motion coordination algorithms. Future work will include understanding how the resulting trees of our algorithm compare to minimum spanning trees, developing systematic ways to encode preference rearrangements in connection with other coordination algorithms, and exploring the properties of the algorithm in conjunction with other proximity graphs, such as the visibility graph.

Acknowledgments

This research was supported in part by NSF CAREER Award ECS-0546871.

References

1. Ando, H., Oasa, Y., Suzuki, I., Yamashita, M.: Distributed memoryless point convergence algorithm for mobile robots with limited visibility. IEEE Transactions on Robotics and Automation 15(5), 818–828 (1999)
2. Cortés, J., Martínez, S., Bullo, F.: Spatially-distributed coverage optimization and control with limited-range interactions. ESAIM. Control, Optimisation & Calculus of Variations 11(4), 691–719 (2005)
3. de Gennaro, M.C., Jadbabaie, A.: Decentralized control of connectivity for multi-agent systems. In: IEEE Conf. on Decision and Control, San Diego, CA, pp. 3628–3633 (December 2006)
4. Gaertner, F.C.: A survey of self-stabilizing spanning-tree construction algorithms. Technical report, Ecole Polytechnique Fdrale de Lausanne (2003), http://infoscience.epfl.ch/search.py?recid=52545
5. Ganguli, A., Cortés, J., Bullo, F.: Multirobot rendezvous with visibility sensors in nonconvex environments. In: IEEE Transactions on Robotics (accepted, 2008) (to appear)
6. Lin, J., Morse, A.S., Anderson, B.D.O.: The multi-agent rendezvous problem. Part 1: The synchronous case. SIAM Journal on Control and Optimization 46(6), 2096–2119 (2007)
7. Lynch, N.A.: Distributed Algorithms. Morgan Kaufmann, San Francisco (1997)
8. Martínez, S., Bullo, F., Cortés, J., Frazzoli, E.: On synchronous robotic networks – Part I: Models, tasks and complexity. IEEE Transactions on Automatic Control 52(12), 2199–2213 (2007)
9. Peleg, D.: Distributed Computing. A Locality-Sensitive Approach. In: Monographs on Discrete Mathematics and Applications. SIAM, Philadelphia (2000)
10. Savla, K., Notarstefano, G., Bullo, F.: Maintaining limited-range connectivity among second-order agents. SIAM Journal on Control and Optimization (2007); special issue on Control and Optimization in Cooperative Networks (submitted, November 2006) (to appear)
11. Schuresko, M.D.: CCLsim. A simulation environment for robotic networks (2008), http://www.soe.ucsc.edu/~mds/cclsim
12. Schuresko, M.D., Cortés, J.: Distributed motion constraints for algebraic connectivity of robotic networks. Journal of Intelligent and Robotic Systems (2008); special Issue on Unmanned Autonomous Vehicles (submitted)
13. Schuresko, M.D., Cortés, J.: Distributed tree rearrangements for reachability and robust connectivity. SIAM Journal on Control and Optimization (submitted, 2009)
14. Yang, P., Freeman, R.A., Gordon, G., Lynch, K.M., Srinivasa, S., Sukthankar, R.: Decentralized estimation and control of graph connectivity for mobile sensor networks. In: American Control Conference, Seattle, WA (2008)
15. Zavlanos, M.M., Pappas, G.J.: Controlling connectivity of dynamic graphs. In: IEEE Conf. on Decision and Control and European Control Conference, Seville, Spain, pp. 6388–6393 (December 2005)
16. Zavlanos, M.M., Pappas, G.J.: Potential fields for maintaining connectivity of mobile networks. IEEE Transactions on Robotics 23(4), 812–816 (2007)

The Sensitivity of Hybrid Systems Optimal Cost Functions with Respect to Switching Manifold Parameters*

Farzin Taringoo and Peter E. Caines

Department of Electrical and Computer Engineering and Centre for Intelligent Machines, McGill University, Montreal, Canada
{taringoo,peterc}@cim.mcgill.ca

Abstract. This paper presents an analysis of hybrid systems performance with respect to the variation of switching manifold parameters. The problem presented here has two aspects: (i) the general hybrid control problem and (ii) the variation of the switching manifold configurations which determine the autonomous (uncontrolled) discrete state switchings. The optimal cost variation (i.e. derivative) as a function of the switching manifold parameters is described by the solution of a set of differential equations generating the state and costate sensitivity functions. An example is presented to illustrate the main result of the paper.

Keywords: Hybrid Control Systems, Switching Manifolds, Variational Methods, Optimal Control.

1 Introduction

Problems of hybrid systems optimal control (HSOC) has been studied and analyzed in many papers, see e.g [3,4,7]. One direct extension of the optimal control problem for autonomous hybrid systems concerns the notion of switching manifold geometry, where this is interpreted as the shaping and displacement of switching manifolds in order to optimize system performance, [5], [1,2]. The systems studied in [1,2] are (switched) autonomous hybrid control systems (AHCS), that is to say, the hybrid systems have no continuous control in their distinct phases, and the discrete state switchings occur autonomously, i.e. where the continuous component of the state trajectory passes through a switching manifold.

In this paper we analyze HSOC sensitivity with respect to the parameters determining a system's switching manifolds. Although attention is restricted here to the purely autonomous switching case with one switching event, neither restriction is a necessary feature of the theory (see [6]). The results of [1,2] are recovered from those in this paper by removing the continuous inputs. The corresponding multiple switching case is analyzed in [6]. Subsequent work will focus on the influence of geometric properties of the switching manifolds, for instance curvature and global topology, and on the optimality, sensitivity and robustness properties of the associated autonomous HOC problem solutions.

* This work was supported by an NSERC Discovery Grant.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 475–479, 2009.

2 Hybrid Systems

Within the standard hybrid systems framework (see e.g. [3]), the formulation of a hybrid system $\mathbf{H} := \{H = Q \times \mathbf{R^n}, I = \mathbf{\Sigma} \times U, F, A, \mathbf{\Gamma}, \mathcal{M}\}$ is given in terms of discrete state space Q; continuous state space R^n; discrete input set Σ; continuous input value space U; family of controlled vector fields F, discrete state transition automaton A, time independent (partially defined) discrete transition map Γ and set of switching manifolds $\mathcal{M}$. A hybrid system input is a triple $I := (\tau, \sigma, u)$ defined on a half open interval $[t_0, T), T \leq \infty$, where $u \in \mathcal{U}$ and (τ, σ) is a hybrid switching sequence $(\tau, \sigma) = ((t_0, \sigma_0), (t_1, \sigma_1), (t_2, \sigma_2), \ldots)$ of pairs of switching times and discrete input events, $\sigma_0 = id, \sigma_i \in \Sigma, i \geq 1$, and where σ is called a location or discrete state sequence. Let $\{l_j\}_{j \in Q}, l_j \in C^k(R^n \times U; R_+), k \geq 1$ be a family of loss functions and $h \in C^k(R^n; R_+), k \geq 1$, a terminal cost satisfying the following hypotheses:
A1: There exist $K_l < \infty$ and $1 \leq \gamma < \infty$ such that $|l_j(x,u)| \leq K_l(1 + ||x||^\gamma), x \in R^n, u \in U, j \in Q$.
A2:There exist $K_h < \infty, 1 \leq \delta < \infty$ such that $|h(x)| \leq K_h(1 + ||x||^\delta), x \in R^n$.
Consider the HSOC with initial time t_0, final time $t_f < \infty$, initial hybrid state $h_0 = (q_0, x_0)$, and $\bar{L} < \infty$. Let $S_L = ((t_0, \sigma_0), (t_1, \sigma_1), \ldots, (t_L, \sigma_L))$ be a hybrid switching sequence and let $I_L := (S_L, u), u \in \mathcal{U}$, be a hybrid input trajectory, where $L \leq \bar{L} < \infty$ is the number of switchings. A hybrid cost function is defined as

