Counterexample Guided Synthesis of Switched Controllers for Reach-While-Stay Properties

Hadi Ravanbakhsh and Sriram Sankaranarayanan *

May 7, 2015

Abstract

We introduce a counter-example guided inductive synthesis (CEGIS) framework for synthesizing continuous-time switching controllers that guarantee reach while stay (RWS) properties of the closed loop system. The solution is based on synthesizing specially defined class of control Lyapunov functions (CLFs) for switched systems, that yield switching controllers with a guaranteed minimum dwell time in each mode. Next, we use a CEGIS-based approach to iteratively solve the resulting quantified exists-forall constraints, and find a CLF. We introduce relaxations to guarantee termination, as well as heuristics to increase convergence speed. Finally, we evaluate our approach on a set of benchmarks ranging from two to six state variables. Our evaluation includes a preliminary comparison with related tools. The proposed approach shows the promise of nonlinear SMT solvers for the synthesis of provably correct switching control laws.

1 Introduction

In this paper, we study the problem of automatically synthesizing continuous-time switching controllers for ensuring RWS property of a polynomial system. Our approach considers plant models with continuous state variables whose dynamics are described by ODEs. The dynamics depend, however, on the choice of finitely many control modes. The goal of the controller synthesis is to find a continuous-time, switching controller that chooses a control mode, given the current mode and continuous state.

Our approach works by synthesizing a control Lyapunov function (CLF), which can be made to decrease along the traces of the closed loop system, guaranteeing that the traces reach a designated, desirable region while stays in the safe region. As such, finding a CLF yields a switching function that simply chooses an appropriate control mode that ensures its decrease. However, for continuous-time switching, we are faced with the problem of *zenoness* caused by the controller switching infinitely often in a finite time interval, and thus preventing time from diverging. Therefore, we provide sufficient conditions on the CLF that ensure that the resulting switching function respects a minimum dwell time for each control mode.

The synthesis procedure iteratively searches for a CLF using Satisfiability Modulo Theory (SMT) solvers through a well-known procedure for program synthesis called *counter-example guided inductive synthesis* (CEGIS) [1] [2] [3]. Whereas CEGIS was originally proposed for synthesizing unknown parameters for programs (called *sketches*) so that assertions (safety properties) in the program are satisfied by all executions, we propose to reuse the basic insights for synthesizing controllers.

Since the search space is infinite (and continuous), there is no guarantee that the process terminates. We show a adaptation of CEGIS to our setting that ensures eventual termination of the CEGIS algorithm.

We provide an implementation of the CEGIS approach for synthesizing controllers using the SMT solvers Z3 for linear arithmetic [4] and the dReal δ−satisfiability solver for nonlinear arithmetic constraints [5]. The evaluation shows the ability of our approach to effectively synthesize switching controllers with guaranteed minimal dwell time for a number of small but interesting benchmarks. On the other hand, our approach is currently restricted to small systems due to the high complexity of nonlinear arithmetic decision procedures. Nevertheless, we provide a preliminary comparison that suggests that our approach is quite competitive.

* Authors are with the Department of Computer Science, University of Colorado - Boulder, USA
with other state-of-the-art approach for the synthesis of controllers to guarantee temporal logic objectives. However, we are exploring relaxations for the non-linear constraints using well-known schemes such as SOS programming [6]. An important technical limitation of these relaxations that prevents their direct use in this paper lie in the lack of useful witnesses in case a given set of constraints are satisfiable. In summary, the contributions of this paper are as follows:

- We introduce a sufficient condition on CLFs for synthesizing a minimum dwell-time enforcing controller that guarantees RWS.
- We adapt the well-known CEGIS algorithm to discover CLFs for polynomial switched systems [11][2][3].
- We show how the CEGIS search for candidate CLFs can be modified to guarantee termination in finitely many steps.
- We employ a heuristic to find better witness points that significantly improve the behavior of the proposed approach.
- We provide an experimental evaluation on a number of small but interesting benchmarks that demonstrates the promise of our approach, as well as its limitations.

1.1 Related Works

Verification: The stability of cyber-physical systems has been studied widely in the literature. Lyapunov functions remain a simple, yet powerful, approach for these analysis. The problem of synthesizing Lyapunov functions has been well studied using ideas such as SOS programming that reduces the conditions for a Lyapunov function for nonlinear system to a semi-definite optimization problem [7][8][8]. Results through Lyapunov functions naturally extend to liveness properties (such as RWS) as well as switched and hybrid systems.

Synthesis: Beyond verification, much work has focused on designing correct-by-construction controllers for various liveness properties, especially stability. The problem for synthesis is generally much harder than verification. A common approach to synthesizes control Lyapunov functions (CLFs) whose values can be decreased at each time instant through the choice of an appropriate control [9].

One approach is to find a static feedback law for the controller that guarantees the liveness properties of the system. A necessary condition on Lyapunov functions for this class of controllers is proposed by Artstein through control Lyapunov functions [9]. The problem formulation for finding CLFs usually yields a bilinear matrix inequality (BMI), and this BMI problem is either solved directly [10] or using a heuristic such as V-K iteration [11] (or elsewhere called policy iteration [12]), which is often susceptible to failures due to local minima. Tan et. al. [13] formulate these conditions as a BMI and use off-the-shelf approaches to tackle the resulting BMIs. Rifford [14] discusses the converse results on control Lyapunov function, i.e. if a system is globally asymptotically controllable, then there exists a locally Lipschitz control Lyapunov function. This justifies the use of Lyapunov function based methods for controller synthesis.

Switched Controllers: In this article, the problem is to find a switching logic such that the closed loop system satisfies a RWS property. A large volume of work on switched system has focused on linear switched systems and the use of linear matrix inequalities to find controllers. Details are available from the textbook by Liberzon [15], and the survey articles by Lin and Ansaklis [16][17]. Our approach here considers the continuous-time switched systems of more general polynomial dynamical systems. Furthermore, our focus is on synthesis to guarantee a minimum dwell time in each switching mode, and the use of CEGIS to find CLFs. These aspects are, to the best of our knowledge, unique to this work.

Another approach to controller synthesis is proposed by Taly et al. [18]. They consider the problem of synthesizing switching conditions for hybrid systems, so that the resulting system guarantees safety and liveness properties. The proposed method proves reachability by finding some progress certificates similar to Lyapunov functions. They reduce their synthesis to solving a system of nonlinear constraints. Our work here differs in the certificates used to prove desired property (these certificates are much simpler in our method) and the process of finding such certificate.

Another paradigm for synthesizing controllers is to define an abstract system and find a simulation (or approximate bisimulation) relation between the abstract system and the original system. These approaches are able to handle more general specifications (usually a sub-class of LTL) and they are not restricted to liveness properties, per se. The PESSOA tool [19] uses finite abstraction to discretize a continuous-time
system, and solves a completely discrete problem. One problem with this approach is the number of the abstract states, which can grow very large if we want them to be precise. Recently, Câmara et. al. [20] proposed multi-scale abstraction to keep the number of states small and subsequently, Nilsson et. al. [21] proposed a CEGAR-based approach to refine the abstraction, whenever it is needed to avoid large number of abstract states. Our approach does not directly partition the state-space. On the other hand, the nonlinear SMT solvers such as dReal that are used in the CEGIS approach implicitly partitions the state-space during the search for a CLF. However, such a partitioning is adaptive and is guided by the formula whose satisfiability is being decided. The preliminary evaluation provided shows that our approach can potentially be much faster in terms of time.

Fast Switches Most of aforementioned works consider discrete feedback for switched systems. When the feedback is continuous, extra care should be taken for infinitely fast switches. This phenomena is common in many types of control, including sliding mode control [22, 23]. Asarin et. al. [24] propose another method for enforcing min-dwell time property for finite-abstraction based synthesis. Taly et. al. [18] use “Progress Invariants” to prove min-dwell time properties (for each switch the value of Lyapunov function should decrease at least $\epsilon > 0$ unit). In this article, we develop a much simpler certificate (slightly modified standard CLF) along with switching strategy to guarantee a minimum dwell time.

