Efficient Finite Abstraction of Mixed Monotone Systems

Samuel Coogan
scoogan@eecs.berkeley.edu

Murat Arcak
arcak@eecs.berkeley.edu

Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, CA 94720

ABSTRACT

We present an efficient computational procedure for finite abstraction of discrete-time *mixed monotone* systems by considering a rectangular partition of the state space. Mixed monotone systems are decomposable into increasing and decreasing components, and significantly generalize the well known class of monotone systems. We tightly overapproximate the one-step reachable set from a box of initial conditions by computing a *decomposition function* at only two points, regardless of the dimension of the state space. We apply our results to verify the dynamical behavior of a model for insect population dynamics and to synthesize a signaling strategy for a traffic network.

Categories and Subject Descriptors

I.2.8 [**ARTIFICIAL INTELLIGENCE**]: Problem Solving, Control Methods, and Search—*Control theory*; D.2.4 [**SOFTWARE ENGINEERING**]: Software/Program Verification—*Formal methods*

Keywords

Mixed monotone systems, monotone systems, finite state abstractions

1. INTRODUCTION

Complex systems often possess intrinsic structure that significantly simplifies analysis and control. An important class of systems exhibiting such structure is *monotone systems* for which trajectories maintain a partial ordering on states [1, 2]. The notion of monotonicity is applicable to both continuous-time systems [2] and discrete-time systems [3], and has been extended to control systems with inputs in [4].

References [5, 6, 7, 8] have observed that dynamics which are not monotone may nonetheless be decomposable into increasing and decreasing components. Such systems are called *mixed monotone* and significantly generalize the class of monotone systems. Unlike the references above which exploit mixed monotonicity for stability analysis, here we demonstrate that mixed monotonicity enables efficient finite state abstraction.

Increased interest in verification and synthesis of *cyber-physical* systems has motivated symbolic models that abstract the underlying system into a finite set of symbols and transitions between symbols which reflect the dynamics [9, 10, 11]. The main reason for obtaining finite state abstractions is to allow formal verification and synthesis for specifications given in, *e.g.*, temporal logic [12, 13, 14, 15].

In rare cases, exact symbolic models exactly capture the underlying dynamics [16, 17]. In other cases, exact symbolic models are either impossible to obtain or computationally prohibitive, however it is still useful to obtain an abstraction which approximately captures the underlying dynamics [13, 18, 19, 20]. For example, in [21, 22], the authors consider piecewise affine (PWA) systems and construct a finite state abstraction using polyhedral computations.

In this work, we compute finite state abstractions of mixed monotone, discrete-time systems by considering a rectangular partition of the state space. In particular, we show that the reachable set from a box of initial conditions is efficiently overapproximated by evaluating a decomposition function, obtained from the mixed monotone system, at only two points. We accommodate disturbance inputs in the dynamics by suitably generalizing the definition of a mixed monotone system in [6]. Furthermore, we characterize a special class of mixed monotone systems in which the dynamics are componentwise monotone and show that our overapproximation is tight in a particular sense to be made precise. Additionally, we suggest an efficient algorithm for identifying a class of spurious trajectories from the abstraction.

The importance of monotonicity for reachability computation and abstraction has been noted in [23, 24, 25]. In particular, the authors of [23] study discrete-time systems that are monotone with respect to the positive orthant in Euclidean space and show that the reachable set from a box of initial conditions is overapproximated by propagating only the least and greatest points within this box. The present paper studies the much broader class of mixed monotone systems and recovers [23] as a special case.

In Section 2, we introduce the notation. In Section 3, we pose the general problem statement and introduce mixed monotone systems. In Section 4, we present an algorithm for efficiently constructing finite state abstractions of mixed monotone systems. In the case studies of Section 5, we analyze a model for insect population dynamics and synthesize a signal controller for a traffic network.

2. PRELIMINARIES

For $x \in \mathbb{R}^n$, we use superscripts to index the elements of x, *i.e.*, x^i is the ith component of x and $x = (x^1, \ldots, x^n)$, except in the case studies of Section 5 where we use subscripts for clarity. Let $\mathbb{R}_{\geq 0} = \{x \mid x \geq 0\}$ and $\mathbb{R}^n_{\geq 0} = (\mathbb{R}_{\geq 0})^n$. For a set $\mathcal{Z} \subset \mathbb{R}^n$, $\text{int}(\mathcal{Z})$ denotes the interior of $\mathcal{Z}$.

Consider a set $\mathcal{X} \subset \mathbb{R}^n$ along with a *positive cone* $\mathcal{Y}_+ \subset \mathbb{R}^n$ satisfying $\alpha\mathcal{Y}_+ \subset \mathcal{Y}_+$ for all $\alpha \in \mathbb{R}_{\geq 0}$, $\mathcal{Y}_+ + \mathcal{Y}_+ \subset \mathcal{Y}_+$, and $\mathcal{Y}_+ \cap (-\mathcal{Y}_+) = 0$. The positive cone $\mathcal{Y}_+$ induces an order relation $\leq$ on $\mathcal{X}$ defined by: $x \leq y$ if and only if $y - x \in \mathcal{Y}_+$ for $x, y \in \mathcal{X}$. Given $x, y \in \mathcal{X}$ with $x \leq y$, we define the *interval*

$$[x, y] \triangleq \{z \in \mathcal{X} \mid x \leq z \leq y\}. \tag{1}$$

For $\mathcal{Y}_+ = \mathbb{R}^n_{\geq 0}$, $\leq$ denotes coordinate-wise inequality; we distinguish this partial order by $\leq_+$ and generalize it to arbitrary orthants in the following way: Let $\nu = (\nu_1, \ldots, \nu_n)$ with $\nu_i \in \{0, 1\}$ for all i, and define $K_\nu = \{x \in \mathbb{R}^n \mid (-1)^{\nu_i} x^i \geq 0 \ \forall i\}$. K_ν is a cone corresponding to an orthant of $\mathbb{R}^n$, and we denote the induced *orthant order* by $\leq_{K_\nu}$.

For a matrix $M \in \mathbb{R}^{n \times p}$, we additionally interpet $0 \leq_+ M$ to mean M is elementwise nonnegative.

A set $\mathcal{Z} \subset \mathbb{R}^n$ is said to be a *box* if it is the Cartesian product of closed intervals of $\mathbb{R}$, that is, if there exists $a_i, b_i \in \mathbb{R}$ for $i = 1, \ldots n$ such that $a_i \leq b_i$ and $\mathcal{Z} = \prod_{i=1}^n [a_i, b_i]_{\mathbb{R}_{\geq 0}}$ where $[\cdot, \cdot]_{\mathbb{R}_{\geq 0}}$ denotes the usual interval on $\mathbb{R}$.

We let $x^+ = F(x, d)$ describe a discrete-time dynamical system where the state x^+ at the next time step is a function of the current state x and a disturbance input d. We denote the ith coordinate mapping of F by F^i, that is, $(x^i)^+ = F^i(x, d)$ and $F(x, d) = (F^1(x, d), \ldots, F^n(x, d))$.

3. MIXED MONOTONE SYSTEMS

3.1 Problem Statement

We first consider discrete-time dynamical systems of the form

$$x^+ = F(x, d) \tag{2}$$

with state $x \in \mathcal{X} \subset \mathbb{R}^n$, disturbance input $d \in \mathcal{D} \subset \mathbb{R}^p$, and a continuous map $F : \mathcal{X} \times \mathcal{D} \to \mathcal{X}$. We present a technique for efficiently computing a finite state abstraction of (2) when F is *mixed monotone* as defined below. The resulting symbolic model is amenable to standard formal methods techniques to verify desirable properties, as demonstrated in the case study in Section 5.1.

Next, we consider the problem of controlling the switched discrete-time dynamical system

$$x^+ = F_m(x, d) \tag{3}$$

for $m \in \mathcal{M}$ where $\mathcal{M}$ is a finite set of modes and each $F_m : \mathcal{X} \times \mathcal{D} \to \mathcal{X}$ is continuous. For switched systems of the form (3), the control input is the mode m at each time step. When each F_m satisfies a mixed monotonicity property, we propose an efficient algorithm for obtaining a finite state abstraction. As demonstrated in the case study of Section 5.2, this abstraction is amenable to synthesis algorithms to meet complex control objectives expressible in, *e.g.*, *Linear Temporal Logic* (LTL).

3.2 Basic Definitions and Results

For systems of the form (2), we let $\leq_\mathcal{X}$ and $\leq_\mathcal{D}$ denote order relations on $\mathcal{X} \subset \mathbb{R}^n$ and $\mathcal{D} \subset \mathbb{R}^p$, respectively, induced by positive cones. The notation $[\cdot, \cdot]_\mathcal{X}$ (resp. $[\cdot, \cdot]_\mathcal{D}$) denotes an interval with respect to $\leq_\mathcal{X}$ (resp. $\leq_\mathcal{D}$). For systems of the form (3), we wish to allow potentially different order relations on $\mathcal{X}$, and thus consider a set $\{\leq_m\}_{m \in \mathcal{M}}$ of order relations on $\mathcal{X}$ and $\leq_\mathcal{D}$, a fixed order relation on $\mathcal{D}$. The notation $[\cdot, \cdot]_m$ denotes an interval with respect to $\leq_m$. For notational convenience, we assume that the same partial order on $\mathcal{D}$ holds for all modes, however different partial orders on $\mathcal{D}$ for each mode are possible with suitable alterations to the development below.