$$J(t_0, t_f, h_0; I_L, \bar{L}, \mathcal{U}) := \sum_{i=0}^{L} \int_{t_i}^{t_{i+1}} l_{q_i}(x_{q_i}(s), u(s))ds + h(x_{q_L}(t_f) \qquad (2.1)$$

where (see [3] for all details) the continuous dynamics of the hybrid system are specified as follows:

$$\dot{x}_{q_i}(t) = f_{q_i}(x_{q_i}(t), u(t)), \quad a.e.\ t \in [t_i, t_{i+1}),$$

$$u(t) \in U \subset R^u, u(.) \in L_\infty(U), \quad h_0 = (q_0, x_0), \quad i = 0, 1, \ldots, L,$$

$$x_{q_{i+1}}(t_{i+1}) = lim_{t \to t_{i+1}} x_{q_i}(t), \quad t_{L+1} = t_f < \infty. \qquad (2.2)$$

∎

3 Problem Formulation and Main Result

The switching manifolds $\mathcal{M}$ considered in this paper depend upon time, state and, in addition, a parameter $\alpha \in R^m$. Locally they are specified by the equations

$$m_{p,q}(x, t, \beta) = 0, \quad x \in R^n, \quad \beta \in \mathcal{N}(\alpha) \subset R^m, p, q \in Q, \qquad (3.3)$$

where $\mathcal{N}(\alpha)$ is an open neighborhood of the nominal parameter α and $m_{p,q}$ is a continuously differentiable function. Let $V(t_0, t_f, h_0, \alpha)$ denote the value function,

$$V(t_0, t_f, h_0, I_L, \bar{L}, \mathcal{U}, \alpha) := inf_{I_L} J(t_0, t_f, h_0, I_L, \bar{L}, \mathcal{U}, \alpha), \qquad (3.4)$$

and denote by t^α the associated optimal switching time for the nominal manifold parameter α, which is assumed to be unique. In this setting, the motivating problem for this work is to find values of α which infimize the total α dependent value function. Let us write $x^\alpha(.), x^\beta(.)$ for the optimal state trajectories corresponding to the nominal and perturbed parameters respectively, and let $u^\alpha(.), u^\beta(.)$ be the associated optimal controls. Define

$$H_q(x,\lambda,u) = \lambda^T f_q(x,u) + l_q(x,u), \quad x,\lambda \in R^n, u \in U, q \in Q. \tag{3.5}$$

Then to each optimal trajectory there is associated a piecewise absolutely continuous adjoint process satisfying

$$\dot{\lambda}_j = -\frac{\partial H_j}{\partial x}(x,\lambda,u), \qquad t \in (t_j, t_{j+1}], \tag{3.6}$$

together with the boundary conditions given in [3]. Let us define the state and adjoint variables sensitivities for the nominal and perturbed manifold parameters as:

$$y(t) = lim_{\delta t^\alpha \to 0} \frac{\delta x(t)}{\delta t^\alpha}, \qquad z(t) = lim_{\delta t^\alpha \to 0} \frac{\delta \lambda(t)}{\delta t^\alpha}, \qquad t \in [t_0, t_f]. \tag{3.7}$$

where $\delta x(t) := x^\beta(t) - x^\alpha(t), \quad \delta\lambda(t) := \lambda^\beta(t) - \lambda^\beta(t), \quad \delta t^\alpha := t^\beta - t^\alpha$. Assume that there exists a continuously differentiable one to one mapping between α and t^α in the neighborhood $\mathcal{N}(\alpha)$, so locally $\beta \to \alpha$ if and only if $t^\beta \to t^\alpha$.

Theorem 1. [6] Consider a hybrid system (2.2) possessing two modes q_1, q_2:

$$\dot{x}_1 = f_1(x_1(t), u_1(t)), \quad t \in [0, t_s), \quad \dot{x}_2 = f_2(x_2(t), u_2(t)), \quad t \in [t_s, t_f], \tag{3.8}$$

for which the cost function is defined by (2.1). Assume that $f_i, l_i \in C^2, \quad i = 1, 2$. Then the optimal state and adjoint variable sensitivities with respect to the switching time are given by

$$y(t) = \int_{t_0}^{t} F_{1(x,\lambda)}(y(\tau), z(\tau))d\tau, \qquad t \in [0, t^\alpha), \tag{3.9}$$

$$y(t) = \int_{t_0}^{t^\alpha} F_{1(x,\lambda)}(y(\tau), z(\tau))d\tau + R_1 + \int_{t^\alpha}^{t} F_{2(x,\lambda)}(y(\tau), z(\tau))d\tau, \qquad t \in [t^\alpha, t_f],$$

together with

$$\begin{aligned} z(t) \;=& \frac{\partial^2 h}{\partial x^2} y(t_f) + \int_{t^\alpha}^{t_f} H^2_{2(x,\lambda)}(y(\tau), z(\tau))d\tau + \int_{t}^{t^\alpha} H^2_{1(x,\lambda)}(y(\tau), z(\tau))d\tau \\ &+ \bar{H}_{(1,2)}(x,\lambda) + \frac{\partial p^\alpha \nabla_x m(x^\alpha, t^\alpha)}{\partial x^\alpha}(y(t^\alpha) + f_1(x(t^\alpha), \lambda_1^\alpha(t^\alpha))) \\ &+ \frac{\partial p^\alpha \nabla_x m(x^\alpha, t^\alpha)}{\partial \lambda_1}(z^-(t^\alpha) - \frac{\partial H_1}{\partial x}(x^\alpha(t^\alpha), \lambda_1^\alpha(t^\alpha))) \\ &+ \frac{\partial p^\alpha \nabla_x m(x^\alpha, t^\alpha)}{\partial \lambda_2}(z^+(t^\alpha) - \frac{\partial H_2}{\partial x}(x^\alpha(t^\alpha), \lambda_2^\alpha(t^\alpha))) \\ &+ \frac{\partial p^\alpha \nabla_x m(x^\alpha, t^\alpha)}{\partial t^\alpha} + \frac{\partial p^\alpha \nabla_x m(x^\alpha, t^\alpha)}{\partial \alpha}\frac{\partial \alpha}{\partial t^\alpha}, \qquad t \in [0, t^\alpha). \end{aligned} \tag{3.10}$$

$$z(t) = \int_t^{t_f} H^2_{2(x,\lambda)}(y(\tau), z(\tau))d\tau + \frac{\partial^2 h}{\partial x^2}(x^\alpha(t_f))y(t_f), \qquad t \in [t^\alpha, t_f]. \tag{3.11}$$