CEGIS: The CEGIS framework has been used widely for solving $\exists \forall$ formulae in synthesis problems [3]. The idea here is to find a candidate solution based on some finite number of examples. Although CEGIS was first proposed in the computer science literature by Solar-Lezama et al. [1], variants of this approach are not unknown to the hybrid systems community. This strategy has been mainly used to find a solution for $\exists \forall$ formulae to solve parameter synthesis problems in programming languages (to mention few) and hybrid systems [27, 28]. Topcu et al. consider a simulation-based approach for finding maximal region of attraction for continuous systems [29]. They employ a CEGIS-like approach that avoids solving a BMI through sampling finitely many witness points that are likely to belong to the region of attraction. A LMI is used to search for a Lyapunov function that includes these witness points. Also Kapinski et. al. [30] employ a CEGIS approach for synthesizing Lyapunov functions based on simulation results (initiated from witness points). In contrast, our approach considers switched systems and focuses on synthesizing CLFs. We also do not perform any sort of simulations for the witness points in our approach.

2 Preliminaries

Let $\mathbb{N}$, $\mathbb{R}$ and $\mathbb{R}^+$ denote the set of natural, real and nonnegative real numbers respectively. Let $\mathbf{0}$ be the zero vector of proper size. For $n \in \mathbb{N}$ and real number $\delta > 0$, let $B_\delta(x_c)$ be a ball with radius $\delta$ and $x_c$ as its center ($B_\delta(x_c) = \{ x \ | \ |x - x_c| \leq \delta \}$). For a set $S$, let $\partial S$ and $\text{int}(S)$ be boundary and interior of $S$, respectively. Let $\mathbb{R}[x]$ denote the set of all polynomials involving variables in $x$, wherein each polynomial is written as a finite sum $p : \sum_{\alpha \in \mathbb{N}^n} c_\alpha x^\alpha$, where the multi-index $\alpha$ is used to denote a monomial $x^\alpha$ and $c_\alpha \in \mathbb{R}$ is a coefficient. A template polynomial over coefficients $C$ is a polynomial $F(x, c) : \sum_{\alpha \in \mathbb{N}^n} c_\alpha x^\alpha$ whose coefficients are parameterized by a set of template variables $c_\alpha \in C$. Given a function $f : \mathbb{R} \rightarrow \mathbb{R}$, $f^-(t)$ denotes the left limit: $\lim_{s \rightarrow t^-} f(s)$ and $f^+(t)$ denotes the right limit: $\lim_{s \rightarrow t^+} f(s)$. As a convention, let $\dot{f}(t)$ denote the right derivative of the function: $\lim_{h \rightarrow 0} \frac{f(t+h)-f(t)}{h}$ at $x = t$. If $f$ is differentiable then $\dot{f}$ coincides with its derivative.

System Model: We first discuss the system model for our controller synthesis problem. The system has a plant model which describes the physical environment with continuous dynamics. Also, the system has a controller which provides a mode for the plant (Figure 1).

The plant has continuous state variables $x \in \mathbb{R}^n$. The controller continuously chooses a mode from a finite set of possible control modes $q \in Q$, wherein the dynamics of the continuous state variables may depend on the chosen control mode $q$. Formally, we model the plant using a switched polynomial system:

Definition 1 (Switched Polynomial System) A switched polynomial system is a tuple $\Psi : (Q, X, f)$, consisting of (A) Continuous state-

![Figure 1: The closed loop model of the plant and the controller.](image-url)
space: \( X \subseteq \mathbb{R}^n \) (\( n \) is the number of continuous state variables); (B) A finite set \( Q \) of (control) modes; and (C) A map from each mode \( q \in Q \) to a polynomial vector field \( f_q \in \mathbb{R}[x]^n \), specifying its dynamics.

The controller is modeled as a memoryless state feedback switched controller.

**Definition 2 (Switching Controller)** Given a plant \( \Psi : (Q, X, f) \), a switching controller switch is specified by a function \( \text{switch}(q, x) \) that maps each current mode \( q \in Q \) and plant state \( x \in X \) to a next mode \( \hat{q} \in Q \).

A control implementation conforms to this specification by choosing the next mode specified in \( \text{switch}(q, x) \).

**Definition 3 (Closed Loop System)** The composition of the plant \( \Psi \) and a controller switch yields a deterministic switched system \( \Phi(X, Q, f, G) \) with continuous variable \( x \), modes \( Q \), dynamics given by \( f_q \) in each mode \( q \in Q \). The transition from mode \( q \) to \( \hat{q} \) has the guard set \( G_{q, \hat{q}} : \{x | \text{switch}(q, x) = \hat{q}\} \)

The set of traces of the closed-loop switched system represents all executions of the switched system that respects the plant dynamics and switches according to the controller specification. Formally, a trace \( \text{tr} \in \mathbb{R}^+ \rightarrow Q \times X \) is a function mapping time \( t \in \mathbb{R}^+ \) to the mode and state of the plant at that time. Let \( \text{tr}_X(\text{tr}_Q) \) denote the projections of the trace \( \text{tr} \) onto the sets \( X \) (and \( Q \)). For a trace to be valid, it must satisfy the following conditions:

- The set of switching times \( \text{SwitchTimes}(\text{tr}) : \{t \in \mathbb{R}^+ | \text{tr}_Q^+(t) \neq \text{tr}_Q^-(t)\} \) is a finite, or countably infinite set.

- For all non-switching times \( t \in \mathbb{R}^+ \setminus \text{SwitchTimes}(\text{tr}) \), writing \( q : \text{tr}_Q(t) \), we have \( \text{tr}_X \) is differentiable at \( t \) and \( \frac{d\text{tr}_X}{dt} = f_q(\text{tr}_X(t)) \).

- For all switching times \( t \in \mathbb{R}^+ \cap \text{SwitchTimes}(\text{tr}) \), writing \( q : \text{tr}_Q(t) \) and \( \hat{q} : \text{tr}_Q^+(t) \), we have \( \hat{q} = \text{switch}(q, \text{tr}_X(t)) \) and the right derivative \( \frac{d\text{tr}_X}{dt} = f_q(\text{tr}_X(t)) \).

A trace is time divergent if for all \( \Delta > 0 \), \( \text{SwitchTimes}(\text{tr}) \cap [0, \Delta] \) is a finite set.

**Specification:** We want the continuous state \( x \) to reach from initial (compact) set \( I \subset X \) to goal (compact) set \( G \subset X \), while staying in safe (compact) sets \( S \subset X \).

**Definition 4 (Reach While Stay)** Given compact sets \( S, G \) and \( I \), the closed loop switched system \( \Phi \) satisfies RWS w.r.t \( (I, G, S) \) if and only if for all traces \( \text{tr} \), \( (\text{tr}_X \in I) \implies (\text{tr}_X \in S) \cup (\text{tr}_X \in G) \).

To eliminate some technical arguments we assume \( I \subseteq \text{int}(S) \). The problem we study in this paper is synthesizing a controller that guarantees RWS w.r.t \( (I, G, S) \) for the closed loop system. Also we are interested in finding a big region \( W \supseteq I \) s.t. the system satisfies RWS w.r.t \( (W, G, S) \).

**Problem 1** Given compact regions \( S, G \) and \( I \subseteq \text{int}(S) \) and a plant \( \Psi \), find a switch function and a region \( W \supseteq I \) s.t. the closed loop switched system \( \Phi \) satisfies RWS w.r.t \( (W, G, S) \).

### 3 Lyapunov Function for Switched Systems

Lyapunov functions are an useful tool for proving stability and other liveness properties. We recall these concepts in this section and then how such function can be employed for switched system. By Lyapunov function, in this section, we mean Lyapunov functions which are useful for proving RWS problems.

**Definition 5** A Lyapunov function for RWS w.r.t \( (I, G, S) \) is a continuous function \( V : X \rightarrow \mathbb{R}^+ \) iff there exists a constant \( \beta \) s.t.

1. \( \forall x \in \partial S \setminus G \) \( V(x) > \beta \)
2. \( \forall x \in I \setminus G \) \( V(x) < \beta \)
3. \( \forall t, t'' > t' \geq 0 \) \( \langle \text{tr}_X(t') \notin \text{int}(G) \land V(\text{tr}_X(0)) < \beta \rangle \) \( \implies V(\text{tr}_X(t'')) < V(\text{tr}_X(t')) \)
For a given trace $\text{tr}$, let $\text{tr}_V(t) = V(\text{tr}_X(t))$. Because of the continuity of $\text{tr}_X$ and $V$, the third condition is equivalent to

$$\forall \text{tr}, (\forall t \geq 0) \ (\text{tr}_X(t) \notin \text{int}(G) \wedge V(\text{tr}_X(t)) < \beta) \implies \text{tr}_V(t) = \frac{d}{dt}V(\text{tr}_X(t)) < 0$$

(1)

The third condition simply implies that the value of Lyapunov function decreases through time.