We begin with the well-known class of monotone dynamical systems:

Definition 1 (Monotonicity). *The system* (2) *is* monotone with respect to $\leq_\mathcal{X}$ and $\leq_\mathcal{D}$, *or simply* monotone, *if*

$$x_1 \leq_\mathcal{X} x_2 \text{ and } d_1 \leq_\mathcal{D} d_2 \implies F(x_1, d_1) \leq_\mathcal{X} F(x_2, d_2). \tag{4}$$

We say that the switched system (3) *is* monotone with respect to $\{\leq_m\}_{m \in \mathcal{M}}$ and $\leq_\mathcal{D}$, *or simply* monotone, *if each mode m is monotone with respect to $\leq_m$ and $\leq_\mathcal{D}$.*

We next provide a significant generalization of Definition 1:

Definition 2 (Mixed monotonicity). *The system* (2) *is said to be* mixed monotone with respect to $\leq_\mathcal{X}$ and $\leq_\mathcal{D}$, *or simply* mixed monotone *[6], if there exists a function $f : \mathcal{X} \times \mathcal{D} \times \mathcal{X} \times \mathcal{D} \to \mathcal{X}$ satisfying:*

C1) $\forall x \in \mathcal{X}, \ \forall d \in \mathcal{D}$: $F(x, d) = f(x, d, x, d)$

C2) $\forall x_1, x_2, y \in \mathcal{X}, \ \forall d_1, d_2, e \in \mathcal{D}$: $x_1 \leq_\mathcal{X} x_2$ *and* $d_1 \leq_\mathcal{D} d_2$ *implies* $f(x_1, d_1, y, e) \leq_\mathcal{X} f(x_2, d_2, y, e)$

C3) $\forall x, y_1, y_2 \in \mathcal{X}, \ \forall d, e_1, e_2 \in \mathcal{D}$: $y_1 \leq_\mathcal{X} y_2$ *and* $e_1 \leq_\mathcal{D} e_2$ *implies* $f(x, d, y_2, e_2) \leq_\mathcal{X} f(x, d, y_1, e_1)$.

We say that the switched system (3) *is* mixed monotone with respect to $\{\leq_m\}_{m \in \mathcal{M}}$ and $\leq_\mathcal{D}$, *or simply* mixed monotone, *if each mode $x^+ = F_m(x, d)$ is mixed monotone with respect to $\leq_m$.*

The function f is nondecreasing in the first pair of variables and nonincreasing in the second pair of variables, and is henceforth called a *decomposition function*:

Definition 3 (Decomposition function). *A function f satisfying C1)–C3) above is a* decomposition function *for $F(x, d)$.*

Clearly every monotone system is mixed monotone with $f(x, d, y, e) \triangleq F(x, d)$. In the case of a switched system (3), we denote by f_m a corresponding decomposition function for each mode $m \in \mathcal{M}$.

Example 1. *Consider the system*

$$x^+ = G(x, d) - H(x, d) \tag{5}$$

for $x \in \mathcal{X} \subset \mathbb{R}^n$, $d \in \mathcal{D} \subset \mathbb{R}^p$, and $G, H : \mathcal{X} \times \mathcal{D} \to \mathcal{X}$ such that $x^+ = G(x, d)$ and $x^+ = H(x, d)$ are monotone systems for $\leq_\mathcal{X} = \leq_+$ and $\leq_\mathcal{D} = \leq_+$. Then (5) *is mixed monotone for $\leq_\mathcal{X} = \leq_+$ and $\leq_\mathcal{D} = \leq_+$ and $f(x, d, y, e) \triangleq G(x, d) - H(y, e)$ is a decomposition function.*

EXAMPLE 2. *Consider the system*

$$x^+ = A(x,d)x + B(x,d)d =: F(x,d) \tag{6}$$

for $x \in \mathcal{X} \subset \mathbb{R}^n_{\geq 0}$, $d \in \mathcal{D} \subset \mathbb{R}^p_{\geq 0}$, *such that:*

- $0 \leq_+ A(x,d)$ *and* $0 \leq_+ B(x,d)$ *for all* $x \in \mathcal{X}$ *for all* $d \in \mathcal{D}$,
- $x_1 \leq_+ x_2$ *and* $d_1 \leq_+ d_2 \implies A(x_2,d_2) \leq_+ A(x_1,d_1)$ *and* $B(x_2,d_2) \leq_+ B(x_1,d_1)$.

Equations of the form (6) *arise in the study of population dynamics, [26]. Taking* $f(x,d,y,e) = A(y,e)x + B(y,e)d$, *system* (6) *is mixed monotone for* $\leq_\mathcal{X}=\leq_+$ *and* $\leq_\mathcal{D}=\leq_+$.

We now characterize a special class of mixed monotone systems in terms of the sign of the entries in $\partial F/\partial x$ and $\partial F/\partial d$, the Jacobians of F with respect to x and d.

PROPOSITION 1. *Consider the system* (2) *where* $x \in \mathcal{X} \subset \mathbb{R}^n$, $d \in \mathcal{D} \subset \mathbb{R}^p$, $\mathcal{X}$ *and* $\mathcal{D}$ *are boxes, and* F *is continuously differentiable. If for all* $i \in \{1,\dots,n\}$,

$$\forall j \in \{1,\dots,n\}\ \exists s_j \in \{0,1\} : (-1)^{s_j}\frac{\partial F^i}{\partial x^j}(x,d) \geq 0\ \forall x,d \tag{7}$$

and

$$\forall j \in \{1,\dots,p\}\ \exists \sigma_j \in \{0,1\} : (-1)^{\sigma_j}\frac{\partial F^i}{\partial d^j}(x,d) \geq 0\ \forall x,d \tag{8}$$

then (2) *is mixed monotone with respect to any orthant order on* $\mathcal{X}$ *and* $\mathcal{D}$.

PROOF. Let $\nu \in \{0,1\}^n$ and $\mu \in \{0,1\}^p$ characterize arbitrary orthant orders $\leq_{K_\nu}$ and $\leq_{K_\mu}$ on $\mathcal{X}$ and $\mathcal{D}$, respectively. Define

$$f^i(x,d,y,e) \triangleq F^i(z^i,w^i) \tag{9}$$

where $z^i = (z^{i,1},\dots,z^{i,n})$, $w^i = (w^{i,1},\dots,w^{i,p})$, and

$$z^{i,j} \triangleq \begin{cases} x^j & \text{if } (-1)^{\nu_i+\nu_j}\partial F^i/\partial x^j \geq 0\ \forall x \in \mathcal{X}, d \in \mathcal{D} \\ y^j & \text{if } (-1)^{\nu_i+\nu_j}\partial F^i/\partial x^j \leq 0\ \forall x \in \mathcal{X}, d \in \mathcal{D} \end{cases} \tag{10}$$

$$w^{i,j} \triangleq \begin{cases} d^j & \text{if } (-1)^{\nu_i+\mu_j}\partial F^i/\partial d^j \geq 0\ \forall x \in \mathcal{X}, d \in \mathcal{D} \\ e^j & \text{if } (-1)^{\nu_i+\mu_j}\partial F^i/\partial d^j \leq 0\ \forall x \in \mathcal{X}, d \in \mathcal{D}. \end{cases} \tag{11}$$

If $\partial F^i/\partial x^j = 0\ \forall x,d$ for some i,j, then the assignment to $z^{i,j}$ is arbitrary, likewise for $w^{i,j}$ if $\partial F^i/\partial d^j = 0\ \forall x,d$ for some i,j. Let $f(x,d,y,e) = (f^1(x,d,y,e),\dots,f^n(x,d,y,e))$. Clearly $f(x,d,x,d) = F(x,d)$, and a straightforward modification of the well-known *Kamke* conditions for monotonicity [2, Section 3.1] proves that f satisfies the remaining conditions of Definition 2. □

Proposition 1 states that if the partial derivatives of F are *sign stable* over $\mathcal{X}\times\mathcal{D}$, then (2) is mixed monotone with respect to any orthant order on $\mathcal{X}$ and $\mathcal{D}$. The special class characterized in Proposition 1 plays an important role in the case study of Section 5.2; see [27] for a similar characterization that excludes disturbance inputs.

EXAMPLE 3. *Let* $\mathcal{X} = \mathbb{R}^2_{\geq 0}$, $\mathcal{D} = \mathbb{R}^2_{\geq 0}$, *and consider the system*

$$\begin{aligned} x^+ = F(x,d) &= (F^1(x,d), F^2(x,d)) & (12)\\ &= (5x_1 - x_2^3 + 5d_1^2, x_1^2 + 3x_2x_1 - 6d_1d_2) & (13) \end{aligned}$$

where $x = (x_1,x_2) \in \mathbb{R}^2_{\geq 0}$ *and* $d = (d_1,d_2) \in \mathbb{R}^2_{\geq 0}$ *(we momentarily abandon our superscript convention for notational convenience). For all* $x \in \mathcal{X}$, $d \in \mathcal{D}$,

$$\partial F^1/\partial x_1 = 5 \geq 0 \qquad \partial F^1/\partial x_2 = -3x_2^2 \leq 0 \tag{14}$$

$$\partial F^2/\partial x_1 = 2x_1 + 3x_2 \geq 0 \qquad \partial F^2/\partial x_2 = 3x_1 \geq 0 \tag{15}$$

$$\partial F^1/\partial d_1 = 10d_1 \geq 0 \qquad \partial F^1/\partial d_2 = 0 \tag{16}$$

$$\partial F^2/\partial d_1 = -6d_2 \leq 0 \qquad \partial F^2/\partial d_2 = -6d_1 \leq 0. \tag{17}$$

Thus, the system is mixed monotone by Proposition 1. Taking $\leq_\mathcal{X}=\leq_+$ *and* $\leq_\mathcal{D}=\leq_+$, *we have that*

$$f(x,d,y,e) = (5x_1 - y_2^3 + 5d_1^2, x_1^2 + 3x_2x_1 - 6e_1e_2) \tag{18}$$

is a decomposition function where $y = (y_1,y_2)$, $e = (e_1,e_2)$.

We remark that, while Proposition 1 assumed that F is continuously differentiable, the results in fact hold if F is continuous and *piecewise differentiable*, and thus nondifferentiable on a set of measure zero as in the case study of Section 5.2.