where p^α is the adjoint variable discontinuity parameter and

$$z^-(t^\alpha) = lim_{t\uparrow t^\alpha} z(t), \quad z^+(t^\alpha) = lim_{t\downarrow t^\alpha} z(t) \tag{3.12}$$

$$R_1 = f_1(x^\alpha(t^\alpha), \lambda_1^\alpha(t^\alpha)) - f_2(x^\alpha(t^\alpha), \lambda_2^\alpha(t^\alpha)), \tag{3.13}$$

$$F_{i(x,\lambda)}(y(t), z(t)) = \nabla_{(x,\lambda)} f_i(x^\alpha(t^\alpha), \lambda_i^\alpha(t^\alpha)).[y(t), z(t)]^T, \quad i = 1, 2. \tag{3.14}$$

$$\bar{H}_{(1,2)}(x, \lambda) = \frac{\partial H_1}{\partial x}(x^\alpha(t^\alpha), \lambda_1^\alpha(t^\alpha)) - \frac{\partial H_2}{\partial x}(x^\alpha(t^\alpha), \lambda_2^\alpha(t^\alpha)). \tag{3.15}$$

$$H^2_{i(x,\lambda)}(y(t), z(t)) = \frac{\partial}{\partial x}\nabla_{(x,\lambda)} H_i(x^\alpha(t), \lambda_i^\alpha(t)).[y(t), z(t)]^T, \quad i = 1, 2. \tag{3.16}$$

∎

Here we note that the optimal control $u(.)$ in $H(.,.)$ is replaced by $\lambda(.)$, for detailed information see [6].

4 Example

Here we present an example which illustrates the results above. In this case, since analytic solutions are not available, the optimal switching time and state for the nominal α are obtained numerically via the HMPC algorithm [3]. Consider the hybrid system with two modes given by:

$$\dot{x}(t) = x(t) + u(t), \quad t \in [0, t^\alpha], \quad \dot{x}(t) = -x(t) + u(t), \quad t \in (t^\alpha, 2], \tag{4.17}$$

where the switching manifold is the following time varying structure:

$$m(x(t), \alpha, t) = x - t - \alpha = 0, \quad t \in [0, 2]. \tag{4.18}$$

The cost function for the hybrid system is chosen to be

$$J(0, 2, h_0, I_L, \bar{L}, \mathcal{U}) = \frac{1}{2}\int_0^{t^\alpha} u^2(t)dt + \frac{1}{2}\int_{t^\alpha}^2 u^2(t)dt, \tag{4.19}$$

where $\bar{L} = 2$ and $h_0 = (0, 1)$. In this example we vary α between 0 to 0.5 and the optimal cost is then obtained as a function of the manifold parameter. Figure 1 displays the optimal cost variation (i.e. derivative) displayed as a function of the switching time t^α and the optimal cost variation (i.e. derivative) as a function of the manifold parameter α.

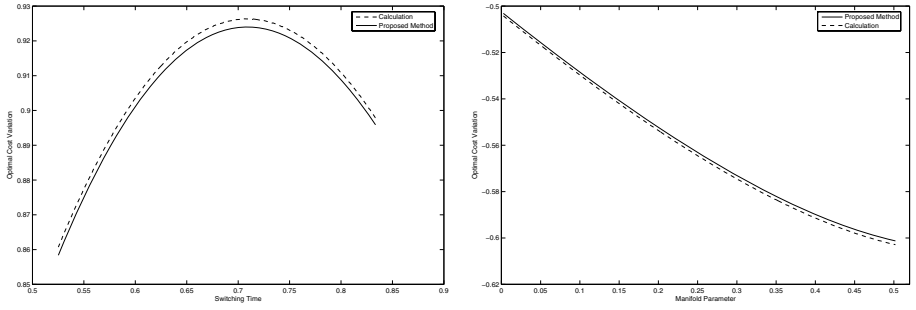


Fig. 1. Optimal Cost Derivative versus Switching Time (left) and Manifold Parameter (right). Solid lines are computed via Theorem 1 and dashed lines are obtained by direct calculation.

References

1. Boccadoro, M., Valigi, P., Wardi, Y.: A method for the design of optimal switching surfaces for autonomous hybrid systems. In: Bemporad, A., Bicchi, A., Buttazzo, G. (eds.) HSCC 2007. LNCS, vol. 4416, pp. 650–655. Springer, Heidelberg (2007)
2. Boccadoro, M., Wardi, Y., Egersted, M., Verriest, E.: Optimal Control of Switching Surfaces in Hybrid Dynamical systems. Journal of Discrete Event Dynamic Systems 15, 433–448 (2005)
3. Shaikh, M.S., Caines, P.E.: On the Hybrid Optimal Control Problems: Theory and Algorithms. IEEE Trans on Automatic Control 52(9), 1587–1603 (2007)
4. Shaikh, M.S., Caines, P.E.: On the Optimal Control of Hybrid Systems: Analysis and Algorithms for Trajectory and schedule Optimization. In: Proc. 42nd IEEE Int. Conf. Decision Control, Maui, HI, pp. 2144–2149 (2003)
5. Schild, A., Lunze, J.: Stabilization of Limit Cycles of Discreetly Controlled Continuous Systems by Controlling Switching Surfaces. In: Bemporad, A., Bicchi, A., Buttazzo, G. (eds.) HSCC 2007. LNCS, vol. 4416, pp. 515–528. Springer, Heidelberg (2007)
6. Taringoo, F., Caines, P.E.: Optimization of Switching Manifolds in Optimal Hybrid Control Systems, Technical Report, McGill University (January 2008)
7. Xu, X., Antsaklis, J.: Optimal Control of Switched Systems Based on Parametrization of the Switching Instants. IEEE Trans. Automatic Control 49(1), 2–16 (2004)

STORMED Hybrid Games

Vladimeros Vladimerou, Pavithra Prabhakar, Mahesh Viswanathan, and Geir Dullerud

University of Illinois at Urbana-Champaign

Abstract. We introduce STORMED hybrid games (*SHG*), a generalization of STORMED Hybrid Systems [15], which have natural specifications, allow rich continuous dynamics and admit various properties to be decidable. We solve the control problem for *SHG* using a reduction to bisimulation on game graphs. This reduction generalizes to a greater family of games, which includes o-minimal hybrid games [5]. We also solve the optimal-cost reachability problem for Weighted *SHG*.

1 Introduction

Hybrid automata are a popular formalism for modelling and verifying embedded systems. *Hybrid games* [11,5,6] have been extensively used for modelling and designing *hybrid* controllers to control embedded systems. They are defined similar to hybrid automata but with discrete transitions partitioned into controllable and uncontrollable transitions.

We introduce STORMED hybrid games (*SHG*) defined using STORMED hybrid systems (*SHS*) formalism [15]. *SHG* games allow for richer continuous dynamics than the other popular decidable formalisms like rectangular hybrid games [11] and timed games [2,3]. Also they admit a stronger coupling between the continuous and discrete state components than found in o-minimal hybrid games [5].