**Definition 6 (Region for Lyapunov Function)** Given a Lyapunov function $V$, let $\beta$ be the constant in Def. 3. We say that $W$ is an associated region to $V$ iff $W$ satisfies $V(x) \leq \beta \cap S$.

It is easy to show that $W$ is a compact set, $x \in \partial W \implies V(x) = \beta, I \setminus G \subseteq \text{int}(W)$ and $W \setminus G \subseteq \text{int}(S)$. Ultimately, we want to show $(\text{tr}_X \in W) \implies (\text{tr}_X \in W) \cup (\text{tr}_X \in G)$ and since $W \subseteq \text{int}(S)$, the system satisfies RWS w.r.t. $(W, G, S)$. Our overall strategy for controller synthesis is to synthesize a (control) Lyapunov function. Such function can provide a control strategy as well as region $W$ such that the closed loop system satisfies the specification. However, while Lyapunov functions extend to proving stability for switched/hybrid systems [15], care must be taken to ensure that these techniques are not applied to systems with *time-convergent* traces. Defining the asymptotic behavior of such traces as $t \to \infty$ is clearly not meaningful, when the time $t$ never diverges. It is possible for such trajectories to “converge” to a non-target state, even when a Lyapunov function decreases and Lyapunov function is not sufficient to prove the desired specification.

Secondly, since our goal is to synthesize controllers, time convergent behaviors represent physically unrealizable control strategies, and must be avoided in our closed loop systems. Therefore, it is quite essential that our controller synthesis technique guarantee that the traces of the resulting closed loop system are all time divergent.

### 3.1 Control Lyapunov Function

We focus, in this work, on finding polynomial control Lyapunov functions for guaranteeing RWS.

**Definition 7 (Control Lyapunov Function)** A control Lyapunov function (CLF) w.r.t. $(I, G, S)$ and a plant $\Psi$ is a polynomial function $V(x)$ iff there exist a $\beta$ s.t.

$$\begin{align*}
(\forall x \in S \setminus G) \ V(x) &> \beta \\
(\forall x \in I \setminus G) \ V(x) &< \beta \\
(\forall x \in S \setminus \text{int}(G)) \ (\exists q \in Q) \ V_q(x) &= \frac{d}{dx}f_q(x) < 0
\end{align*}$$

(2)

Given a control Lyapunov function $V$ w.r.t. $(I, G, S)$, we define an associated set of switching functions that define controllers, as follows.

**Definition 8 (Switching Function for CLF)** Given a CLF $V$ and a function $\text{switch} : Q \times X \mapsto Q$, we say that $\text{switch}$ is compatible with $V$ iff for every state $x \in S \setminus G$ and mode $q$, the mode $\hat{q} : \text{switch}(q, x)$ is such that $V_{\hat{q}}(x) < 0$.

In other words, the controller at any state and any mode chooses an input $q \in Q$ that makes the control Lyapunov function decrease “instantaneously”. Given a CLF, we wish to synthesize a controller that establishes RWS w.r.t. $(W, G, S)$. However, we cannot yet guarantee that the trajectories of the closed loop system will all be time divergent. As a result, we first tackle the problem of finding a switching function that yields a minimum dwell time for each mode $q \in Q$. In other words, whenever the controller switches into a mode $q \in Q$, it must stay in that mode for at least some time $\delta_m > 0$ before transitioning to a different mode $\hat{q}$. The presence of such a dwell time is a minimum requirement for implementing a controller in practice, and allows us to talk about temporal properties of its time divergent traces.

**Non-Zeno CLFs** We define a class of CLFs that we term *Non-Zeno CLFs*, and an associated class of switching functions that can guarantee a minimal dwell time requirement.

**Definition 9 (Non-Zeno CLFs)** A control Lyapunov function (satisfying Definition 7) is said to be non-zeno iff there exists a constant $\epsilon_Q > 0$ such that

$$\forall x \in S \setminus G \ (\exists q \in Q) \ V_q(x) = \frac{dV}{dx}f_q(x) < -\epsilon_Q$$
Note that a non-zeno CLF replaces the third condition in Def. [7] by a stronger condition. We will now explain how non-zeno CLFs lead to controllers that can enforce a minimal dwell time on switching by first presenting a canonical switching strategy associated with a non-zeno CLF.

**Controllers for Non-Zeno CLF:** As usual, the goal of the controller is to ensure that the non-zeno CLF \( \dot{V}(x) \) decreases along any trace. Suppose the current control mode is given by \( q \). The controller calculates the value of \( \dot{V}(x) \), the Lie derivative of \( V \) according to the dynamics in mode \( q \). The controller distinguishes 3 situations:

1. **RESIDE** \((q)\): \( \dot{V}(x) \leq -\epsilon Q \). In this case, the controller stays in mode \( q \).

2. **TRANSIT** \((q)\): \(-\epsilon_Q < \dot{V}(x) < -\alpha Q\), for a constant \( \alpha_Q \) \((0 < \alpha_Q < \epsilon_Q)\). The controller continues in mode \( q \), but may potentially transit to another mode \( \hat{q} \) in the future.

3. **SWITCH** \((q)\): \( \dot{V}(x) \geq -\alpha Q \). At this instant, the controller switches to a new mode \( \hat{q} \) that satisfies \( \dot{V}(\hat{q}) \leq -\epsilon_Q \). By the definition of a CLF, such a mode \( \hat{q} \) exists (when \( x \in S \setminus G \)), and the controller jumps to the **RESIDE** \((\hat{q})\) mode for \( \hat{q} \).

The key result is that whenever the controller transitions from **RESIDE** to **TRANSIT**, there is a fixed lower bound on the time for the plant to reach the **SWITCH** state. As a result, a minimum dwell time can always be guaranteed.

**Theorem 1** Given compact sets \( S, G, I \subseteq \text{int}(S) \), a plant \( \Psi \) and a non-zeno CLF \( V(x) \) w.r.t \((I,G,S)\) with associated region \( W \), there exists a class of switching functions such that the closed loop \( \Phi \) satisfies the following properties.

1. System satisfies RWS w.r.t \((W,G,S)\).
2. All the traces of the closed loop system starting from \( W \) are time divergent before reaching \( G \).

**Model-Predictive Control:** Using the CLF, we use a model predictive control scheme. The scheme uses a software-based controller that executes in a time triggered fashion. At each invocation, it is assumed to be able to preset future mode switches over a time horizon \( \Delta > 0 \). More precisely, given the current state \( x \) and mode \( q \in Q \), it computes through its predictive model a time \( \delta > 0 \) for which the control can “safely” continue in the mode \( q \) according to the predictive model. It then sets a timer-based interrupt to awaken after this time \( \delta \) and command the plant to switch modes. Theorem 1 guarantees that the controller does not “wake up” more than some finite time in finite time horizon.

### 4 Synthesizing CLFs

The described solution in previous section reduces Problem 1 to finding a non-zeno CLF. In this section, we focus on the problem of searching a non-zeno CLF. First, we introduce a general CEGIS framework for solving \( \exists \forall \) formula for real arithmetics.

The general problem we wish to solve has the following form:

\[
(\exists c \in C) \begin{cases}
\forall_j \bigwedge_k G_{j,k}(c) < 0 \\
(\forall x \in R_1) \bigvee_k F_{1,k}(x,c) < 0 \\
(\forall x \in R_2) \bigvee_k F_{2,k}(x,c) < 0 \\
\vdots \\
(\forall x \in R_m) \bigvee_k F_{m,k}(x,c) < 0
\end{cases}
\]

where \( R_j \) is a fixed compact region and \( F_{j,k} \) are functions polynomial in \( x \) and linear in \( c \). Also \( G_{j,k} \) are functions linear in \( c \). The CEGIS algorithm was first introduced to tackle \( \exists \forall \) constraints such as (3) by Solar-Lezama et al. [2][1]. The basic idea is to maintain two sets:

1. A finite set of witnesses: \( X' : \{x_1, \ldots, x_m\} \subseteq X \), namely, the \( X \)-space.
2. A subset \( C_i \subseteq C \), namely, the \( C \)-space.
The $C$-space represents the set of candidates which are to be examined by our procedure while the $X$-space represents test points over which we will test a candidate. The $i^{th}$ iteration involves the following steps:

Step 1) Choose an arbitrary $c_i \in C_i$ to get candidate solution for Eq. (3).