3.3 Reachable Set Computation

In this section, we show that an overapproximation of the reachable set from a box of initial states is efficiently computed by evaluating the decomposition function at only two points, regardless of the state space dimension. In the next section, we use this result to obtain finite state abstractions of mixed monotone systems.

We begin with the following key theorem:

THEOREM 1. *Let* (2) *be a mixed monotone system with decomposition function* $f(x,d,y,e)$. *Given* $x_1,x_2 \in \mathcal{X}$ *and* $d_1,d_2 \in \mathcal{D}$ *with* $x_1 \leq_\mathcal{X} x_2$ *and* $d_1 \leq_\mathcal{D} d_2$,

$$\begin{aligned} f(x_1,d_1,x_2,d_2) \leq_\mathcal{X} F(x,d) \leq_\mathcal{X} f(x_2,d_2,x_1,d_1)\\ \forall\ x \in [x_1,x_2]_\mathcal{X}\ \forall\ d \in [d_1,d_2]_\mathcal{D}. \end{aligned} \tag{19}$$

PROOF. Consider x,d,y,e satisfying

$$x_1 \leq_\mathcal{X} x \text{ and } d_1 \leq_\mathcal{X} d, \text{ and} \tag{20}$$

$$y \leq_\mathcal{X} x_2 \text{ and } e \leq_\mathcal{X} d_2. \tag{21}$$

It follows that

$$f(x_1,d_1,x_2,d_2) \leq_\mathcal{X} f(x,d,y,e), \qquad \textit{and} \tag{22}$$

$$f(y,e,x,d) \leq_\mathcal{X} f(x_2,d_2,x_1,d_1). \tag{23}$$

Restricting to the set $\{(x,d,y,e) \mid x = y \text{ and } d = e\}$, we obtain

$$\begin{aligned} f(x_1,d_1,x_2,d_2) &\leq_\mathcal{X} f(x,d,x,d) = F(x,d)\\ &\leq_\mathcal{X} f(x_2,d_2,x_1,d_1). \end{aligned} \tag{24}$$

□

The analogous result for monotone systems is:

COROLLARY 1. *Given* $x_1,x_2 \in \mathcal{X}$ *and* $d_1,d_2 \in \mathcal{D}$ *with* $x_1 \leq_\mathcal{X} x_2$ *and* $d_1 \leq_\mathcal{D} d_2$. *If system* (2) *is monotone, then*

$$\begin{aligned} F(x_1,d_1) \leq_\mathcal{X} F(x,d) \leq_\mathcal{X} F(x_2,d_2)\\ \forall\ x \in [x_1,x_2]_\mathcal{X}\ \forall\ d \in [d_1,d_2]_\mathcal{D}. \end{aligned} \tag{25}$$

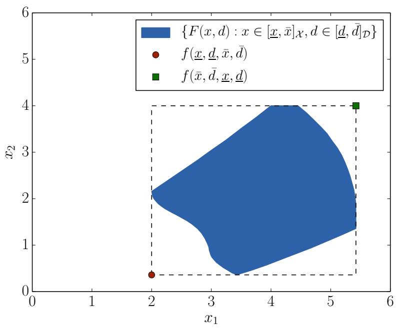

Figure 1: The mixed monotone system in Examples 3 and 4. This system satisfies the conditions of Theorem 1, thus we bound $F(x,d)$ when x and d are confined to lie within a given rectangle by evaluating the decomposition function at two points, and the bounding is tight. This example readily generalizes to higher dimensions.

The result in [23] is a special case of Corollary 1 restricted to systems with no disturbance input and $\leq_{\mathcal{X}}=\leq_{+}$.

For $\mathcal{X}' \subseteq \mathcal{X}$ and $\mathcal{D}' \subseteq \mathcal{D}$, we define the shorthand notation

$$F(\mathcal{X}', \mathcal{D}') \triangleq \{F(x,d) \mid x \in \mathcal{X}' \text{ and } d \in \mathcal{D}'\}. \tag{26}$$

Then we respectively write (19) and (25) as

$$F([x_1, x_2]_{\mathcal{X}}, [d_1, d_2]_{\mathcal{D}}) \subseteq [f(x_1, d_1, x_2, d_2), f(x_2, d_2, x_1, d_1)]_{\mathcal{X}} \tag{27}$$

and

$$F([x_1, x_2]_{\mathcal{X}}, [d_1, d_2]_{\mathcal{D}}) \subseteq [F(x_1, d_1), F(x_2, d_2)]_{\mathcal{X}}. \tag{28}$$

Definition 4. *The set $F(\mathcal{X}', \mathcal{D}')$ given in (26) is the* one-step reachable set *from $\mathcal{X}'$ and $\mathcal{D}'$.*

Example 4. *Consider again Example 3 and let $\underline{x} = (0.6, 0.3)$, $\bar{x} = (1,1)$, $\underline{d} = (0,0)$, $\bar{d} = (0.3, 0.3)$. From Theorem 1, it follows that*

$$\begin{aligned} F([\underline{x}, \bar{x}]_{\mathcal{X}}, [\underline{d}, \bar{d}]_{\mathcal{D}}) &\subset [f(\underline{x}, \underline{d}, \bar{x}, \bar{d}), f(\bar{x}, \bar{d}, \underline{x}, \underline{d})]_{\mathcal{X}} \\ &= [(2, 0.36), (5.423, 4)]_{\mathcal{X}}. \end{aligned} \tag{29}$$

Figure 1 shows the set $\{F(x,d) \mid x \in [\underline{x}, \bar{x}]_{\mathcal{X}}, d \in [\underline{d}, \bar{d}]_{\mathcal{D}}\}$ as a shaded region, and plots $f(\underline{x}, \underline{d}, \bar{x}, \bar{d})$ $f(\bar{x}, \bar{d}, \underline{x}, \underline{d})$ as two corners of a box that bounds this set.

For monotone systems, Corollary 1 provides tight bounds since the upper and lower bounds are achieved. For mixed monotone systems satisfying (7)–(8) of Proposition 1, the bounds given in Theorem 1 are also tight as suggested in Figure 1 for Example 4. We make this precise in the following proposition, which follows immediately from the definition in (9):

Proposition 2. *Suppose $\leq_{\mathcal{X}}=\leq_{K_\nu}$ and $\leq_{\mathcal{D}}=\leq_{K_\mu}$ for some orthants K_ν and K_μ. If (2) is mixed monotone by (7)–(8) of Proposition 1, and f is the decomposition function as defined in (9)–(11), then for all $i \in \{1, \dots, n\}$ there exists $\underline{z}^i, \bar{z}^i \in [x_1, x_2]_{\mathcal{X}}$ and $\underline{w}^i, \bar{w}^i \in [d_1, d_2]_{\mathcal{D}}$ such that*

$$f^i(x_1, d_1, x_2, d_2) = F^i(\underline{z}^i, \underline{w}^i), \quad \text{and} \tag{30}$$

$$f^i(x_2, d_2, x_1, d_1) = F^i(\bar{z}^i, \bar{w}^i). \tag{31}$$

In particular, $\underline{z}^i$ as in (10) with $x = x_1$ and $y = x_2$, and $\underline{w}^i$ as in (11) with $d = d_1$ and $e = d_2$ satisfies (30). A symmetric results holds for (31) after interchanging x_1, x_2 and d_1, d_2.

4. ABSTRACTION OF MIXED MONOTONE SYSTEMS

We have seen that for mixed monotone systems, an over-approximation of the one-step reachable set from the set $[x_1, x_2]_{\mathcal{X}}$ under a disturbance input from the set $[d_1, d_2]_{\mathcal{D}}$ can be computed by evaluating the decomposition function f at only two particular points. We now exploit Theorem 1 and Corollary 1 and present an efficient algorithm for computing a *symbolic model*, or *finite state abstraction* of a mixed monotone system. For systems of the form (2), we wish to *verify* that a certain property, usually given in a temporal logic, holds under all possible disturbance inputs. For switched systems of the form (3), we wish to *synthesize* a mode selection policy such that the resulting system satisfies a given property.

4.1 Finite State Abstraction

Now we introduce a partition of the domain $\mathcal{X}$ by intervals and construct a finite state abstraction from the partition. We discuss systems of the form (3), since (2) is a special case.

Assume system (3) is mixed monotone with respect to $\{\leq_m\}_{m \in \mathcal{M}}$ and $\leq_{\mathcal{D}}$. Furthermore, assume $\mathcal{D}$ is representable as the union of intervals:

$$\mathcal{D} = \bigcup_{\ell=1}^{L} \mathcal{D}^\ell \tag{32}$$

where $\mathcal{D}^\ell \triangleq [d_1^\ell, d_2^\ell]_{\mathcal{D}}$ for $d_1^\ell \leq_{\mathcal{D}} d_2^\ell$.

Definition 5 (Interval Partition). *The collection $\{\mathcal{I}_q\}_{q \in \mathcal{Q}}$ for finite set $\mathcal{Q}$ with $\mathcal{I}_q \subseteq \mathcal{X}$ for all $q \in \mathcal{Q}$ is an* interval partition *of $\mathcal{X}$ if:*

1. *For all $m \in \mathcal{M}$ and for all $q \in \mathcal{Q}$, there exists $x_1^{q,m}, x_2^{q,m} \in \mathcal{X}$ satisfying $x_1^{q,m} \leq_m x_2^{q,m}$ and $\mathcal{I}_q = [x_1^{q,m}, x_2^{q,m}]_m$,*
2. *$\bigcup_{q \in \mathcal{Q}} \mathcal{I}_q = \mathcal{X}$,*
3. *$int(\mathcal{I}_q) \cap int(\mathcal{I}_{q'}) = \emptyset$ for all $q, q' \in \mathcal{Q}, q \neq q'$.*

In other words, $\{\mathcal{I}_q\}_{q \in \mathcal{Q}}$ is an *interval partition* of $\mathcal{X}$ if the sets $\mathcal{I}_q$, $q \in \mathcal{Q}$ partition $\mathcal{X}$ and each $\mathcal{I}_q$ is representable as an interval of $\mathcal{X}$ with respect to each order $\leq_m$. In defining a partition, we ignore the set of measure zero where intervals overlap for notational convenience, as is done in, *e.g.*, [21]. However, as noted in [21] and [22], if the dynamics are such that trajectories remain within the boundaries after a certain time, one should account for such sets.