Our main result is that for regular winning objectives, the controller synthesis problem is decidable, provided the o-minimal theory used to describe the *SHG* is decidable.

Next we consider weighted hybrid games, where there is a cost associated with each of the game choices, and the goal is to design optimal (cost) winning strategies for the controller. We show that weighted *SHS*(WSHS) with reachability objectives are decidable (and the controller synthesizable) when the underlying o-minimal theory is decidable.

Related Work

The controller synthesis problem for real-time and hybrid systems has attracted a lot of attention since [2] and [13]. Symbolic algorithms for the controller synthesis problem were first presented in [10]. Generally one assumes that a controller can examine the state at various times, and can influence the transitions taken. When the controller can choose not only a discrete transition but also when it is taken, the problem is known to be undecidable for initialized rectangular automata [12], and decidable for timed automata [13], and o-minimal hybrid automata [5] with decidable theories. Here we extend these observations to STORMED systems.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 480–484, 2009.

Zeno behavior must be dealt in a dense time setting [9]. As in [10,8] we eliminate it via by semantic constraints imposed on the winning conditions.

Weighted timed games were first considered in [1,4]. Synthesizing the optimal cost controller for reachability is undecidable for timed automata [7], but decidable for o-minimal hybrid systems [6] with decidable underlying theories. We show that optimal reachability is decidable for STORMED games.

Note. Due to lack of space, we refer the reader to [16] for more details, including proofs and intermediate lemmas and definitions. Here, we present only the most important definitions and results in the following sections.

2 Decidability of Control for STORMED Hybrid Games

Definition 1. *A* hybrid game $\mathcal{H}$ *is a tuple* $(Loc, Act_C, Act_U, Labels, Cont, Edge, Inv, Flow, Reset, Guard, Lfunc)$ *where:*

- Loc *is a finite set of locations,*
- Act_C *is a finite set of controllable actions,*
- Act_U *is a finite set of uncontrollable actions,*
- $Labels$ *is a finite set of state labels,*
- $Cont = \mathbb{R}^n$ *for some* n*, is a set of continuous states,*
- $Edge \subseteq Loc \times (Act_C \cup Act_U) \times Loc$ *is a set of edges,*
- $Inv : Loc \to 2^{Cont}$ *is a function that associates with every location an invariant,*
- $Flow : Loc \times Cont \to (\mathbb{R}^+ \to Cont)$ *is a flow function,*
- $Guard : Edge \to 2^{Cont}$ *is a function that assigns to each edge a guard,*
- $Reset : Edge \to 2^{Cont \times Cont}$ *is a reset function, and*
- $Lfunc : Loc \times Cont \to Labels$ *is a state labeling function.*

At each step of the game, the controller and the environment have two choices: either to let time pass for t time units or to take a controllable (or uncontrollable) transition enabled at the state. If both the controller and the environment pick time, then the system evolves continuously for the shorter of the two durations. If exactly one of them picks a discrete transition, then the discrete transition chosen is taken and finally, in the case when both pick discrete transitions, the controller's choice is respected. A play is an alternating sequence of states and transitions. From each state both the controller and the environment propose a transition, and the transition followed by it in the play is chosen according to the above rule. A strategy for the controller tells the transition that needs to be taken given the information of the play till then. A play conforms to a strategy if the controller selects the transitions according to the strategy. A trace is an alternating sequence of state labels and actions. The trace of a play is the sequence of labels of its states and the actions. A winning condition is a set of admissible traces. A strategy for the controller is winning with respect to a winning condition if the trace of every play conforming with the strategy is admissible according to the winning condition. We consider winning conditions which are ω-regular. The *control problem* is to decide given a hybrid game and a winning condition if the controller has a strategy which is winning. Further, the controller synthesis problem is to come up with such a strategy. The formal semantics of a hybrid game is given in terms of a game graph and can be found in [16].

We now define STORMED hybrid games.

Definition 2. *A* STORMED hybrid game *is defined as a hybrid game with the following restrictions.*

S. Guards are *S*eparable:
For all $l_1, l_2 \in \mathit{Loc}$ such that $l_1 \neq l_2$, $\mathit{dist}(\mathcal{G}(l_1), \mathcal{G}(l_2)) = \inf\{||x-y|| \,|\, x \in \mathcal{G}(l_1), y \in \mathcal{G}(l_2)\} > 0$.

T. The flow is time-independent spatially consistent (*T*ISC):
For every state $(l, x) \in \mathit{Loc} \times \mathit{Cont}$, $\mathit{Flow}(l, x)$ is continuous and $\mathit{Flow}(l, x)(0) = x$, and for all $t, t' \in \mathbb{R}^+$, $\mathit{Flow}(l, x)(t + t') = \mathit{Flow}(l, \mathit{Flow}(l, x)(t))(t')$.

O. The guards, invariants, flows and resets are definable in an *o*-minimal[1] theory, that is, by a first order formula of the theory.

RM. Resets and flows are monotonic along some vector ϕ:
There exists $\epsilon > 0$ such that for all $l \in \mathit{Loc}, x \in \mathit{Cont}$ and $t, \tau \in \mathbb{R}^+$, $\phi \cdot (\mathit{Flow}(l, x)(t + \tau) - \mathit{Flow}(l, x)(t)) \geq \epsilon ||\mathit{Flow}(l, x)(t + \tau) - \mathit{Flow}(l, x)(t)||$.
There exist $\epsilon, \zeta > 0$ such that for all $l_1, l_2 \in \mathit{Loc}$ and $x_1, x_2 \in \mathit{Cont}$ such that $(x_1, x_2) \in \mathit{Reset}(l_1, l_2)$:
- if $l_1 = l_2$, then either $x_1 = x_2$ or $\phi \cdot (x_2 - x_1) \geq \zeta$, and
- otherwise $\phi \cdot (x_2 - x_1) \geq \epsilon ||x_2 - x_1||$.

ED. Guards are *e*nds-*d*elimited along ϕ: The set $\{\phi \cdot x \,|\, x \in \mathcal{G}(l), l \in \mathit{Loc}\} \subseteq [b^-, b^+]$ for some b^- and b^+.

The following theorem states that the control problem is decidable for this class.

Theorem 1. *Given a STORMED hybrid game $\mathcal{H}$ and a winning condition $\mathcal{W}$ which is ω-regular, the control problem is decidable if the underlying o-minimal theory is decidable. The controller synthesis problem is also decidable.*

Proof. Details of the proof are in [16]. It proceeds as follows: We first prove that under special acyclicity conditions, bisimulation equivalence on the *time-abstract* transition system defined by the *SHG* preserves winning (and losing) states which is not true in general for hybrid systems [5]. The time-abstract transition system is the labelled transition system semantics of the *SHG* that ignores the distinction between controllable and uncontrollable transitions and abstracts the time when continuous transitions are taken. We show that both STORMED systems and o-minimal systems meet this technical acyclicity condition. Further the observations that the time-abstract transition system for a *SHS* has a finite bisimulation quotient [15] (which is effectively constructable when the underlying o-minimal theory is decidable) and the fact that finite games with regular objectives are decidable [14], allow us to conclude the decidability of *SHG*. The same argument holds for o-minimal systems. □

3 Weighted Hybrid Games and Hybrid Systems

We now consider weighted games, where transitions have associated costs, and the goal is to minimize these costs while meeting certain qualitative objectives. We consider

[1] A theory is o-minimal if the sets definable by formulas with one variable are a finite union of intervals.

the problem of designing optimal controllers for reachability objectives, and also the problem of verifying hybrid systems with costs.