Step 2) Check if Eq. (3) holds for $c_i$.

(a) If Eq. (3) holds, procedure terminates immediately.

(b) Otherwise, we obtain a point $x_i$ at which Eq. (3) fails. We add $x_i$ to the set of test points $(X_{i+1} : X_i \cup \{x_i\})$.

Step 3) We now refine the $C$-Space by removing all candidates which fail at $x_i$ by not satisfying (3).

The procedure terminates successfully if a solution is found. Alternatively, if $C_j = \emptyset$ then it terminates without finding a solution. Finally, the procedure may run out of time/memory due to the high complexity of various constraint solver calls.

**Representing the C-space:** We represent each set $C_i$ using a linear arithmetic formula $\psi_i(c)$ such that $C_i = \{ c \mid \psi_i(c) \text{ holds} \}$. The initial formula $C_0$ is simply $\psi_0 : \bigvee_j \bigwedge_k G_{j,k}(c) < 0$. Step 1 is implemented using an SMT solver to check if $\psi_i$ is satisfiable and obtain a candidate $c_i$ as a solution to $\psi_i$. Likewise, step 3 is implemented by augmenting $\psi_i$ to yield a formula

$$\psi_{i+1} : \psi_i \land \bigwedge_j \psi_{i,j}$$

where $\psi_{i,j}$ is True if $x_i \notin R_j$ and $\bigvee_k F_{j,k}(x_i, c) < 0$ otherwise.

**Finding Witnesses:** On the other hand, finding witnesses requires us to check the satisfiability of a non-linear constraints obtained by negating (3):

$$\begin{align*}
    x &\in R_1 \land \bigwedge_k F_{1,k}(x, c_i) \geq 0 \\
    &\vdots \\
    x &\in R_m \land \bigwedge_k F_{m,k}(x, c_i) \geq 0
\end{align*}$$

(5)

If yes, we obtain a witness $x_i$ at which the current candidate $c_i$ fails to satisfy Eq. (3) Otherwise, we conclude problem is solved. However, solving this constraint requires a nonlinear arithmetic solver that is capable of finding witnesses. Relaxations such as SOS programming can be used to check whether the formula (5) is unsatisfiable [6], but failing this, they do not provide useful witnesses to generate future candidates. Therefore, we resort to a promising approach for checking (5) implemented in the tool dReal [5]. dReal checks if the formula is unsatisfiable, and if it reports UNSAT, we conclude that the constraints (5) are indeed unsat. Otherwise, it reports that a $\delta$-perturbation is satisfiable, and provides us a witness. Therefore, using dReal, we run the risk of obtaining additional possibly spurious witnesses, and not recognizing if a solution has already been found. However, the resulting procedure will not yield a wrong solution.

### 4.1 Searching for CLF

Given a plant $\Psi$, and regions $S$, $G$ and $I$, We fix a *template* polynomial form $V(x, a)$ parameterized by variables in $a \in A$ as the desired CLF. The space $A$ is a compact set chosen as a hyper-rectangle limiting each $a_i \in [L_i, U_i]$. Formally, we wish to find $c = (a, \epsilon_Q, \beta) \in C : \mathbb{R}^m \times \mathbb{R}_+ \times \mathbb{R}$ that satisfies the conditions in Def. 9

$$\exists c \in C \left\{ \begin{array}{l}
    (\epsilon_Q > 0 \land \bigwedge_k a_k > L_i \land a_k < U_i) \\
    (\forall x \in \partial S \setminus G) \ V(x, a) > \beta \land \\
    (\forall x \in I \setminus G) \ V(x, a) < \beta \land \\
    (\forall x \in S \setminus G) \left( \bigvee_{q \in Q} V_q(x, a) < -\epsilon_Q \right) 
\end{array} \right\}$$

(6)

Since the form $V(x, a)$ is linear over the parameters in $a$, the above Equation is a typical case that can be solved by the CEGIS framework described above.
4.2 Adopting CEGIS to Real Arithmetics

We now briefly discuss the termination of the CEGIS procedure. We noted that termination is possible if a solution of the desired form in (3) exists, or alternatively, the C-space is exhausted. However, neither situation may result and the algorithm may run forever. In this section, we provide a simple strengthening of Eq. (4) that guarantees termination. We strengthen (4) as follows (when \( x_i \in R_j \)):

\[
\psi_{i,j} : \forall_k F_{j,k}(x_i, c) < -\epsilon_{T_j}
\]

wherein \( \epsilon_{T_j} > 0 \) are positive constants. The two constraints are identical when \( \epsilon_{T_j} = 0 \). Let \( c_i \) be a candidate examined at the \( i^{th} \) iteration of the CEGIS procedure modified to use Eq. (7). Suppose there exists a counter example \( x_i \), corresponding to \( c_i \). We compute a new refined C-space \( C_{i+1} \). It is easily shown the \( c_i \notin C_{i+1} \). Furthermore, by using (7), we obtain the following result that any candidate in a \( \eta \)-ball around \( c_i \) is also eliminated.

**Theorem 2** If the CEGIS procedure were modified using Eq. (7) with a given \( \epsilon_{T_j} > 0 \), then there exists a constant \( \eta > 0 \) such that at each iteration \( i \), \( B_\eta(c_i) \cap C_{i+1} = \emptyset \).

As a result, starting from an initial set \( C_0 \), given \( C_0 \) is a compact set, we note that employing the stronger rule (7) guarantees that at each step, an \( \eta \)-ball around the current solution is also removed. Thus, either a CLF is found or the C-space is empty in finitely many iterations. If we exhaust the C-space for a given values of \( \epsilon_{T_j} \), it is possible to repeat the search by halving \( \epsilon_{T_j} \) to alleviate against the loss of possible solutions due to the strengthening of Eq. (4) by (7).

**Faster Termination**

A first cut application of the CEGIS approach, presented thus far, resulted in a prohibitively large number of witnesses, failing on most of our benchmarks. This happens because the witnesses and candidate functions returned by the SMT solvers are similar (close in term of Euclidean distance). We discuss a heuristic to select witnesses \( x_i \) at each step of the CEGIS procedure, that led to the successful implementation of the overall procedure.

Given a current candidate \( c_i \), we may split the search for a witness into \( m \) parts: find a witness that violates the \( \bigvee_k F_{j,k}(x, c_i) < 0 \) (for each \( 1 \leq j \leq m \)). We will search for a counterexample that produces the “most-egregious” violation of the constraints possible. Therefore, we wish to maximize \( \min_k F_{j,k}(x, c_i) \). However, solvers such as dReal currently lack the ability to optimize. Therefore, we simply fix a constant \( \gamma \) and search for \( x_i \) satisfying \( \bigwedge_k F_{j,k}(x, c_i) - \gamma \geq 0 \). A larger \( \gamma \) leads to a more “egregious” violation and a larger set of candidates ruled out in the C-space and it is less likely to find a candidate that is similar to the previously selected candidate. The parameter \( \gamma \) itself is iteratively reduced to find a witness or conclude that no witness exists when \( \gamma = 0 \).

Also, the method is considerably improved by seeding with an initial set of points \( X_0 \).

4.3 Complexity and Incompleteness

There are many sources of incompleteness: (a) The polynomial template on the CLF with a maximum degree; (b) The use of \( \epsilon_{T_j} \) in Eq. (7) and finally (c) the use of a \( \delta \)-satisfiability solver for nonlinear constraints. However, it is possible to reduce this incompleteness by making \( \delta \) smaller.

In terms of complexity, solving linear arithmetic constraints and quantifier free nonlinear constraints are well-known to be NP-hard. In addition, while it is guaranteed that there will be a finite number of iterations in the CEGIS procedure, this number can be really large. Though we provided some heuristics to decrease the number of iterations, the worst case can be in the order of \( O(d^m) \), where \( m \) is the number of unknown coefficients in the template and \( d \) is a function of \( L_i, U_i \) and \( \eta \) in Theorem 2.