For example, if each $\leq_m$ is an orthant order, then a partition $\{\mathcal{I}_q\}_{q \in \mathcal{Q}}$ with each $\mathcal{I}_q$ a box constitutes an interval partition of $\mathcal{X} \subset \mathbb{R}^n$. For this special case, we further call the partition a *gridded partition* if for each $i \in \{1, \dots, n\}$ there exists $N_i > 0$ and $\{\xi_{i,1}, \dots, \xi_{i,N_i+1}\}$ such that $\mathcal{Q} = \prod_{i=1}^n \{1, \dots, N_i\}$ and for each $q = (\iota_1, \dots, \iota_n) \in \mathcal{Q}$, we have $\mathcal{I}_q = \prod_{i=1}^n [\xi_{i,\iota_i}, \xi_{i,\iota_i+1}]_{\mathbb{R}_{\geq 0}}$. Figure 2 shows schematic depictions of two interval partitions, one of which is a gridded partition.

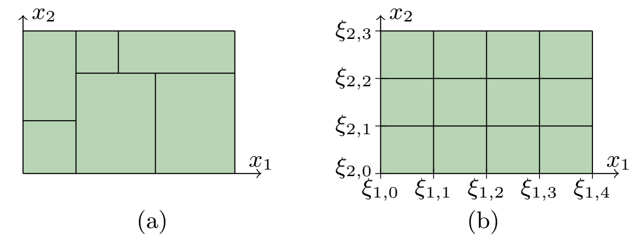

Figure 2: Stylized depiction of (a) an interval partition, and (b) a gridded partition.

When clear from context, we refer to the index set $\mathcal{Q}$ itself as an interval partition with the associated notation as above. From such a partition, we readily construct a finite state abstraction of the resulting dynamics.

Consider a map $\delta : \mathcal{Q} \times \mathcal{M} \to 2^{\mathcal{Q}}$ that satisfies the following property:

$$\begin{aligned} &\text{If } \exists x \in \mathcal{I}_q,\ \exists d \in \mathcal{D} \ \text{ such that } F_m(x, d) \in \mathcal{I}_{q'} \\ &\text{Then } q' \in \delta(q, m). \end{aligned} \tag{33}$$

The map δ includes a transition from q to q' whenever it is possible for the state x to transition from the interval $\mathcal{I}_q$ to $\mathcal{I}_{q'}$ (although δ may also include additional transitions).

Definition 6 (Interval finite state abstraction). *An* interval finite state abstraction *or simply* abstraction *of system* (3) *is a tuple $\mathcal{T} = (\mathcal{Q}, \mathcal{M}, \delta)$ where $\mathcal{Q}$ is an interval partition of $\mathcal{X}$ and δ satisfies* (33). *We call δ a* transition function *and say $q' \in \mathcal{Q}$ is a* successor *of q in mode m if $q' \in \delta(q, m)$.*

$\mathcal{T}$ is a nondeterministic transition system, *i.e.*, $\delta(q, m)$ is, in general, not a singleton set. The nondeterminism arises because $\mathcal{T}$ abstracts an entire set of states into one state or *symbol*, and the transitions account for all possible states in the symbol as well as the disturbance. Nonetheless, $\mathcal{T}$ is a transition system that overapproximates the dynamics (3), that is, for every trajectory $x[t]$ satisfying $x[t+1] = F_{m[t]}(x[t], d[t])$ such that $m[t] \in \mathcal{M}$ and $d[t] \in \mathcal{D}$ for all t, there exists $q[t]$ such that $x[t] \in \mathcal{I}_{q[t]}$ and $q[t+1] \in \delta(q[t], m[t])$ for all t.

Computing a transition function δ that is useful in practice is a serious difficulty for standard abstraction approaches. Many existing results apply only to linear or piecewise linear systems, and even in this case, scale poorly with the state space. For example, the polytope-based computations suggested in [21] require computing F_m at a number of points that scales exponentially with the dimension of the state space and disturbance space. By exploiting the mixed monotonicity properties of system (3), we propose an efficient method for computing an abstraction that requires evaluating f_m at only two points for each $q \in Q$ and $m \in \mathcal{M}$.

Theorem 2. *Consider the mixed monotone system* (3) *with interval partition $\mathcal{Q}$. Let $\delta : \mathcal{Q} \times \mathcal{M} \to 2^{\mathcal{Q}}$ be given by $q' \in \delta(q, m)$ if and only if*

$$\begin{aligned} &\exists \ell : [f_m(x_1^{q,m}, d_1^\ell, x_2^{q,m}, d_2^\ell), f_m(x_2^{q,m}, d_2^\ell, x_1^{q,m}, d_1^\ell)]_{\mathcal{X}} \\ &\cap [x_1^{q',m}, x_2^{q',m}]_{\mathcal{X}} \neq \emptyset. \end{aligned} \tag{34}$$

Then $\mathcal{T} = (\mathcal{Q}, \mathcal{M}, \delta)$ is a finite state abstraction of (3).

function FiniteStateAbtraction(*system*, $\mathcal{D}$, $\mathcal{Q}$) **returns** $\mathcal{T}$
  inputs: *system*, a mixed monotone system (3) with domain $\mathcal{X}$, modes $\mathcal{M}$ and decomposition functions $\{f_m\}_{m \in \mathcal{M}}$
    $\mathcal{D}$, the disturbance set $\mathcal{D} = \cup_{\ell=1}^L \mathcal{D}^\ell$ with $\mathcal{D}^\ell \triangleq [d_1^\ell, d_2^\ell]_{\mathcal{D}}$ for $d_1^\ell \leq_{\mathcal{D}} d_2^\ell$
    $\mathcal{Q}$, an interval partition $\mathcal{X}$
  for each $m \in \mathcal{M}$ **do**
    for each $q \in \mathcal{Q}$ **do**
      $\delta(q, m) := \emptyset$
      for $\ell := 1$ **to** L **do**
        $y_1 := f_m(x_1^{q,m}, d_1^\ell, x_2^{q,m}, d_2^\ell)$
        $y_2 := f_m(x_2^{q,m}, d_2^\ell, x_1^{q,m}, d_1^\ell)$
        $\mathcal{Q}' :=$ ComputeSuccessors$(y_1, y_2, \mathcal{Q})$
        $\delta(q, m) := \delta(q, m) \cup \mathcal{Q}'$
      end for
    end for
  end for
  return $\mathcal{T} := (\mathcal{Q}, \mathcal{M}, \delta)$ ▷ abstraction of (3)
end function

Algorithm 1: Algorithm for computing an interval finite state abstraction of (3)**.**

Proof. Consider $x \in \mathcal{I}_q$ and $d \in \mathcal{D}$ such that $x' = F_m(x, d) \in \mathcal{I}_{q'} = [x_1^{q',m}, x_2^{q',m}]_m$. Let $\ell \in \{1, \dots, L\}$ be such that $d \in \mathcal{D}^\ell$. From Theorem 1, it holds that also

$$x' \in [f_m(x_1^{q,m}, d_1^\ell, x_2^{q,m}, d_2^\ell), f_m(x_2^{q,m}, d_2^\ell, x_1^{q,m}, d_1^\ell)]_m,$$

which implies $q' \in \delta(q, m)$, thus δ satisfies (33). ☐

Corollary 2. *Consider monotone system* (3) *with interval partition $\mathcal{Q}$. Let $\delta : \mathcal{Q} \times \mathcal{M} \to 2^{\mathcal{Q}}$ be given by $q' \in \delta(q, m)$ if and only if*

$$\exists \ell : [F_m(x_1^{q,m}, d_1^\ell), F_m(x_2^{q,m}, d_2^\ell)] \cap [x_1^{q',m}, x_2^{q',m}] \neq \emptyset. \tag{35}$$

Then $\mathcal{T} = (\mathcal{Q}, \mathcal{M}, \delta)$ is a finite state abstraction of (3).

We summarize the algorithm implied by Theorem 2 in Algorithm 1. For systems of the form (2), we interpret $\mathcal{M}$ as a singleton and proceed as above. We then notationally omit $\mathcal{M}$ and instead write $\mathcal{T} = (\mathcal{Q}, \delta)$, and $\delta(q) \subset \mathcal{Q}$.

4.2 Computing Successor States

Theorem 2 and Corollary 2 provided a method for overapproximating the one-step reachable set of an interval. How do we identify the successor states from this overapproximation? Lemma 1 below provides an efficient method for determining if two intervals overlap. With this lemma, we establish a universal algorithm for computing the set of one-step reachable intervals in Figure 2.