A *Weighted hybrid game* is a pair $(\mathcal{G}, \mathsf{Cost})$, where the hybrid game $\mathcal{G}$ is equipped with a non-negative and time-non-decreasing cost function $\mathsf{Cost} : \mathit{Loc} \times \mathbb{R}_+ \to \mathbb{R}_+$, i.e., $\mathsf{Cost}(q,t) \geq 0$ for all t and $\mathsf{Cost}(q,t_1) \geq \mathsf{Cost}(q,t_2)$ if $t_1 > t_2$. In addition the cost function satisfies the following additive property $\mathsf{Cost}(q, t_1 + t_2) = \mathsf{Cost}(q,t_1) + \mathsf{Cost}(q,t_2)$. Hence with every move of the controller and the environment, there is an associated cost (the cost associated with a discrete transition can be assumed to be 0). The cost associated with a play is the sum of the cost of all the moves. The cost associated with a controller strategy is the supremum of the costs of all plays conforming with the strategy. Given a Goal, a set of states, a strategy is said to reach the goal if all the plays conforming with the strategy reach the goal. Given a Goal (a set of states), the *optimal cost reachability problem* is to compute the infimum of the costs of the controller strategies which reach goal. If there exists a strategy which achieves the infimum, we call it an optimal strategy.

A *Weighted STORMED hybrid game* is a pair $(\mathcal{G}, \mathsf{Cost})$, where $\mathcal{G}$ is a STORMED hybrid game and the cost function Cost is definable is the o-minimal theory in which $\mathcal{G}$ is defined. The next theorem states that we can solve the optimal-cost reachability problem for Weighted STORMED hybrid games.

Theorem 2. *Given a Weighted STORMED hybrid game* $(\mathcal{G}, \mathsf{Cost})$*, where the underlying theory* $\mathcal{M}$ *is decidable, and a* Goal*, where* Goal *is definable in* $\mathcal{M}$*, the optimal-cost reachability problem is decidable. In fact, we can compute an optimal strategy if one exists.*

First observe that, when considering non-zeno plays [2], if there is a winning strategy λ for the controller then there is a winning strategy in which the controller does not choose a time step if in the previous step the controller chose a time step shorter than the environment. This is challenging to prove due to the fact that these games may not have memoryless winning strategies. We then conclude that for non-zeno reachability games for *SHS*, we need only consider bounded step strategies, and therefore can not only compute the cost of the optimal strategy but also synthesize it. Again, details of the proof are in [16].

References

1. Alur, R., La Torre, S., Pappas, G.J.: Optimal paths in weighted timed automata. In: Di Benedetto, M.D., Sangiovanni-Vincentelli, A.L. (eds.) HSCC 2001. LNCS, vol. 2034, pp. 49–62. Springer, Heidelberg (2001)
2. Asarin, E., Maler, O., Pnueli, A.: Symbolic controller synthesis for discrete and timed systems. In: Antsaklis, P.J., Kohn, W., Nerode, A., Sastry, S.S. (eds.) HS 1994. LNCS, vol. 999, pp. 1–20. Springer, Heidelberg (1995)
3. Asarin, E., Maler, O., Pnueli, A., Sifakis, J.: Controller synthesis for timed automata 1. In: Proc. IFAC Symposium on System Structure and Control, pp. 469–474 (1998)

[2] In the games we consider, zeno plays are allowed; the environment (or controller) could simply pick shorter and shorter time steps, and thereby starve her opponent.

4. Behrmann, G., Fehnker, A., Hune, T., Larsen, K.G., Pettersson, P., Romijn, J.M.T., Vaandrager, F.W.: Minimum-cost reachability for priced timed automata. In: Di Benedetto, M.D., Sangiovanni-Vincentelli, A.L. (eds.) HSCC 2001. LNCS, vol. 2034, pp. 147–161. Springer, Heidelberg (2001)
5. Bouyer, P., Brihaye, T., Chevalier, F.: Control in o-minimal hybrid systems. In: LICS 2006: Proceedings of the 21st Annual IEEE Symposium on Logic in Computer Science, Washington, DC, USA, pp. 367–378. IEEE Computer Society Press, Los Alamitos (2006)
6. Bouyer, P., Brihaye, T., Chevalier, F.: Weighted O-minimal hybrid systems are more decidable than weighted timed automata! In: Artemov, S.N., Nerode, A. (eds.) LFCS 2007. LNCS, vol. 4514, pp. 69–83. Springer, Heidelberg (2007)
7. Brihaye, T., Bruyère, V., Raskin, J.-F.: Model-checking for weighted timed automata. In: Lakhnech, Y., Yovine, S. (eds.) FORMATS 2004 and FTRTFT 2004. LNCS, vol. 3253, pp. 277–292. Springer, Heidelberg (2004)
8. Brihaye, T., Henzinger, T.A., Prabhu, V.S., Raskin, J.-F.: Minimum-time reachability in timed games. In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 825–837. Springer, Heidelberg (2007)
9. Cassez, F., Henzinger, T.A., Raskin, J.-F.: A comparison of control problems for timed and hybrid systems. In: Tomlin, C.J., Greenstreet, M.R. (eds.) HSCC 2002. LNCS, vol. 2289, pp. 134–148. Springer, Heidelberg (2002)
10. de Alfaro, L., Henzinger, T.A., Majumdar, R.: Symbolic algorithms for infinite-state games. In: Larsen, K.G., Nielsen, M. (eds.) CONCUR 2001. LNCS, vol. 2154, pp. 536–550. Springer, Heidelberg (2001)
11. Henzinger, A., Horowitz, B., Majumdar, R.: Rectangular hybrid games. In: Baeten, J.C.M., Mauw, S. (eds.) CONCUR 1999. LNCS, vol. 1664, pp. 320–335. Springer, Heidelberg (1999)
12. Henzinger, T.A., Kopke, P.W., Puri, A., Varaiya, P.: What's decidable about hybrid automata? In: Proc. 27th Annual ACM Symp. on Theory of Computing (STOC), pp. 373–382 (1995)
13. Maler, O., Pnueli, A., Sifakis, J.: On the synthesis of discrete controllers for timed systems. In: Mayr, E.W., Puech, C. (eds.) STACS 1995. LNCS, vol. 900, pp. 229–242. Springer, Heidelberg (1995)
14. Rosner, R.: Modular synthesis of reactive systems. Ph.D. thesis, Weizmann Institute of Science (1992)
15. Vladimerou, V., Prabhakar, P., Viswanathan, M., Dullerud, G.E.: Stormed hybrid systems. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part II. LNCS, vol. 5126, pp. 136–147. Springer, Heidelberg (2008)
16. Vladimerou, V.: Specifications for decidable Hybrid Automata and Games. PhD thesis, University of Illinois at Urbana - Champaign (2009)

Symbolic Branching Bisimulation-Checking of Dense-Time Systems in an Environment*

Farn Wang

Department of Electrical Engineering & Graduate Institute of Electronic Engineering
National Taiwan University, Taiwan, ROC
A full version of the paper is available at
http://cc.ee.ntu.edu.tw/~{}farn.
Tool is available at
http://sourceforge.net/projects/redlib.