5 Evaluation and Discussion

Our approach was implemented as a Python script that wraps around the Z3 [4] and dReal [5] solvers. For the implementation, by a change of bases we make \( 0 \in G \) and we assumed \( I \) and \( G \) are balls with radiuses
σ_I and σ_G respectively (I : B_{σ_I}(0), G : B_{σ_G}(0)). Also, to reduce the number of unknowns to be discovered in our formula, value of ε_Q is given for each problem instance and β is fixed (e.g., β = 1).

The inputs to our procedure include a description of the plant model, the set S (taken to be a box), σ_G and σ_I. Our approach uses positive constants ε_{T_1} = ε_{T_2} for first two inequalities and ε_{T_3} for third one in Eq. [4]. Finally, we assume a quadratic CLF template in all cases with default intervals for the template coefficients A in the form of \( \prod_i [L_i, U_i] \), that can be modified by the user. This way, we make sure the procedure terminates since initial set C_0 is compact.

**Benchmarks** We collected 20 benchmark instances that are used in our evaluation. These benchmarks are taken from many sources and adapted to produce problem instances for our evaluation. We also provided a preliminary comparison with three implementations: the PESSOA tool by Mazo et al. [19], the CoSyMa tool by Camara et al. [20, 36] and the prototype corresponding to the recent work by Nilsson et al. [21]. The implementations for the other related approaches could not be obtained. Each of the compared approaches uses finite abstraction to solve synthesis problem.

Unfortunately, just 9 out of the 20 benchmarks could be successfully compared. Reasons included the lack of support for some required features, implementation issues and the lack of proper documentation. Therefore, the comparisons are restricted to the 9 cases that either (a) ran successfully on our machines, or (b) instances whose results/running time were reported in the corresponding references. Table 2 shows

### Table 1: Results of running our implementation on the benchmark suite.

*Legend: n: # state variables, |Q|: # modes, \( δ \): dReal precision, itr : # iterations, time: total computation time, Z3 T: time taken by Z3, dReal T: time taken by dReal, ⚫: Proper Radial CLF Found, ⚫: Failed. All timings are in seconds.*

| ID | n | q | \( ε_{T_1} \) | \( ε_{Q} \) | \( ε_{T_3} \) | \( δ \) | itr | Z3 T | dReal T | Tot. Time | Status |
|----|---|---|----|----|----|---|---|----|----|--------|---------|
| 1  | 2 | 2 | 0.1 | 0.01 | 0.01 | 10^{-8} | 9 | 0.2 | 12.3 | 12.7 | ✓        |
| 2  | 2 | 2 | 0.1 | 0.0001 | 0.1 | 10^{-9} | 15 | 0.5 | 7.1  | 8.0  | ✓        |
| 3  | 2 | 2 | 0.1 | 0.001 | 0.001 | 10^{-3} | 3  | 0.0 | 78.4 | 78.6 | ✓        |
| 4  | 2 | 2 | 0.1 | 0.001 | 0.001 | 10^{-8} | 10 | 0.1 | 7.0  | 7.6  | ✓        |
| 5  | 2 | 5 | 0.1 | 0.0001 | 0.0001 | 10^{-8} | 1  | 0.0 | 1.1  | 1.1  | ✓        |
| 6  | 2 | 2 | 0.1 | 0.001 | 0.001 | 10^{-8} | 4  | 0.0 | 4.7  | 5.0  | ✓        |
| 7  | 2 | 3 | 0.1 | 0.001 | 0.001 | 10^{-3} | 9  | 0.0 | 47.7 | 48.4 | ✓        |
| 8  | 2 | 2 | 0.1 | 0.001 | 0.001 | 10^{-8} | 5  | 0.0 | 2.6  | 2.9  | ✓        |
| 9  | 2 | 2 | 0.2 | 0.0001 | 0.0001 | 10^{-9} | 6  | 0.1 | 1.5  | 1.9  | ✓        |
| 10 | 3 | 4 | 0.1 | 0.001 | 0.001 | 10^{-3} | 1  | 0.0 | 2.8  | 2.8  | ✓        |
| 11 | 4 | 5 | 0.1 | 0.001 | 0.001 | 10^{-3} | 1  | 0.0 | 27.8 | 27.8 | ✓        |
| 12 | 5 | 6 | 0.1 | 0.001 | 0.001 | 10^{-3} | 1  | 0.0 | 704  | 704  | ✓        |
| 13 | 6 | 4 | 0.1 | 0.001 | 0.001 | 10^{-3} | 2  | 0.5 | 2994 | 2995.6 | ✓        |
| 14 | 2 | 3 | 0.2 | 0.001 | 0.1  | 10^{-8} | 8  | 0.5 | 6.6  | 7.5  | ✓        |
| 15 | 3 | 3 | 0.1 | 0.0001 | 0.01 | 10^{-8} | 15 | 29  | 80   | 110.5 | ✓        |
| 16 | 3 | 5 | 0.1 | 0.0001 | 0.1  | 10^{-8} | 8  | 8.9 | 43.8 | 53.8 | ✓        |
| 17 | 3 | 2 | 0.5 | 0.001 | 0.5  | 10^{-8} | 17 | 64.2 | 132.2 | 197.8 | ✓        |
| 18 | 3 | 2 | 0.1 | 0.0001 | 0.2  | 10^{-8} | 16 | 2.0 | 73.4 | 76.7 | ✓        |
| 19 | 4 | 2 | 0.1 | 0.0001 | 0.01 | 10^{-3} | 4  |    | – >1hr – |        |          |
| 20 | 4 | 2 | 0.1 | 0.001 | 0.1  | 10^{-6} | 36 |    | – >1hr – |        | ⚫        |
Table 2: Preliminary comparison of running times our technique against 3 different related implementations.

| ID | CEGIS Time | Tool | Tool Time | Rem. |
|----|------------|------|-----------|------|
| 6  | 5.0        | PESSOA | 42.846    | (r1) |
| 7  | 48.4       | PESSOA | 6881.1    | (r1) |
| 9  | 1.9        | CoSyMA | 3.2       | (r2) |
| 10 | 2.8        | CoSyMA | 1.8       | (r1) |
| 11 | 27.8       | CoSyMA | 494.0     | (r3) |
| 12 | 704.1      | CoSyMA | 571.0     | (r3) |
| 13 | 2995.6     | CoSyMA | out of memory | (r3) |
| 14 | 7.5        | [21]  | -         | (r4) |
| 18 | 76.7       | [21]  | 5319.5    | see Fig. 2 |

(r1): Parameters as reported in related works. (r2): Parameters: $N = 2, \tau = 0.1, \eta = 0.008$. Controllability Ratio 47.2% (r3): Reproduced with out of memory exception (timings as reported in [36]) (r4): Time not reported. no implementation is available.

![Figure 2: Region $W$ (red) is shown for our method and winning region after 350 iterations (blue) is shown for the CEGAR-based approach of Nilsson et al. [21].](image-url)

While our approach performs better, it has some shortcomings for now and these issues should be addressed in future works. First, our method only works for RWS properties and systems without disturbances. Second, choose of constants $\epsilon_{T_j}$ are not straightforward. Choosing $\epsilon_{T_1} (= \epsilon_{T_2})$ depend on $\beta$ and $\epsilon_{T_3}$ depends on $\epsilon_Q$. As a rule of thumb, we chose $\epsilon_{T_1} = \epsilon_{T_2} = 0.1\beta$ and $\epsilon_{T_3} = \epsilon_Q$ initially. If the procedure timed out, we increase values of $\epsilon_{T_j}$ iteratively. Alternatively, one can start from large values for $\epsilon_{T_j}$ and if procedure terminates with no solution, these values can be decreases. However, to make the procedure is completely automatic, more investigation on this part is needed as proper value for dReal precision also depend on these values.