Lemma 1. *Consider $[\alpha_1, \beta_1]_{\mathcal{X}}$ and $[\alpha_2, \beta_2]_{\mathcal{X}}$ for $\alpha_1, \beta_1 \in \mathcal{X}$ and $\alpha_2, \beta_2 \in \mathcal{X}$. Then $[\alpha_1, \beta_1]_{\mathcal{X}} \cap [\alpha_2, \beta_2]_{\mathcal{X}} \neq \emptyset$ implies $\alpha_1 \leq_{\mathcal{X}} \beta_2$ and $\alpha_2 \leq_{\mathcal{X}} \beta_1$.*

Proof. Choosing $x \in [\alpha_1, \beta_1]_{\mathcal{X}} \cap [\alpha_2, \beta_2]_{\mathcal{X}}$, the lemma follows from transitivity of $\leq_{\mathcal{X}}$. ☐

For special types of partitions, however, more efficient methods exist for computing the successor states. In particular, when each $\leq_m$ is an orthant order and $\mathcal{Q}$ is a gridded partition, computing successor states is accomplished

function ComputeSuccessors(y_1, y_2, $\mathcal{Q}$) **returns** $\mathcal{Q}'$
  inputs: y_1 and y_2, points in domain $\mathcal{X} \subset \mathbb{R}^n$
    $\mathcal{Q}$, an interval partition of $\mathcal{X}$
  initialize: $\mathcal{Q}' = \emptyset$
  for each $q' \in \mathcal{Q}$ **do**
    if $(y_1 \leq_m x_2^{q',m}) \wedge (x_1^{q',m} \leq_m y_2)$ **then**
      $\mathcal{Q}' := \mathcal{Q}' \cup \{q'\}$
    end if
  end for
  return $\mathcal{Q}'$ ▷ successor states from $[y_1, y_2]_m$
end function

Algorithm 2: A universal algorithm for overapproximating successor states.

function ComputeSuccessors(y_1, y_2, $\mathcal{Q}$) **returns** $\mathcal{Q}'$
  inputs: y_1 and y_2, points in domain $\mathcal{X} \subset \mathbb{R}^n$
    where $y_j = (y_j^1, \dots, y_j^n)$ for $j = 1, 2$
    $\mathcal{Q}$, a grid interval partition of $\mathcal{X}$, *i.e.*,
    $\mathcal{Q} = \prod_{i=1}^n \{1, \dots, N_i\}$ and
    $\mathcal{I}_q = \prod_{i=1}^n [\xi_{i,\iota_i}, \xi_{i,\iota_i+1}]_{\mathbb{R}_{\geq 0}}$ for each
    $q = (\iota_1, \dots, \iota_n) \in \mathcal{Q}$
  for $i := 1$ **to** n **do**
    if $\min\{y_1^i, y_2^i\} \leq \xi_{i,1}$ **then** $\underline{\iota}_i := 1$ **else**
      $\underline{\iota}_i := \arg\max_{\iota \in \{1,\dots,N_i\}}$ s.t. $\xi_{i,\iota} \leq \min\{y_1^i, y_2^i\}$
    if $\xi_{i,N_i+1} \leq \max\{y_1^i, y_2^i\}$ **then** $\bar{\iota}_i := N_i$ **else**
      $\bar{\iota}_i := \arg\min_{\iota \in \{1,\dots,N_i\}}$ s.t. $\xi_{i,\iota+1} \geq \max\{y_1^i, y_2^i\}$
  end for
  return $\mathcal{Q}' := \{(\iota_1, \dots, \iota_n) \mid \iota_i \in \{\underline{\iota}_i, \dots, \bar{\iota}_i\}\ \forall i\}$
end function

Algorithm 3: An algorithm to identify successor states when $\mathcal{Q}$ is a gridded partition of $\mathbb{R}^n$.

by considering each coordinate separately, as in Algorithm 3. This algorithm scales linearly with $\sum_{i=1}^n N_i$. When all N_i are approximately the same, the algorithm scales approximately linearly with n.

4.3 Spurious Self-Loops

An abstraction may produce *spurious* trajectories that do not correspond to any trajectories of (3). While such spurious trajectories are often unavoidable, we can identify and ameliorate the effect of a particular type of spurious trajectory that are generated from "self-loops" of the finite state abstraction. $\mathcal{T}$ contains a *self-loop* at state $q^* \in \mathcal{Q}$ for modes $\mathcal{M}' \subseteq \mathcal{M}$ if $q^* \in \delta(q^*, m)$ for all $m \in \mathcal{M}'$. A self-loop implies that under any control action satisfying $\sigma[t] \in \mathcal{M}'$ for all t, the trajectory $q[t] = q^*$ for all t is possible in $\mathcal{T}$. If there is no corresponding trajectory in the original system (3) for any such choice of $\sigma[t]$, the state and input set pair $(q^*, \mathcal{M}')$ is said to be *stuttering*. A similar definition of stuttering inputs is given [21]. For systems of the form (2), we instead say q^* is *stuttering* if the above holds with $\mathcal{M}'$ interpreted to be the singleton set corresponding to F.

In verification problems where the dynamics have the form (2), it is sometimes possible to simply remove stuttering transitions. In particular, this is possible if the condition to be verified belongs to the fragment of LTL without the "next" operator [28]. In other cases, knowledge of stuttering inputs leads to less conservative control strategies; see [21] for a detailed discussion.

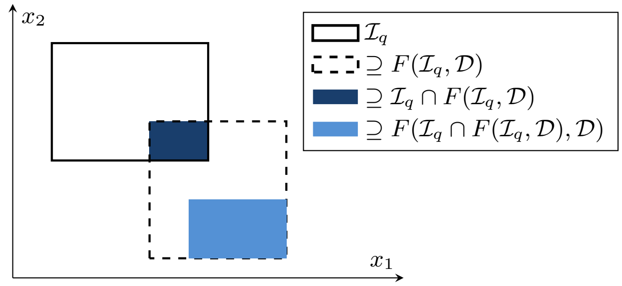

Figure 3: Finding stuttering inputs. The solid outline denotes $\mathcal{I}_q$, and the dashed outline denotes the overapproximation of the one-step reachable set from $\mathcal{I}_q$. By overapproximating the one-step reachable set (lightly shaded region) of the intersection (darkly shaded region), we determine that q is stuttering because this region no longer intersects $\mathcal{I}_q$.

A sufficient condition for determining if $(q, \mathcal{M}')$ is stuttering is to compute a sequence of one-step reachable sets that eventually do not intersect $\mathcal{I}_q$. As an illustration, consider system (2) with the standard order $\leq_+$ on $\mathcal{X} \subset \mathbb{R}^2$. Figure 3 shows that the overapproximation of $F(\mathcal{I}_q, \mathcal{D})$ intersects $\mathcal{I}_q$, and thus $q \in \delta(q)$. We then overapproximate the one-step reachable set from $\mathcal{I}_q \cap F(\mathcal{I}_q, \mathcal{D})$, which no longer intersects $\mathcal{I}_q$, and thus we conclude that q is stuttering because no trajectory of (2) can remain within $\mathcal{I}_q$ for all time.

We generalize this idea and provide Algorithm 4 for determining if $(q, \mathcal{M}')$ is stuttering. The algorithm requires a function getNewInt which returns a set of points $\{\zeta_1^m, \zeta_2^m\}_{m \in \mathcal{M}'}$ such that for each $m \in \mathcal{M}'$,

$$[\zeta_1^m, \zeta_2^m]_m \supseteq [y_1^{m',\ell}, y_2^{m',\ell}]_{m'} \cap \mathcal{I}_q \quad \forall m' \in \mathcal{M}' \quad \forall \ell \in \{1, \dots, L\}. \tag{36}$$

These points are used in the next iteration when computing the one-step reachable set. In Figure 3, these points correspond to the points defining the darkly shaded interval. As suggested by this example, implementing getNewInt for Euclidean spaces with orthant orders can be done coordinate-wise.

4.4 Computational Requirements

We now address the computational requirements of the proposed algorithms. Determining $\delta(q, m)$ requires first evaluating the decomposition function f_m at $2L$ points where L is the number of boxes constituting the disturbance set $\mathcal{D}$. For each $\ell = 1, \dots L$, the corresponding pair of evaluations of f_m is then used to determine successor states representing an overapproximation of the reachable set from q. In Algorithm 2, this requires $2|\mathcal{Q}|$ order comparisons of vectors in $\mathbb{R}^n$, and each comparison scales linearly with n. For gridded partitions, determining successor states requires $\sum_{i=1}^n N_i$ scalar order comparisons as seen in Algorithm 3.

Thus, computing δ scales linearly with $|\mathcal{M}|$ and linearly with L. Using Algorithm 2, the computation further scales quadratically with $|\mathcal{Q}|$ and linearly with n, and using Algorithm 3, it scales linearly with $|\mathcal{Q}|$ and linearly with $\sum_{i=1}^n N_i$. In contrast, computing successor states from a polyhedral region as in, *e.g.*, [21] requires polyhedral computations that scale exponentially in both n and p [29]. Above, we have assumed that f_m requires constant computation time. This

function STUTTERING(*system*, $\mathcal{D}$, $\mathcal{T}$, $(q, \mathcal{M}')$) **returns** `isStuttering`
  inputs: *system*, a mixed monotone system (3)
    $\mathcal{D}$, the disturbance set $\mathcal{D} = \cup_{\ell=1}^{L} \mathcal{D}^{\ell}$
    $\mathcal{T} = (\mathcal{Q}, \mathcal{M}, \delta)$, abstraction
    $(q, \mathcal{M}')$, a stuttering pair candidate
  initialize: `isStuttering` := `Null`
    `iter` := 1
    $\zeta_1^m := x_1^{q,m}$, $\zeta_2^m := x_2^{q,m}$ **for all** $m \in \mathcal{M}'$
  while `iter` $\leq N_{\max}$ **do**
    for $\ell := 1$ **to** L **do**
      for each $m \in \mathcal{M}'$ **do**
        $y_1^{m,\ell} := f_m(\zeta_1^m, d_1^{\ell}, \zeta_2^m, d_2^{\ell})$
        $y_2^{m,\ell} := f_m(\zeta_2^m, d_2^{\ell}, \zeta_1^m, d_1^{\ell})$
      end for
    end for
    if $\exists \ell \in \{0, \ldots, L\}$ $\exists m \in \mathcal{M}'$ s.t. $(y_1^{m,\ell} \preceq_m x_2^{q,m}) \wedge (x_1^{q,m} \preceq_m y_2^{m,\ell})$ **then**
      $\{\zeta_1^m, \zeta_2^m\}_{m \in \mathcal{M}'} :=$ GETNEWINT$(\{y_1^{m,\ell}, y_2^{m,\ell}\}_{m \in \mathcal{M}'}, \mathcal{I}_q)$
      `iter` := `iter` + 1
    else
      `isStuttering` := `True`
      break
    end if
  end while
  return `isStuttering`
end function

Algorithm 4: An algorithm to determine if $(q, \mathcal{M}')$ is stuttering. The parameter $N_{\max}$ determines how many time steps should be considered. The function getNewInt returns a set of points $\{\zeta_1^m, \zeta_2^m\}_{m \in \mathcal{M}}$ such that for each $m \in \mathcal{M}'$, $[\zeta_1^m, \zeta_2^m]_m \supseteq [y_1^{m',\ell}, y_2^{m',\ell}]_{m'} \cap \mathcal{I}_q$ for all $m' \in \mathcal{M}'$ and $\ell = 1, \ldots, L$.

is reasonable in some cases, such as the case study in Section 5.2 where intrinsic sparsity of traffic networks implies that the required computation time of f_m does not scale with n or p. However, in other cases, the complexity of evaluating f_m must be taken into account.