Abstract. We present *timed branching bisimulation in an environment* which allows us to design a new bisimulation-checking algorithm with enhanced performance by reducing state spaces with shared environment state information between the model and the specification automatas. We also propose non-Zeno bisimulation in an environment that fully characterizes TCTL formulas. We then report our implementation and experiments with the ideas.

Keywords: branching bisimulation, TCTL, model-checking, timed automata, algorithms, experiment.

1 Introduction

Two systems are *timed branching bisimulation* (*TB-bisimulation*) equivalent [4,7] if any transition that one system can make at a particular time can also be matched by the other system at the same time, and vice versa. Conceptually, a TB-bisimulation is a binary relation between the state sets of two automata. A state pair is in a TB-bisimulation if and only if its elements are TB-bisimulation equivalent. For convenience, given an automaton A, we use $\mathbb{S}(A)$ to denote the set of state equivalence classes of A. In practice, a model $\mathcal{M}$ and a specification $\mathcal{S}$ can themselves be product automata and share a lot of common components. With the traditional techniques of TB-bisimulation checking [3,7,10], we need to manipulate a preliminary TB-bisimulation image of $|\mathbb{S}(\mathcal{E} \times \mathcal{M})| \times |\mathbb{S}(\mathcal{E} \times \mathcal{S})|$ state pairs where for each automata A_1 and A_2, $A_1 \times A_2$ is a product automaton. As can be seen in this image, there is duplicate information in the two $\mathcal{E}$ components. It is our goal to develop a new framework of *TB-bisimulation in a common environment* (*TBE-bisimulation*) to allows us to present a new branching bisimulation checking algorithm that only needs to manipulate $|\mathbb{S}(\mathcal{E} \times \mathcal{M} \times \mathcal{S})|$ state pairs in a preliminary bisimulation image.

For untimed systems, branching bisimulation preserves all properties expressible in the propositional μ-calculus, which subsumes CTL* [2, 5] in

* The work is partially supported by NSC, Taiwan, ROC under grants NSC 97-2221-E-002-129-MY3.

R. Majumdar and P. Tabuada (Eds.): HSCC 2009, LNCS 5469, pp. 485–489, 2009.

expressiveness. However, our TBE-bisimulation does not preserve all properties expressible in the dense-time counterpart of CTL, i.e., TCTL [1]. Another contribution in this work is that we present the *non-Zeno bisimulation in an environment* (*NZE-bisimulation*) which preserves the TCTL properties. We prove that a TBE-bisimulation is strictly contained in its corresponding NZE-bisimulation.

We have implemented our ideas in our TCTL model-checker `RED` 7.1 and carried out experiments to evaluate the ideas. Algorithms and related work are discussed in the full version.

2 Communicating Timed Automata

A *process timed automaton (PTA)* is equipped with a finite set of dense-time clocks and *synchronization events*. Given a PTA A, we let L_A, X_A, and Σ_A denote the mode set, clock set, and event set of A respectively. A *Communicating timed automaton (CTA)* [6] is a pair $\langle A_0, A_1\rangle$ such that A_0 and A_1 are PTAs with $L_{A_0} \cap L_{A_1} = \emptyset$, $X_{A_0} \cap X_{A_1} = \emptyset$, and $\Sigma_{A_0} = \Sigma_{A_1}$.

$\mathbb{R}^{\geq 0}$ is the set of nonnegative real numbers. $\mathbb{N}$ is the set of nonnegative integers. A state ν of a CTA $\langle A_0, A_1\rangle$ is a valuation of $X_{A_0} \cup L_{A_0} \cup X_{A_1} \cup L_{A_1}$ with the following constraints.

- For each $q \in L_{A_0} \cup L_{A_1}$, $\nu(q) \in \{false, true\}$. Moreover for each $i \in \{0, 1\}$, there exists a unique $q \in L_{A_i}$ such that $\nu(q) \wedge \forall q' \in L_{A_i} - \{q\}(\neg\nu(q'))$ is true. Given $q \in L_{A_i}$, if $\nu(q)$ is true, we denote q as $\mathtt{mode}_{A_i}(\nu)$.
- For each $x \in X_{A_0} \cup X_{A_1}$, $\nu(x) \in \mathbb{R}^{\geq 0}$.

In addition, we require that $\nu \models V_{A_0} \wedge V_{A_1}$. Also for convenience, we now change the meaning of $\mathbb{S}\langle A_0, A_1\rangle$ to the set of states of $\langle A_0, A_1\rangle$.

For any state ν and real number $t \in \mathbb{R}^{\geq 0}$, $\nu + t$ is a state identical to ν except that for every clock $x \in X_{A_0} \cup X_{A_1}$, $(\nu + t)(x) = \nu(x) + t$. Also given a state ν and process transition e of A, νe is the destination state from ν through e.

A process transition e_1 of a PTA A_0 and an e_2 of another PTA A_1 are *compatible* iff they observe the same events. A *global transition* of a CTA $\langle A_0, A_1\rangle$ is a pair of two compatible transitions respectively of A_0 and A_1. A *run* of a CTA $\langle A_0, A_1\rangle$ is an infinite sequence of state-time pairs $(\nu_0, t_0)(\nu_1, t_1)\ldots(\nu_k, t_k)\ldots\ldots$ such that $t_0 t_1 \ldots t_k \ldots\ldots$ is a non-decreasing and divergent real-number sequence; and for all $k \geq 0$, there is an $(e, f) \in T\langle A_0, A_1\rangle$ such that the CTA can go from ν_k to ν_{k+1} through a global transition (e, f) after $t_{k+1} - t_k$ time units, in symbols $\nu_k \xrightarrow{t_{k+1}-t_k,(e,f)} \nu_{k+1}$. A *run segment* is a finite prefix of a run.

3 Timed Branching Bisimulations in an Environment

Suppose we have three PTAs $\mathcal{E}$, $\mathcal{M}$, and $\mathcal{S}$ which represent an environment, a model, and a specification respectively. We let $\mathbb{S}\langle\mathcal{E}, \mathcal{M}, \mathcal{S}\rangle$ denote the set of state pairs (μ, ν) such that $\mu \in \mathbb{S}\langle\mathcal{E}, \mathcal{M}\rangle$, $\nu \in \mathbb{S}\langle\mathcal{E}, \mathcal{S}\rangle$, and for all $x \in L_{\mathcal{E}} \cup X_{\mathcal{E}}$, $\mu(x) = \nu(x)$. A run segment $(\nu_0, t_0)\ldots(\nu_k, t_k)$ of $\langle\mathcal{E}, \mathcal{S}\rangle$ is called a *pre-matching segment* of length $t \in \mathbb{R}^{\geq 0}$ for a $\mu \in \mathbb{S}\langle\mathcal{E}, \mathcal{M}\rangle$ and a binary relation

$B \subseteq \mathbb{S}\langle \mathcal{E}, \mathcal{M}, \mathcal{S}\rangle$ iff the following constraints are satisfied. $t_k - t_0 = t$; for every $h \in [0, k)$ and $t' \in [0, t_{h+1} - t_h]$, $(\mu + t_h - t_0 + t', \nu_h + t') \in B$; and for every $h \in [0, k)$, $\nu_h \xrightarrow{t_{h+1}-t_h,(\perp_q, g_{h+1})} \nu_{h+1}$ for some global transition $(\perp_q, g_{h+1})$ of $\langle \mathcal{E}, \mathcal{S}\rangle$. Similarly, we can also define the pre-matching segments of length $t \in \mathbb{R}^{\geq 0}$ for a $\nu \in \mathbb{S}\langle \mathcal{E}, \mathcal{S}\rangle$ and a $B \subseteq \mathbb{S}\langle \mathcal{E}, \mathcal{M}, \mathcal{S}\rangle$. We now first extend TB-bisimulation to dense-time systems in an environment as follows.