5.1 Simulation

In this section we consider one of the examples for simulation. The example is inverted pendulum on a cart. The problem is to keep the pendulum near vertical position. The dynamics of the system is described by the ODEs $\dot{\theta} = \omega$, $\dot{\omega} = \frac{g}{l} \sin(\theta) - \frac{h}{ml^2} \omega + \frac{1}{ml^2} \cos(\theta) u$, where $g = 9.8, h = 2, l = 2$ and $m = 0.5$. We used Taylor expansion to approximate the trigonometric function and also the input $u$ is discretized to be in set $\{-30, 30\}$. Considering region $S = [-1.5, 1.5] \times [-4, 4], \sigma_I = 0.5, \sigma_G = 0.2, \epsilon_Q = 0.05, \epsilon_{T_1} = 0.1$ and $\epsilon_{T_3} = 0.05$, the following CLF is found $V([\theta, \omega]^T) = 0.65625y^2 + 0.69043xy + 2.2539x^2$. Notice that, to make
region $W$ bigger, one can find some $\beta > 1$ s.t. $V$ is still a CLF. In this example, $\beta = 4.5$ gives a bigger region $W$. In order to implement the controller, we need a minimal dwell time $\delta_m$ for switching strategy for reaching $G$. Such $\delta_m$ exists by Theorem 1 and we can find a lower-bound for $\delta_m$ from the expression derived in its proof. However, this lower-bound is quite conservative which results in needlessly fast switching.

Simulating to find minimal dwell time: We can find a better bound on $\delta_m$ through simulation. We implement the plant and the controller in a Simulink diagram in MATLAB and simulate the system until the continuous state reaches $G$. While in practice, closed loop system satisfies RWS property with big $\delta_m$ (e.g. $\delta_m = 0.02s$), the $\dot{t}V$ is not negative all the times. To make sure $\dot{t}V$ is negative for all time instances, we used $\delta_m = 0.00002s$ and a simulation for initial state $[\theta, \omega] = [1 - 2]$ is shown in Figure 3.

6 Conclusion

We have demonstrated a CEGIS procedure for synthesizing CLFs for switched systems to ensure RWS. Our approach ensures that the resulting controllers satisfy a minimal dwell time requirement. Through a prototype implementation, we have shown some preliminary evidence of applicability to small examples. However, even examples as large as 4 or 5 variables pose a challenge. Moving forward, we are exploring the use of relaxations such as sum-of-squares (SOS) programming and Bernstein polynomials while ascribing witnesses when the formula turns out to be satisfiable. Finally, nonlinear solvers remain an important and promising area of research where solvers such as dReal have made a real impact. We are hoping that as such solvers mature, our technique can begin to address larger and more challenging examples.

References

[1] A. Solar-Lezama, “Program synthesis by sketching,” Ph.D. dissertation, 2008.

[2] A. Solar-Lezama, L. Tancau, R. Bodik, S. Seshia, and V. Saraswat, “Combinatorial sketching for finite programs,” in ACM Sigplan Notices, vol. 41, no. 11. ACM, 2006, pp. 404–415.

[3] R. Alur, R. Bodik, G. Juniwal, M. M. Martin, M. Raghothaman, S. A. Seshia, R. Singh, A. Solar-Lezama, E. Torlak, and A. Udupa, “Syntax-guided synthesis,” in Formal Methods in Computer-Aided Design (FMCAD), 2013. IEEE, 2013, pp. 1–17.
[4] L. de Moura and N. Bjørner, “Z3: An efficient SMT solver,” in TACAS, ser. LNCS, vol. 4963. Springer, 2008, pp. 337–340.

[5] S. Gao, S. Kong, and E. M. Clarke, “dreal: An SMT solver for nonlinear theories over the reals,” in Automated Deduction - CADE-24 - 24th International Conference on Automated Deduction, Lake Placid, NY, USA, June 9-14, 2013. Proceedings, 2013, pp. 208–214.

[6] P. A. Parillo, “Semidefinite programming relaxation for semialgebraic problems,” Mathematical Programming Ser. B, vol. 96, no. 2, pp. 293–320, 2003.

[7] A. Papachristodoulou and S. Prajna, “On the construction of lyapunov functions using the sum of squares decomposition,” in Decision and Control, 2002, Proceedings of the 41st IEEE Conference on, vol. 3. IEEE, 2002, pp. 3482–3487.

[8] B. Tibken, “Estimation of the domain of attraction for polynomial systems via lmis,” in IEEE CDC, vol. 4. IEEE Press, 2000, pp. 3860–3864 vol.4.

[9] Z. Artstein, “Stabilization with relaxed controls,” Nonlinear Analysis: Theory, Methods & Applications, vol. 7, no. 11, pp. 1163–1173, 1983.

[10] D. Henrion, J. Lofberg, M. Kocvara, and M. Stingl, “Solving polynomial static output feedback problems with penbmi,” in Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC’05. 44th IEEE Conference on. IEEE, 2005, pp. 7581–7586.

[11] L. El Ghaoui and V. Balakrishnan, “Synthesis of fixed-structure controllers via numerical optimization,” in Decision and Control, 1994., Proceedings of the 33rd IEEE Conference on, vol. 3. IEEE, 1994, pp. 2678–2683.

[12] S. Gaubert, E. Goubault, A. Taly, and S. Zennou, “Static analysis by policy iteration on relational domains,” in Programming Languages and Systems. Springer, 2007, pp. 237–252.

[13] W. Tan and A. Packard, “Searching for control lyapunov functions using sums of squares programming,” in Allerton Conference, 2004, pp. 210–219.

[14] L. Rifford, “Existence of lipschitz and semiconcave control-lyapunov functions,” SIAM Journal on Control and Optimization, vol. 39, no. 4, pp. 1043–1064, 2000.

[15] D. Liberzon, Switching in systems and control. Springer, 2003.

[16] H. Lin and P. J. Antsaklis, “Stability and stabilizability of switched linear systems: a survey of recent results,” Automatic control, IEEE Transactions on, vol. 54, no. 2, pp. 308–322, 2009.

[17] H. Lin and P. Antsaklis, “Hybrid dynamical systems: An introduction to control and verification,” Found. Trends Syst. Control, vol. 1, no. 1.

[18] A. Taly and A. Tiwari, “Switching logic synthesis for reachability,” in Proceedings of the tenth ACM international conference on Embedded software. ACM, 2010, pp. 19–28.

[19] J. Mazo, Manuel, A. Davitian, and P. Tabuada, “PESSOA: a tool for embedded controller synthesis,” in CAV, ser. LNCS. Springer, 2010, vol. 6174, pp. 566–569.

[20] J. Cámara, A. Girard, and G. Gössler, “Synthesis of switching controllers using approximately bisimilar multiscale abstractions,” in Proceedings of the 14th international conference on Hybrid systems: computation and control. ACM, 2011, pp. 191–200.

[21] P. Nilsson and N. Ozay, “Incremental synthesis of switching protocols via abstraction refinement,” in Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on. IEEE, 2014.

[22] H. Lin and P. J. Antsaklis, “Switching stabilizability for continuous-time uncertain switched linear systems,” Automatic Control, IEEE Transactions on, vol. 52, no. 4, pp. 633–646, 2007.
[23] J. C. Geromel and P. Colaneri, “Stability and stabilization of continuous-time switched linear systems,” *SIAM Journal on Control and Optimization*, vol. 45, no. 5, pp. 1915–1930, 2006.

[24] E. Asarin, O. Bournez, T. Dang, O. Maler, and A. Puueldi, “Effective synthesis of switching controllers for linear systems,” *Proceedings of the IEEE*, vol. 88, no. 7, pp. 1011–1025, 2000.

[25] S. Itzhaky, S. Gulwani, N. Immerman, and M. Sagiv, “A simple inductive synthesis methodology and its applications,” in *ACM Sigplan Notices*, vol. 45, no. 10. ACM, 2010, pp. 36–46.

[26] S. Gulwani, S. Jha, A. Tiwari, and R. Venkatesan, “Synthesis of loop-free programs,” in *ACM SIGPLAN Notices*, vol. 46, no. 6. ACM, 2011, pp. 62–73.

[27] G. Frehse, S. K. Jha, and B. H. Krogh, “A counterexample-guided approach to parameter synthesis for linear hybrid automata,” in *Hybrid Systems: Computation and Control*. Springer, 2008, pp. 187–200.

[28] B. Yordanov and C. Belta, “Parameter synthesis for piecewise affine systems from temporal logic specifications,” in *Hybrid Systems: Computation and Control*. Springer, 2008, pp. 542–555.

[29] U. Topcu, A. Packard, P. Seiler, and T. Wheeler, “Stability region analysis using simulations and sum-of-squares programming,” in *Proc. ACC*. IEEE Press, 2007, pp. 6009–6014.

[30] J. Kapinski, J. V. Deshmukh, S. Sankaranarayanan, and N. Arevichia, “Simulation-guided lyapunov analysis for hybrid dynamical systems,” in *Proceedings of the 17th international conference on Hybrid systems: computation and control*. ACM, 2014, pp. 133–142.