We further remark that $|\mathcal{Q}|$ typically increases exponentially with n. However, this dependence can be mitigated *via* various techniques such as interval partitions that incorporate domain specific knowledge. For example, the authors of [23] consider monotone systems that converge to a low-dimensional manifold, and suggest a methodology for abstracting the low dimensional manifold while retaining the intrinsic high dimensional dynamics. Future research will investigate related techniques for mixed monotone systems.

Thus, we summarize by emphasizing that the computational complexity of the proposed approach effectively does not depend directly on the state-space dimension n; this contrasts with many existing abstraction approaches for which n is a significant bottleneck. However, the complexity still depends crucially on $|\mathcal{Q}|$, the number of partitions in the abstraction.

5. CASE STUDIES

5.1 Verifying Oscillations in Insect Population Dynamics

We consider the following model from [30] for the population dynamics of the flour beetle *Tribolium castaneum*:

$$x^+ = A(x)x, \qquad x = (x_1, x_2, x_3) \in \mathbb{R}^3, \tag{37}$$

$$A(x) = \begin{pmatrix} 0 & 0 & b\exp(-c_{e\ell}x_1 - c_{ea}x_3) \\ p & 0 & 0 \\ 0 & \exp(-c_{pa}x_3) & q \end{pmatrix}, \tag{38}$$

where x_1, x_2, and x_3 represent populations of the insect at various stages of life (larvae, pupae, and adults, respectively), and $p, q \in (0, 1]$ are probabilities of survival. The exponential nonlinearities are the result of cannibalism of eggs and pupae. The dynamics are mixed monotone with $f(x, d, y, e) = A(y)x$ where $\preceq_{\mathcal{X}} = \preceq_+$.

Using parameters from [30], we let $b = 7.88$, $c_{ea} = 0.011$, $c_{e\ell} = 0.014$, $p = 0.839$, $q = 0.5$, and $c_{pa} = 0.0047$, with a time step of 2 weeks. We first note that the domain

$$\mathcal{X} = [0, (265, 225, 450)]_+ \tag{39}$$

is invariant. This follows because $bx_3\exp(-c_{ea}x_3) \leq 265$ for all $x_3 \geq 0$ and, thus, $x_1 \leq 265$ is invariant, from which $x_2 \leq p \cdot 265 \leq 225$. Since $x_3^+ \leq x_2 + qx_3$, we conclude that $x_3 \leq 225/(1-q) = 450$ is invariant.

For certain sets of parameters, the dynamics (37)–(38) induce oscillations in the number of larvae—a phenomenon documented in controlled laboratory experiments [30]. We wish to verify the following LTL formula which is a consequence of this oscillatory behavior:

$$\Box\Big(\big((x_1 \leq 10) \wedge (x_3 \geq 40)\big) \rightarrow \Diamond(x_1 \geq 150)\Big). \tag{40}$$

In words, "if the larvae population (x_1) reduces to a small number or zero and the adult population (x_3) is not too small, then the larvae population will eventually reach a large population size in the future."

We partition the state space into 2,376 intervals using a gridded partition. Computing the finite state abstraction takes less than one second on a standard personal computer. By applying Algorithm 4, we remove 14 self transitions that are stuttering. Checking the model with SPIN [31] took 103 seconds, and we verify that (40) is satisfied. Figure 4 shows a sample trajectory of the population dynamics initialized at $(x_1, x_2, x_3) = (0, 0, 300)$. We see that the larvae population does not reach the desired population 150 immediately, but it does so eventually around week 26.

5.2 Synthesizing Control Laws for Traffic Networks

We next synthesize a traffic signal control policy for a network of signalized intersections. We consider a discrete-time model of traffic flow where each road *link* contains a queue of vehicles waiting to proceed through an intersection. Each intersection signal actuates a subset of its queues at a given time step, and the vehicles in actuated queues are allowed to flow to downstream links if there is available space. This example builds on our recent result in [32] which studied only piecewise affine dynamics. In contrast, here we allow a nonlinear model with the help of the theory developed in Sections 3 and 4.

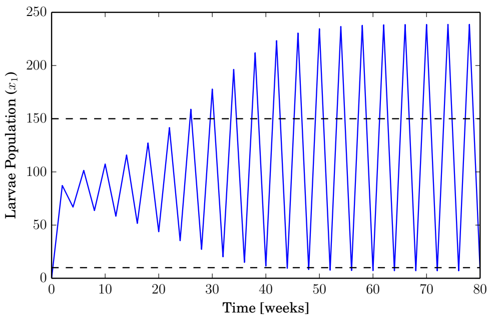

Figure 4: Sample trajectory of the insect population model (37)–(38)**, plotting x_1 over time when the system is initialized at** $(x_1, x_2, x_3) = (0, 0, 300)$**. The trajectory satisfies** (40)**.**

We consider a network of $\mathcal{L}$ *links* and a set $\mathcal{V}$ of *signalized intersections*. We assume each link $\ell \in \mathcal{L}$ has a queue of size $x_\ell \in [0, x_\ell^{\text{crit}}]$ representing the number of vehicles on the link where $x_\ell^{\text{crit}} > 0$ is the capacity of link $\ell \in \mathcal{L}$. By allowing x_ℓ to be continuous, we adopt a *fluid* model of traffic flow.

For $\ell \in \mathcal{L}$, let $\eta(\ell) \in \mathcal{V}$ denote the head node of link ℓ and let $\tau(\ell) \in \mathcal{V} \cup \emptyset$ denote the tail node. A link ℓ with $\tau(\ell) = \emptyset$ serves as an entry-point into the network, and we assume $\eta(\ell) \neq \tau(\ell)$ for all $\ell \in \mathcal{L}$ (*i.e.*, no self-loops). Link $k \neq \ell$ is *upstream* of link ℓ if $\eta(k) = \tau(\ell)$, *downstream* of link ℓ if $\tau(k) = \eta(\ell)$, and *adjacent to* link ℓ if $\tau(k) = \tau(\ell)$. Roads exiting the traffic network are not modeled explicitly. For each $v \in \mathcal{V}$, define

$$\mathcal{L}_v^{\text{in}} = \{\ell \mid \eta(\ell) = v\}, \qquad \mathcal{L}_v^{\text{out}} = \{\ell \mid \tau(\ell) = v\}. \tag{41}$$

For simplicity of notation, we assume each intersection $v \in \mathcal{V}$ has two possible states actuating either "East-West" (EW) incoming links or "North-South" (NS) incoming links. Thus, we have the partition $\mathcal{L} = \mathcal{L}^{\text{EW}} \cup \mathcal{L}^{\text{NS}}$, $\mathcal{L}^{\text{EW}} \cap \mathcal{L}^{\text{NS}} = \emptyset$. At each junction $v \in \mathcal{V}$, we define the signal variable $m_v \in \{0, 1\}$ as follows:

$$m_v = \begin{cases} 1 & \text{if links } \mathcal{L}_v^{\text{in}} \cap \mathcal{L}^{\text{EW}} \text{ are actuated} \\ 0 & \text{if links } \mathcal{L}_v^{\text{in}} \cap \mathcal{L}^{\text{NS}} \text{ are actuated.} \end{cases} \tag{42}$$

Let $m = \{m_v\}_{v \in \mathcal{V}}$ so that $\mathcal{M} = \{0, 1\}^{\mathcal{V}}$. When a link ℓ is actuated, the *turn ratio* $\beta_{\ell k}$ denotes the fraction of vehicles exiting link ℓ that is routed to link k. It follows that $\beta_{\ell k} \neq 0$ only if $\eta(\ell) = \tau(k)$ and

$$\sum_{k \in \mathcal{L}_{\eta(\ell)}^{\text{out}}} \beta_{\ell k} \leq 1. \tag{43}$$

Strict inequality in (43) implies that a fraction of vehicles on link ℓ are routed off the network via unmodeled roads.

Each link $\ell \in \mathcal{L}$ possesses a *demand* function $\Phi_\ell^{\text{out}} : [0, x_\ell^{\text{crit}}] \to \mathbb{R}$ that gives the number of vehicles wishing to flow downstream in one time step and a *supply* function $\Phi_\ell^{\text{in}} : [0, x_\ell^{\text{crit}}] \to \mathbb{R}$ that gives the available road space for incoming upstream vehicles in one time step. Thus, Φ_ℓ^{out} is an increasing function and Φ_ℓ^{in} is a decreasing function of queue length. In this example, we let

$$\Phi_\ell^{\text{out}}(x_\ell) = c_\ell(1 - \exp(-x_\ell/c_\ell)) \tag{44}$$

$$\Phi_\ell^{\text{in}}(x_\ell) = w_\ell(x_\ell^{\text{crit}} - x_\ell) \tag{45}$$

where $c_\ell > 0$ is a *saturation rate* and $0 < w_\ell < 1$ scales the available queue capacity to account for, *e.g.*, vehicles still traveling on the link and not enqueue. This demand-supply approach to vehicular traffic flow is rooted in the Cell Transmission Model [33].