Definition 1. **Timed branching bisimulation in an environment**
Suppose we are given an environment PTA $\mathcal{E}$, a model PTA $\mathcal{M}$, and a specification PTA $\mathcal{S}$. A *(timed branching) bisimulation* between $\mathcal{M}$ and $\mathcal{S}$ in environment $\mathcal{E}$ (*TBE-bisimulation*) is a binary relation $B \subseteq \mathbb{S}\langle \mathcal{E}, \mathcal{M}, \mathcal{S}\rangle$ with the following restrictions on all its element (μ_0, ν_0).

B0: $\mathtt{mode}_{\mathcal{E}}(\mu_0) = \mathtt{mode}_{\mathcal{E}}(\nu_0)$ and for every $x \in X_{\mathcal{E}}(\mu_0(x) = \nu_0(x))$.

B1: For every $t \in \mathbb{R}^{\geq 0}$, global transition (e, f) of $\langle \mathcal{E}, \mathcal{M}\rangle$, and $\mu' \in \mathbb{S}\langle \mathcal{E}, \mathcal{M}\rangle$, if $\mu \xrightarrow{t,(e,f)} \mu'$, then there are a $g \in T_{\mathcal{S}}$, a state $\nu' \in \mathbb{S}\langle \mathcal{E}, \mathcal{S}\rangle$, and a pre-matching segment $(\nu_0, t_0) \ldots (\nu_n, t_n)$ of $\langle \mathcal{E}, \mathcal{S}\rangle$ of length t for μ and B such that g is compatible with f, $\nu_n \xrightarrow{0,(e,g)} \nu'$, and $(\mu', \nu') \in B$.

B2: For every $t \in \mathbb{R}^{\geq 0}$, global transition (e, g) of $\langle \mathcal{E}, \mathcal{S}\rangle$, and $\nu' \in \mathbb{S}\langle \mathcal{E}, \mathcal{S}\rangle$, if $\nu \xrightarrow{t,(e,g)} \nu'$, then there are an $f \in T_{\mathcal{M}}$, a state $\mu' \in \mathbb{S}\langle \mathcal{E}, \mathcal{M}\rangle$, and a pre-matching segment $(\mu_0, t_0) \ldots (\mu_n, t_n)$ of $\langle \mathcal{E}, \mathcal{M}\rangle$ of length t for ν and B such that f is compatible with g, $\mu_n \xrightarrow{0,(e,f)} \mu'$, and $(\mu', \nu') \in B$.

Given a TBE-bisimulation B between $\mathcal{M}$ and $\mathcal{S}$ in $\mathcal{E}$, we denote $\mathcal{M} \overset{B}{\equiv}_{\mathcal{E}} \mathcal{S}$ if for every state $\mu \models I_{\mathcal{E}} \wedge I_{\mathcal{M}}$, there is a $\nu \models I_{\mathcal{E}} \wedge I_{\mathcal{S}}$ with $(\mu, \nu) \in B$; and for every state $\nu \models I_{\mathcal{E}} \wedge I_{\mathcal{S}}$, there is a $\mu \models I_{\mathcal{E}} \wedge I_{\mathcal{M}}$ with $(\mu, \nu) \in B$. If $\exists B(\mathcal{M} \overset{B}{\equiv}_{\mathcal{E}} \mathcal{S})$, we say that $\mathcal{M}$ and $\mathcal{S}$ are *TBE-bisimulation equivalent* in $\mathcal{E}$, in symbols $\mathcal{M} \equiv_{\mathcal{E}} \mathcal{S}$. ■

Given an environment PTA $\mathcal{E}$, we can construct another PTA $\hat{\mathcal{E}}$ that is identical to $\mathcal{E}$ except that all mode names q are replaced by $\hat{q}$ and all variables $x \in L_{\mathcal{E}} \cup X_{\mathcal{E}}$ are replaced by $\hat{x}$. Then we have the following lemma.

Lemma 1. *Suppose we are given an environment PTA $\mathcal{E}$, a model PTA $\mathcal{M}$, and a specification PTA $\mathcal{S}$. $\mathcal{M} \equiv_{\mathcal{E}} \mathcal{S}$ if and only if $\langle \mathcal{E}, \mathcal{M}\rangle \equiv \langle \hat{\mathcal{E}}, \mathcal{S}\rangle$.* ■

4 Non-Zeno Bisimulation in an Environment

However, TBE-bisimulation does not preserve TCTL [1] properties since TCTL formulas are defined with non-Zeno runs. In contrast, TBE-bisimulation does not address this issue. For convenience, we let TCTL(c, L) denote the set of TCTL formulas with timing constants no greater than c and propositions from L. Here we propose the following variation of TBE-bisimulation to patch this gap.

Definition 2. **Non-Zeno bisimulation in an environment** Suppose we are given three PTAs $\mathcal{E}$, $\mathcal{M}$, and $\mathcal{S}$. A *non-Zeno bisimulation* B between $\mathcal{M}$ and $\mathcal{S}$ in $\mathcal{E}$ (*NZE-bisimulation*) is a binary relation $B \subseteq \mathbb{S}\langle \mathcal{E}, \mathcal{M}, \mathcal{S}\rangle$ with the following restrictions on all its elements (μ_0, ν_0).

B0: the same as restriction **B0** in definition 1.

B3: For every $t \in \mathbb{R}^{\geq 0}$, global transition (e, f) of $\langle \mathcal{E}, \mathcal{M} \rangle$, and $\mu' \in \mathbb{S}\langle \mathcal{E}, \mathcal{M} \rangle$, if $\mu_0 \xrightarrow{t,(e,f)} \mu'$ and there is a (non-Zeno) run from μ', then there are a $g \in T_{\mathcal{S}}$, a state $\nu' \in \mathbb{S}\langle \mathcal{E}, \mathcal{S} \rangle$, and a pre-matching segment $(\nu_0, t_0) \ldots (\nu_n, t_n)$ of $\langle \mathcal{E}, \mathcal{S} \rangle$ of length t for μ_0 and B such that g is compatible with f, $\nu_n \xrightarrow{0,(e,g)} \nu'$, and $(\mu', \nu') \in B$.