[31] D. Liberzon and A. S. Morse, “Basic problems in stability and design of switched systems,” *Control Systems, IEEE*, vol. 19, no. 5, pp. 59–70, 1999.

[32] W. Perruquetii, J. Richard, and P. Borne, “Lyapunov analysis of sliding motions: Application to bounded control,” *Mathematical Problems in Engineering*, vol. 3, no. 1, pp. 1–25, 1996.

[33] S. Saat, M. Krug, and S. K. Nguang, “A nonlinear static output controller design for polynomial systems: An iterative sums of squares approach,” in *Mechatronics (ICOM), 2011 4th International Conference On*. IEEE, 2011, pp. 1–6.

[34] L. Greco, “Stability and stabilization issues in switched systems,” Ph.D. dissertation, 2005.

[35] UCLA CyPhy Laboratory, “PESSOA: Software toolbox for the synthesis of correct-by-design embedded control software.”

[36] S. Mouelhi, A. Girard, and G. Gössler, “Cosyma: a tool for controller synthesis using multi-scale abstractions,” in *Proceedings of the 16th international conference on Hybrid systems: computation and control*. ACM, 2013, pp. 83–88.

[37] E. A. Gol, X. Ding, M. Lazar, and C. Belta, “Finite bisimulations for switched linear systems,” in *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*. IEEE, 2012, pp. 7632–7637.

[38] ETH Zurich - Automatic Control Laboratory, “Lyapunov based control,” Sep 2014. [Online]. Available: [http://control.ee.ethz.ch/~apnoco/Lectures2014/096]-Lyapunov based control and Backstepping.pdf

[39] S. Pettersson and B. Lemnartson, “Stabilization of hybrid systems using a min-projection strategy,” in *American Control Conference, 2001. Proceedings of the 2001, vol. 1*. IEEE, 2001, pp. 223–228.

[40] T. X. Nghiem, G. J. Pappas, and R. Mangharam, “Event-based green scheduling of radiant systems in buildings,” in *American Control Conference (ACC), 2013*. IEEE, 2013, pp. 455–460.

[41] W. Zhang, A. Abate, J. Hu, and M. P. Vitus, “Exponential stabilization of discrete-time switched linear systems,” *Automatica*, vol. 45, no. 11, pp. 2526–2536, 2009.

[42] L. Faubourg and J.-B. Pomet, “Design of control lyapunov functions for homogeneous jurdjevic-quinn systems,” 1999.
A Proofs

Theorem 3 Given a \( \lambda > 1 \), there exists a minimum dwell time \( \delta_{m,q} > 0 \) such that if \( \text{tr}_X(t) \in S \setminus G \) and \( \dot{V}_q(\text{tr}_X(t)) \leq -\epsilon_Q \) (i.e., the controller is in the Reside state) and \( \text{tr}_X(\tau) \in S \setminus G \) for all \( \tau \in [t, t + \delta_{m,q}] \), then

\[
(\forall \tau \in [t, t + \delta_{m,q}]) \quad \dot{V}_q(\text{tr}_X(\tau)) < -\frac{\epsilon_Q}{\lambda}
\]

Proof of Theorem 3 A constructive proof is provided here. Assume at time \( t \) s.t.

\[
\dot{V}_q(\text{tr}_X(t)) \leq -\epsilon_Q
\]

and let \( t + \delta \) (\( \delta > 0 \)) be the minimum time where

\[
\dot{V}_q(\text{tr}_X(t + \delta)) \geq -\frac{\epsilon_Q}{\lambda}
\]

if

\[
(\forall \tau \in [t, t + \delta]) \quad \text{tr}_Q(\tau) = q \land \text{tr}_X(\tau) \in S \setminus G
\]

Let \( \tilde{V}_q : X \rightarrow \mathbb{R} \) be

\[
\tilde{V}_q = \frac{d\dot{V}_q}{dx} f_q(x)
\]

Since \( S \setminus G \) is a bounded set and \( \tilde{V}_q \) is a polynomial, there exist \( \epsilon_1 > 0 \) s.t.

\[
(\forall x \in S \setminus G) \quad \tilde{V}_q(x) \leq \epsilon_1
\]

Also

\[
\dot{V}_q(\text{tr}_X(t + \delta)) = \tilde{V}_q(\text{tr}_X(t)) + \int_t^{t+\delta} \tilde{V}_q(\text{tr}_X(\tau)) d\tau
\]

\[
\implies \dot{V}_q(\text{tr}_X(t)) + \epsilon_1 \delta \leq \dot{V}_q(\text{tr}_X(t)) + \epsilon_1 \delta
\]

Then, we can conclude

\[
-\frac{\epsilon_Q}{\lambda} \leq -\epsilon_Q + \epsilon_1 \delta \implies \frac{(\lambda - 1)\epsilon_Q}{\epsilon_1 \lambda} \leq \delta
\]

The above arguments suggest that if \( \dot{V}_q(\text{tr}_X(t)) \leq -\epsilon_Q \) for a mode \( q \), then without switching there exists \( \delta_{m,q} > 0 \) (\( \delta \) in above argument) s.t.

\[
(\forall \tau \in [t, t + \delta_{m,q}]) \quad \dot{V}_q(\text{tr}_X(\tau)) < -\frac{\epsilon_Q}{\lambda}
\]

Proof of Theorem 1 Given compact sets \( S, G, I \subseteq \text{int}(S) \), a plant \( \Psi \) and a non-zeno CLF \( V(x) \) w.r.t \( (I, G, S) \) with associated region \( W \), there exists a class of switching functions such that the closed loop \( \Phi \) satisfies the following properties.

1. System satisfies RWS w.r.t \( (W, G, S) \).
2. All the traces of the closed loop system starting from \( W \) are time divergent before reaching \( G \).

First, we show that \( I \setminus G \subseteq \text{int}(W) \) and \( W \setminus G \subseteq \text{int}(S) \). Remember \( W : \{x | V(x) \leq \beta\} \cap S \). By definition \( W \setminus G \subseteq S \) and \( (\forall x \in \partial S) V(x) > \beta \). Therefore, \( W \subseteq \text{int}(S) \). Also, \( (\forall x \in I \setminus G) V(x) < \beta \) and consequently \( I \setminus G \subseteq \text{int}(W) \).

Let the controller confirms to the relation described below for some \( \lambda > 1 \)

\[
\text{switch}(q, x) := \begin{cases} q & \text{if } \dot{V}_q(x) \geq \frac{\epsilon_Q}{\lambda} \land x \in S \setminus G \\ \dot{q} & \text{otherwise} \end{cases}
\]

(10)
According to Theorem 3 whenever there is a switch to a mode \( q \), then the output of the controller remains \( q \) for at least \( \delta_{m,q} \) time unit unless trace reaches \( G \) or leaves \( S \).

Now we show that \( (\text{tr}_{X} \in W) \implies (\text{tr}_{X} \in W) \cup (\text{tr}_{X} \in G) \).

Assume that \( \text{tr}_{X}(0) \in (W \setminus G) \). \( \text{tr}_{X} \) can not leave \( S \) before reaching \( \partial W \) or \( G \). Let \( \tau \geq 0 \) be the first time that \( \text{tr}_{X}(\tau) \) reaches \( \partial W \) before reaching \( G \). Then \( \text{tr}_{V}(\tau) = 0 \). Since \( \text{tr}_{V}(0) \leq \beta \) and \((\forall 0 \leq t < \tau) \text{tr}_{V}(t) < 0, \ \tau \) has to be zero. At start \( (t = 0) \), since \( \text{tr}_{X}(0) \in (W \setminus G) \), then the controller choose a mode \( q \) such that \( \text{tr}_{V}(0) < 0 \) and therefore, the trace leaves the boundary of \( W \) and the trace goes to the interior of \( W \) (int(W)).

If \((\forall t > 0) \text{tr}_{X}(t) \in W \setminus G \), by the construction of the controller, we can conclude time diverges. Also since \( \text{tr}_{V}(t) < -\frac{\partial W}{X} \) for all times, \( \text{tr}_{V} \) decreases to infinity. However, the value of \( V \) is bounded on compact set \( W \setminus G \). Therefore, \( \text{tr}_{X} \) can not remain in \( W \setminus G \) and can not reach the boundary of \( W \). The only possible outcome for the trace is to reach \( G \). System satisfies RWS w.r.t \((W,G,W)\) and therefore it satisfies RWS w.r.t \((W,G,S)\).