Movement of vehicles among link queues is governed by mass-conservation laws and the state of the signalized intersections. When a link is *actuated*, a maximum of $\Phi_\ell^{\text{out}}(x_\ell)$ vehicles are allowed to flow from link ℓ to links $\mathcal{L}_{\eta(\ell)}^{\text{out}}$ per time step. We let $\alpha_{\ell k}$ denote the fraction of link k's supply available to link ℓ. Since only incoming EW or NS links are actuated in each time step, we have

$$\sum_{\ell \in \mathcal{L}_{\tau(k)}^{\text{in}} \cap \mathcal{L}^{\text{EW}}} \alpha_{\ell k} = \sum_{\ell \in \mathcal{L}_{\tau(k)}^{\text{in}} \cap \mathcal{L}^{\text{NS}}} \alpha_{\ell k} = 1 \tag{46}$$

for all $k \in \mathcal{L}$. It then follows that the dynamics on link ℓ are given by

$$x_\ell^+ = F_m^\ell(x, d) \tag{47}$$

$$\triangleq x_\ell - f_\ell^{\text{out}}(x, m) + \sum_{j \in \mathcal{L}_{\tau(\ell)}^{\text{in}}} \beta_{j\ell} f_j^{\text{out}}(x, m) + d_\ell \tag{48}$$

where

$$f_\ell^{\text{out}}(x, m) = s_\ell(m) \cdot \min\left\{\Phi_\ell^{\text{out}}(x_\ell), \min_{\substack{k \text{ s.t.} \\ \beta_{\ell k} \neq 0}} \frac{\alpha_{\ell k}}{\beta_{\ell k}} \Phi_k^{\text{in}}(x_k)\right\} \tag{49}$$

$$s_\ell(m) = \begin{cases} m_{\eta(\ell)} & \text{if } \ell \in \mathcal{L}^{\text{EW}} \\ 1 - m_{\eta(\ell)} & \text{if } \ell \in \mathcal{L}^{\text{NS}}. \end{cases} \tag{50}$$

Assumption 1. *For all $\ell \in \mathcal{L}$ and all k upstream of ℓ,*

$$\exp\left(\frac{-1}{c_\ell}\left(x_\ell^{crit} - \frac{\beta_{k\ell}}{w_\ell \alpha_{k\ell}} c_k\right)\right) \leq 1 - w_\ell. \tag{51}$$

Assumption 1 ensures that an increase in x_i does not lead to a decrease in x_i^+. This assumption is mild because (51) is satisfied for small enough c_ℓ and c_k, and these parameters decrease for shorter time steps; indeed, violation of the assumption would indicate that the chosen time step is to large to accurately capture the queue dynamics.

Lemma 2. *Assumption 1 ensures that $\frac{\partial F_m^\ell}{\partial x_\ell}(x, d) \geq 0$ for all m whenever the partial derivative exists*[1]*.*

Proof. We have

$$\frac{\partial F_m^\ell}{\partial x_\ell}(x, d) = 1 - \frac{\partial f_\ell^{\text{out}}}{\partial x_\ell}(x_\ell, m) + \sum_{j \in \mathcal{L}_{\tau(\ell)}^{\text{in}}} \beta_{j\ell} \frac{\partial f_j^{\text{out}}}{\partial x_\ell}(x, m). \tag{52}$$

Note that $\frac{\partial f_\ell^{\text{out}}}{\partial x_\ell}(x_\ell, m) \leq 1$. Furthermore, $\frac{\partial f_j^{\text{out}}}{\partial x_\ell}(x, m) \neq 0$ only if $s_j(m) = 1$ and $\frac{\alpha_{j\ell}}{\beta_{j\ell}} \Phi_\ell^{\text{in}}(x_\ell)$ is the minimizer in (49). As $\Phi_j^{\text{out}}(x_j) \leq c_j$, the latter condition can only occur if

$$c_j \geq \frac{\alpha_{j\ell}}{\beta_{j\ell}} w_\ell(x_\ell^{\text{crit}} - x_\ell) \iff x_\ell \geq x_\ell^{\text{crit}} - \frac{\beta_{j\ell}}{w_\ell \alpha_{j\ell}} c_j. \tag{53}$$

It then follows that $\sum_{j \in \mathcal{L}_{\tau(\ell)}^{\text{in}}} \beta_{j\ell} \frac{\partial f_j^{\text{out}}}{\partial x_\ell}(x, m) < 0$ only if there exists $j \in \mathcal{L}_{\tau(\ell)}^{\text{in}}$ such that the inequalities in (53) hold.

[1] The minimization in (49) implies that some partial derivatives do not exist on a set of measure zero. However, as noted above, the results developed in this paper still apply.

But this implies $\frac{\partial f_\ell^{\text{out}}}{\partial x_\ell}(x_\ell, m) \leq 1 - w_\ell$ by Assumption 1 and the fact that $\exp(-\frac{1}{c_\ell}x_\ell)$ decreases in x_ℓ. Furthermore, $\sum_{j \in \mathcal{L}^{\text{in}}_{\tau(\ell)}} \beta_{j\ell} \frac{\partial f_j^{\text{out}}}{\partial x_\ell}(x, m) \geq -w_\ell$, and we thus conclude that $\frac{\partial F_m^\ell}{\partial x_\ell}(x, d) \geq 0$. $\square$

Proposition 3. *The traffic dynamics* (47)–(48) *are mixed monotone.*

Proof. We show for all $\ell, k \in \mathcal{L}$, and all $m \in \mathcal{M}$,

$$\frac{\partial F_m^\ell}{\partial x_k}(x, d) \leq 0 \quad \text{if } \tau(k) = \tau(\ell), k \neq \ell \tag{54}$$

$$\frac{\partial F_m^\ell}{\partial x_k}(x, d) \geq 0 \quad \text{if } k = \ell \text{ or } \tau(k) \neq \tau(\ell) \tag{55}$$

$$\frac{\partial F_m^\ell}{\partial d_k}(x, d) = \begin{cases} 1 & \text{if } k = \ell \\ 0 & \text{if } k \neq \ell. \end{cases} \tag{56}$$

This implies that the conditions of Propositions 1 are satisfied, specifically, we take $s_j = 1$ if and only if $\tau(j) = \tau(i)$, $j \neq i$ and $\sigma_j = 0$ for all j in (7)–(8). Note that (56) follows immediately from (48). We now show (54)–(55) by considering 4 exhaustive cases:
(Case 1, $\tau(k) = \tau(\ell)$, $k \neq \ell$). We have that

$$\frac{\partial F_m^\ell}{\partial x_k}(x, d) = \sum_{j \in \mathcal{L}^{\text{in}}_{\tau(\ell)}} \beta_{j\ell} \frac{\partial f_j^{\text{out}}}{\partial x_k}(x, m). \tag{57}$$

Since $\frac{\partial f_j^{\text{out}}}{\partial x_k} \in \{0, \frac{\alpha_{jk}}{\beta_{jk}} \frac{d\Phi_k^{\text{in}}}{dx_k}\}$ and Φ_k^{in} is decreasing, we have $\frac{\partial F_m^\ell}{\partial x_k}(x, d) \leq 0$.
(Case 2, $\eta(k) = \tau(\ell)$ or $\tau(k) = \eta(\ell)$). We have that $\frac{\partial F_m^\ell}{\partial x_k}(x, d) \in \{0, -\frac{\partial f_\ell^{\text{out}}}{\partial x_k}, \beta_{k\ell}\frac{\partial f_k^{\text{out}}}{\partial x_k}, -\frac{\partial f_\ell^{\text{out}}}{\partial x_k} + \beta_{k\ell}\frac{\partial f_k^{\text{out}}}{\partial x_k}\}$ where the second possibility occurs only if $\tau(k) = \eta(\ell)$, the third occurs only if $\eta(k) = \tau(\ell)$, and the fourth occurs only if $\tau(k) = \eta(\ell)$ and $\eta(k) = \tau(\ell)$. We have $\partial f_k^{\text{out}}/\partial x_k \geq 0$ since Φ_k^{out} is increasing and $\frac{\partial f_\ell^{\text{out}}}{\partial x_k} \in \{0, \frac{\alpha_{\ell k}}{\beta_{\ell k}} \frac{d\Phi_k^{\text{in}}}{dx_k}\} \leq 0$ since Φ_k^{in} is decreasing, thus $\frac{\partial F_m^\ell}{\partial x_\ell}(x, d) \geq 0$.
(Case 3, $k = \ell$). $\frac{\partial F_m^\ell}{\partial x_\ell}(x, d) \geq 0$ by Lemma 2.
(Case 4, else). Trivially, $\frac{\partial F_m^\ell}{\partial x_\ell}(x, d) = 0$. $\square$

Consider the traffic network show in Figure 5 consisting of two signalized intersections and eight links. We have $\mathcal{L}^{\text{EW}} = \{1, 2, 3, 4\}$ and $\mathcal{L}^{\text{NS}} = \{5, 6, 7, 8\}$. The leftmost signal actuates the EW links 1 and 3 simultaneously, or the NS links 5 and 6 simultaneously, and similarly for the rightmost signal. We take the time step to be 15 seconds and assume $c_1 = c_2 = c_3 = c_4 = 20$, $c_5 = c_6 = c_7 = c_8 = 5$, $x_1^{\text{crit}} = x_4^{\text{crit}} = 50$, $x_2^{\text{crit}} = x_3^{\text{crit}} = 60$, $x_5^{\text{crit}} = x_6^{\text{crit}} = x_7^{\text{crit}} = x_8^{\text{crit}} = 40$, $w_\ell = 0.75$ for all ℓ, $\beta_{12} = \beta_{43} = \beta_{52} = \beta_{62} = \beta_{73} = \beta_{83} = 0.5$, $\alpha_{52} = \alpha_{62} = \alpha_{73} = \alpha_{83} = 0.5$, $\alpha_{12} = 1$, and $\alpha_{43} = 1$. For the disturbance input, we assume that at each time step, up to 7 vehicles join each of the queues on links 1 and 3, or up to 8 vehicles join each of the queues on links 5 and 6, or up to 8 vehicles join each of the queues on links 7 and 8.