B4: For every $t \in \mathbb{R}^{\geq 0}$, global transition (e, g) of $\langle \mathcal{E}, \mathcal{S} \rangle$, and $\nu' \in \mathbb{S}\langle \mathcal{E}, \mathcal{S} \rangle$, if $\nu_0 \xrightarrow{t,(e,g)} \nu'$ and there is a (non-Zeno) run from ν', then there are an $f \in T_{\mathcal{M}}$, a state $\mu' \in \mathbb{S}\langle \mathcal{E}, \mathcal{M} \rangle$, and a pre-matching segment $(\mu_0, t_0) \ldots (\mu_n, t_n)$ of $\langle \mathcal{E}, \mathcal{M} \rangle$ of length t for ν_0 and B such that g is compatible with f, $\mu_n \xrightarrow{0,(e,f)} \mu'$, and $(\mu', \nu') \in B$.

Given such a B, if $\mathcal{M} \overset{B}{\equiv}_{\mathcal{E}} \mathcal{S}$, then we say $\mathcal{M}$ and $\mathcal{S}$ are *NZE-bisimulation equivalent* in $\mathcal{E}$, in symbols $\mathcal{M} \overset{NZ}{\equiv}_{\mathcal{E}} \mathcal{S}$. ■

Let $A, \mu \models \phi$ denote that state μ in automaton A satisfies formula ϕ.

Lemma 2. *Let c be the biggest timing constant used in PTAs $\mathcal{E}$, $\mathcal{M}$, and $\mathcal{S}$. Let Q be the mode names in $\mathcal{E}$. Let B be an NZE-bisimulation between $\mathcal{M}$ and $\mathcal{S}$ in $\mathcal{E}$. For any $\mu\nu \in B$ and $\phi \in TCTL(c, Q)$, $\langle \mathcal{E}, \mathcal{M} \rangle, \mu \models \phi$ iff $\langle \mathcal{E}, \mathcal{S} \rangle, \nu \models \phi$.* ■

We can show that TBE-bisimulations are stronger than NZE-bisimulations.

Table 1. Performance data of scalability w.r.t. various bisimulation definitions

benchmarks	versions	m	Traditional		TBE		NZE	
			time	memory	time	memory	time	memory
Fischer's mutual exclusion (m processes)	timed & non-Zeno bisim. eq.	4	>1800s	>8M	103s	425k	101s	424k
		5	N/A		282s	876k	279s	876k
		6			829s	1856k	786s	1856k
	non-Zeno bisim. eq.	4	>1800s	>8M	74.2s	442k	99.9s	436k
		5	N/A		210s	875k	302s	976k
		6			570s	1766k	1136s	3100k
	None	4	>1800s	>8.5M	32.4s	249k	31.9s	249k
		5	N/A		91.6s	478k	90.0s	478k
		6			250s	943k	225s	944k
CSMA/CD (1 bus+ m senders)	timed & non-Zeno bisim. eq.	1	0.236s	102k	0.164s	41k	0.144s	41k
		2	72.9s	1791k	2.90s	211k	2.92s	211k
		3	>1800s	>700M	497s	5362k	506s	5362k
	non-Zeno bisim. eq.	1	0.480s	173k	0.332s	114k	0.788s	76k
		2	>1800s	500M	86.8s	6143k	>1800s	>169M
		3	N/A		>1800s	>75M	N/A	
	None	1	0.144s	103k	0.144s	41k	0.136s	41k
		2	52.9s	3132k	2.87s	222k	2.90s	222k
		3	N/A		48.7s	2613k	50.1s	2613k

data collected on a Pentium 4 1.7GHz with 380MB memory running LINUX;
s: seconds; k: kilobytes of memory in data-structure; M: megabytes of total memory.

Lemma 3. *Given three PTAs* $\mathcal{E}, \mathcal{M}$, *and* $\mathcal{S}$, *if* $\mathcal{M} \equiv_{\mathcal{E}} \mathcal{S}$, *then* $\mathcal{M} \overset{NZ}{\equiv}_{\mathcal{E}} \mathcal{S}$; *but not vice versa.* ■

5 Implementation and Experiments

We have implemented the techniques discussed in this paper in **RED** 7.1, a model-checker for CTAs and parametric safety analysis for LHAs based on CRD (Clock-Restriction Diagram) [8] and HRD (Hybrid-Restriction Diagram) technology [9]. The state-spaces are explored in a symbolic on-the-fly style. We used some parameterized benchmarks [8, 11] from the literature. The performance data is reported in table 1. As can be seen, algorithms for our two new bisimulation definitions perform much better than that for the traditional branching bisimulation.

References

1. Alur, R., Courcoubetis, C., Dill, D.L.: Model Checking for Real-Time Systems. IEEE LICS (1990)
2. Browne, M.C., Clarke, E.M., Grumberg, O.: Characterizing finite Kripke structures in propositional temporal logic. Theoretical Computer Science 59(1,2), 115–131 (1988)
3. Cassez, F., David, A., Fleury, E., Larsen, K.G., Lime, D.: Efficient on-the-fly algorithms for the analysis of timed games. In: Abadi, M., de Alfaro, L. (eds.) CONCUR 2005. LNCS, vol. 3653, pp. 66–80. Springer, Heidelberg (2005)
4. Cerans, K.: Decidability of bisimulation equivalence for parallel timer processes. In: Probst, D.K., von Bochmann, G. (eds.) CAV 1992. LNCS, vol. 663. Springer, Heidelberg (1993)
5. Fisler, K., Vardi, M.Y.: Bisimulation minimization in an automata-theoretic verification framework. In: Gopalakrishnan, G.C., Windley, P. (eds.) FMCAD 1998. LNCS, vol. 1522, pp. 115–132. Springer, Heidelberg (1998)
6. Shaw, A.: Communicating Real-Time State Machines. IEEE Transactions on Software Engineering 18(9) (September 1992)
7. TaşIran, S., Alur, R., Kurshan, R.P., Brayton, R.K.: Verifying abstractions of timed systems. In: Sassone, V., Montanari, U. (eds.) CONCUR 1996. LNCS, vol. 1119. Springer, Heidelberg (1996)
8. Wang, F.: Efficient Verification of Timed Automata with BDD-like Data-Structures. STTT (Software Tools for Technology Transfer) 6(1) (June 2004); Zuck, L.D., Attie, P.C., Cortesi, A., Mukhopadhyay, S. (eds.): VMCAI 2003. LNCS, vol. 2575. Springer, Heidelberg (2003) (special issue)
9. Wang, F.: Symbolic Parametric Safety Analysis of Linear Hybrid Systems with BDD-like Data-Structures. IEEE Transactions on Software Engineering 31(1), 38–51 (2005); a preliminary version is in Alur, R., Peled, D.A. (eds.): CAV 2004. LNCS, vol. 3114. Springer, Heidelberg (2004)
10. Wang, F.: Symbolic Simulation-Checking of Dense-Time Automata. In: Raskin, J.-F., Thiagarajan, P.S. (eds.) FORMATS 2007. LNCS, vol. 4763, pp. 352–368. Springer, Heidelberg (2007)
11. Yovine, S.: Kronos: A Verification Tool for Real-Time Systems. International Journal of Software Tools for Technology Transfer 1(1/2) (October 1997)

Author Index