We partition the domain of the traffic network, representing the state of all queues, into 3,600 boxes using a gridded partition. Using the mixed monotonicity properties of the dynamics, we obtain a finite state abstraction of the dynamics in 43.8 seconds.

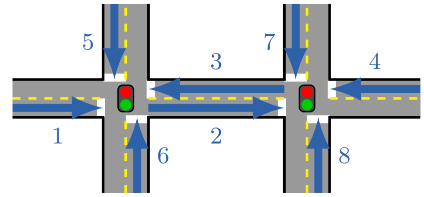

Figure 5: A traffic network with two signalized intersections and 8 links. The blue links represent queues of vehicles. The leftmost signal actuates links 1 and 3 simultaneously, or links 5 and 6 simultaneously. Likewise, the rightmost signal actuates links 2 and 4 or 7 and 8 simultaneously.

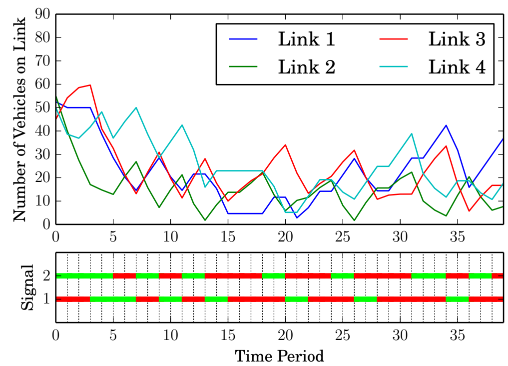

Figure 6: An example trajectory of the traffic network for links 1, 2, 3, and 4. Signal 1 (resp. 2) is the leftmost (resp. rightmost) signal in Figure 5. The trajectory satisfies the given specification. In the lower plot, green (resp. red) indicates that EW (resp. NS) links are actuated.

Next, we wish to find a controller that satisfies the specification:

> "Infinitely often, the cross streets on links 5 and 6 are actuated, AND infinitely often, the cross streets on links 7 and 8 are actuated, AND eventually, the queue lengths on links 2 and 3 are each less than 40 vehicles and remain so for all future time, AND whenever the queue on link 1 exceeds 40 vehicles, it eventually is less than 30 vehicles, AND whenever the queue on link 4 exceeds 40 vehicles, it eventually is less than 30 vehicles."

The above specification can be expressed in linear temporal logic and encoded in a deterministic *Rabin automaton* [15] with 46 states. By solving a Rabin game, we construct a controller that is guaranteed to satisfy the specification. In Figure 6, we plot an example trajectory of the system where we assume the maximum number of allowed vehicles enters the network in each time step. We see in the figure that the trajectory satisfies the above specification.

6. CONCLUSIONS

We have efficiently computed finite state abstractions for mixed monotone discrete-time systems. Mixed monotonicity is a general property encompassing many practical systems and provides a powerful tool for analysis and control. The primary feature that permits efficient abstraction is overapproximation of reachable sets by evaluating a decomposition function at two points. Future research will investigate using mixed monotonicity to reduce the number of intervals required to establish an effective partition of the state space.

Acknowledgements

This work was supported in part by the National Science Foundation under CPS Award 1446145.

7. REFERENCES

[1] M. W. Hirsch, "Systems of differential equations that are competitive or cooperative II: Convergence almost everywhere," *SIAM Journal on Mathematical Analysis*, vol. 16, no. 3, pp. 423–439, 1985.

[2] H. L. Smith, *Monotone dynamical systems: An introduction to the theory of competitive and cooperative systems*. American Math. Soc., 1995.

[3] M. Hirsch and H. Smith, "Monotone maps: a review," *Journal of Difference Equations and Applications*, vol. 11, no. 4-5, pp. 379–398, 2005.

[4] D. Angeli and E. Sontag, "Monotone control systems," *IEEE Transactions on Automatic Control*, vol. 48, no. 10, pp. 1684–1698, 2003.

[5] J. Gouzé and K. Hadeler, "Monotone flows and order intervals," *Nonlinear World*, vol. 1, pp. 23–34, 1994.

[6] H. Smith, "The discrete dynamics of monotonically decomposable maps," *Journal of Mathematical Biology*, vol. 53, no. 4, pp. 747–758, 2006.

[7] H. Smith, "Global stability for mixed monotone systems," *Journal of Difference Equations and Applications*, vol. 14, no. 10-11, pp. 1159–1164, 2008.

[8] G. Enciso, H. Smith, and E. Sontag, "Nonmonotone systems decomposable into monotone systems with negative feedback," *Journal of Differential Equations*, vol. 224, no. 1, pp. 205–227, 2006.

[9] R. Alur, T. A. Henzinger, G. Lafferriere, and G. J. Pappas, "Discrete abstractions of hybrid systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 971–984, 2000.

[10] A. Tiwari and G. Khanna, "Series of abstractions for hybrid automata," in *Hybrid Systems: Computation and Control*, pp. 465–478, Springer, 2002.

[11] P. Tabuada, *Verification and control of hybrid systems: a symbolic approach*. Springer, 2009.

[12] T. Wongpiromsarn, U. Topcu, and R. Murray, "Receding horizon temporal logic planning," *IEEE Transactions on Automatic Control*, vol. 57, pp. 2817–2830, Nov 2012.

[13] J. Liu, N. Ozay, U. Topcu, and R. Murray, "Synthesis of reactive switching protocols from temporal logic specifications," *IEEE Transactions on Automatic Control*, vol. 58, pp. 1771–1785, July 2013.

[14] E. M. Clarke, O. Grumberg, and D. A. Peled, *Model checking*. MIT press, 1999.

[15] C. Baier and J. Katoen, *Principals of Model Checking*. MIT Press, 2008.

[16] T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya, "What's decidable about hybrid automata?," *Journal of Computer and System Sciences*, vol. 57, no. 1, pp. 94–124, 1998.

[17] P. Tabuada and G. Pappas, "Linear time logic control of discrete-time linear systems," *IEEE Transactions on Automatic Control*, vol. 51, no. 12, pp. 1862–1877, 2006.

[18] A. Girard and G. J. Pappas, "Approximation metrics for discrete and continuous systems," *IEEE Transactions on Automatic Control*, vol. 52, no. 5, pp. 782–798, 2007.

[19] A. Girard, G. Pola, and P. Tabuada, "Approximately bisimilar symbolic models for incrementally stable switched systems," *IEEE Transactions on Automatic Control*, vol. 55, no. 1, pp. 116–126, 2010.

[20] G. Reißig, "Computing abstractions of nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 56, no. 11, pp. 2583–2598, 2011.

[21] B. Yordanov, J. Tůmová, I. Černá, J. Barnat, and C. Belta, "Temporal logic control of discrete-time piecewise affine systems," *IEEE Transactions on Automatic Control*, vol. 57, no. 6, pp. 1491–1504, 2012.

[22] D. Adzkiya, B. De Schutter, and A. Abate, "Finite abstractions of max-plus-linear systems," *IEEE Transactions on Automatic Control*, vol. 58, pp. 3039–3053, Dec 2013.

[23] T. Moor and J. Raisch, "Abstraction based supervisory controller synthesis for high order monotone continuous systems," in *Modelling, Analysis, and Design of Hybrid Systems*, pp. 247–265, Springer, 2002.

[24] D. Gromov and J. Raisch, "Detecting and enforcing monotonicity for hybrid control systems synthesis," in *Proc. 2nd IFAC Conf. on Analysis and Design of Hybrid Systems*, pp. 7–9, 2006.

[25] N. Ramdani, N. Meslem, and Y. Candau, "Computing reachable sets for uncertain nonlinear monotone systems," *Nonlinear Analysis: Hybrid Systems*, vol. 4, no. 2, pp. 263–278, 2010.

[26] J. M. Cushing, *An introduction to structured population dynamics*. SIAM, 1998.

[27] M. Kulenovic and O. Merino, "A global attractivity result for maps with invariant boxes," *Discrete and Continuous Dynamical Systems Series B*, vol. 6, no. 1, p. 97, 2006.

[28] D. Peled and T. Wilke, "Stutter-invariant temporal properties are expressible without the next-time operator," *Information Processing Letters*, vol. 63, no. 5, pp. 243–246, 1997.

[29] A. Kurzhanskiy and P. Varaiya, "Computation of reach sets for dynamical systems," in *The Control Systems Handbook*, ch. 29, CRC Press, second ed., 2010.

[30] R. Costantino, J. Cushing, B. Dennis, and R. A. Desharnais, "Experimentally induced transitions in the dynamic behaviour of insect populations," *Nature*, vol. 375, no. 6528, pp. 227–230, 1995.

[31] G. J. Holzmann, "The model checker SPIN," *IEEE Transactions on Software Engineering*, vol. 23, pp. 279–295, 1997.

[32] S. Coogan, E. A. Gol, M. Arcak, and C. Belta, "Traffic network control from temporal logic specifications," 2014. arXiv:1408.1437.

[33] C. F. Daganzo, "The cell transmission model, part II: Network traffic," *Transportation Research Part B: Methodological*, vol. 29, no. 2, pp. 79–93, 1995.