**Proof of Theorem 2** If the CEGIS procedure were modified using Eq. (1) with a given \( \epsilon_{T_{j}} > 0 \), then there exists a constant \( \eta > 0 \) such that at each iteration \( i \), \( B_{\eta}(c_{i}) \cap C_{i+1} = \emptyset \).

Given a counter example \( x_{i} \in R_{j} \) for \( c_{i} \), the following is true

\[ \land_{k} F_{j,k}(x_{i},c_{i}) \geq 0 \] (11)

and the added restriction on \( C\)-Space for next iteration is

\[ \lor_{k} F_{j,k}(x_{i},c_{i}) < -\epsilon_{T_{j}} \] (12)

Let \( F'_{j,k} \) be polynomial \( F_{j,k} \) without monomials which do not have variables in \( c \). Since \( F'_{j,k} \) is a polynomial in \( x \) and linear in \( c \), and \( R_{j} \) is a compact set, there exists \( M_{j,k} > 0 \) s.t. \( (\forall c \in C_{0}) (\forall x \in R_{j}) F'_{j,k}(x,c) \geq -M_{j,k} \) and as a result, there exists a \( \eta > 0 \) s.t.

\[ (\forall c \in B_{\eta}(0)) (\forall x \in R_{j})(\forall k) F'_{j,k}(x,c) \geq -\epsilon_{T_{j}} \]

Also,

\[ (\forall c \in B_{\eta}(c_{i})) (\forall k) F_{j,k}(x_{i},c_{i}) = F_{j,k}(x_{i},c_{i}) + F'_{j,k}(x_{i},c_{i}) \] (13)

where \( c_{\eta} \in B_{\eta}(0) \). Therefore,

\[ (\forall c \in B_{\eta}(c_{i})) \land_{k} F_{j,k}(x_{i},c) \geq -\epsilon_{T_{j}} \]

and by comparing this to Equation (12) it is easy to see that \( x_{i} \) is a counter example for \( c \in B_{\eta}(c_{i}) \). and \( B_{\eta}(c_{i}) \cap C_{i+1} = \emptyset \).

**B Benchmark**

The benchmark used in the experiments are examples adopted from literature. We consider each of these systems as a switched system with RWS as the specification.

**System 1** This system is adopted from [32]. There are two continuous variables \( x \) and \( y \) and the dynamics are

\[ \dot{x} = y \]
\[ \dot{y} = -x + u \]

, where \( u \in \{-1, 1\} \). \( S = [-5, 5]^2 \), \( \sigma_{I} = 2.5 \) and \( \sigma_{G} = 0.5 \).

**System 2** This system is adopted from [32]. There are two continuous variables \( x \) and \( y \) and the dynamics are

\[ \dot{x} = u \]
\[ \dot{y} = y^2 x \]

, where \( u \in \{-4, 4\} \). And region \( S = [-1, 1]^2 \), \( \sigma_{I} = 0.5 \) and \( \sigma_{G} = 0.1 \).
System 3 This system is also adopted from [32]. There are two continuous variables \( x \) and \( y \) and the dynamics are
\[
\begin{align*}
\dot{x} &= -x(0.1 + (x + y)^2) \\
\dot{y} &= (u + x)(0.1 + (x + y)^2)
\end{align*}
\]
, where \( u \in \{-1,1\} \). The region is \( S = [-5 \ 5]^2 \), \( \sigma_I = 2.5 \) and \( \sigma_G = 0.5 \).

System 4 This system is adopted from [38]. There are two continuous variables \( x \) and \( y \) and the dynamics are
\[
\begin{align*}
\dot{x} &= y - x^3 \\
\dot{y} &= u
\end{align*}
\]
, where \( u \in \{-1,1\} \). The region of interest is \( S = [-10 \ 10]^2 \), \( \sigma_I = 5 \) and \( \sigma_G = 0.5 \).

System 5 This system is a switched system adopted from [34]. There are two continuous variables \( x \) and \( y \) and 5 modes \( (q_1,\ldots,q_5) \) the dynamics of each mode is described below
\[
\begin{align*}
q_1 \quad &\begin{cases} 
\dot{x} = 0.0403x + 0.5689y \\
\dot{y} = 0.6771x - 0.2556y
\end{cases} \\
q_2 \quad &\begin{cases} 
\dot{x} = 0.2617x - 0.2747y \\
\dot{y} = 1.2134x - 0.1331y
\end{cases} \\
q_3 \quad &\begin{cases} 
\dot{x} = 1.4725x - 1.2173y \\
\dot{y} = 0.0557x - 0.0412y
\end{cases} \\
q_4 \quad &\begin{cases} 
\dot{x} = -0.5217x + 0.8701y \\
\dot{y} = -1.4320x + 0.8075y
\end{cases} \\
q_5 \quad &\begin{cases} 
\dot{x} = -2.1707x - 1.0106y \\
\dot{y} = -0.0592x + 0.6145y
\end{cases}
\end{align*}
\]
The region \( S \) is \( [-10 \ 10]^2 \), \( \sigma_I = 5 \) and \( \sigma_G = 0.5 \).

System 6 This system is adopted from [19] is a DC motor system. There are two continuous variables \( \omega \) and \( i \), and input \( u \) is the source voltage.
\[
\begin{align*}
\dot{\omega} &= -\frac{B}{J}\omega + \frac{k}{J}i \\
\dot{i} &= -\frac{k}{L}\omega - \frac{R}{L}i + \frac{1}{L}u
\end{align*}
\]
, where \( B = 10^{-4} \), \( J = 25 \times 10^{-5} \), \( k = 0.05 \), \( R = 0.5 \), \( L = 15 \times 10^{-4} \) and \( u \in \{-1,1\} \). The desired point is \( \begin{bmatrix} \omega \\ i \end{bmatrix} = [20 \ 0] \) and by change of basis, we get the following system
\[
\begin{align*}
\dot{\omega}' &= -\frac{B}{J}(\omega' + 20) + \frac{k}{J}i' \\
\dot{i}' &= -\frac{k}{L}(\omega' + 20) - \frac{R}{L}i' + \frac{1}{L}u
\end{align*}
\]
Region of interest \( S = \{[\omega \ i]^T|\omega \in [-10 \ 10], i \in [-10 \ 10]\} \), \( \sigma_I = 4 \) and \( \sigma_G = 0.5 \).
System 7 This system is adopted from [35] is a model of inverted pendulum on a cart. There are two continuous variables $\theta$ (angular position) and $\omega$ (angular velocity), and input $u$ is the applied force to the cart.

$$\dot{\theta} = \omega$$
$$\dot{\omega} = \frac{g}{l} \sin(\theta) - \frac{h}{ml^2} \omega + \frac{1}{ml} \cos(\theta) u$$

, where $g = 9.8$, $h = 2$, $l = 2$, $m = 0.5$ and $u \in \{-30, 30\}$. The region is $S = \{[\theta \ \omega]^T | \theta \in [-1.5 \ 1.5], i \in [-4 \ 4]\}$, $\sigma_I = 0.5$ and $\sigma_G = 0.2$.

System 8 This system is adapted from Example 3.1 in [37]. The system has two discrete mode $(q_1, q_2)$ and two continuous variables $x_1$ and $x_2$ and the following dynamics

$$q_1 \begin{cases} 
  x_1 = -0.65x_1 + 0.32x_2 \\
  x_2 = -0.42x_1 - 0.92x_2 
\end{cases}$$

$$q_2 \begin{cases} 
  x_1 = 0.65x_1 + 0.32x_2 \\
  x_2 = -0.42x_1 - 0.92x_2 
\end{cases}$$

The region $S = [-10 \ 10]^2$, $\sigma_I = 5$ and $\sigma_G = 0.25$.

System 9 This system is a DCDC boost converter adopted from [20] with two discrete mode $(q_1, q_2)$, two continuous variables $i$ and $v$. By a simple change of bases the state $i = 1.35$ and $v = 5.65$ is set as desired point of activity (origin) and the following dynamics are obtained.

$$q_1 \begin{cases} 
  i = 0.0167i + 0.3558 \\
  \dot{i} = -0.0142v - 0.08023 
\end{cases}$$

$$q_2 \begin{cases} 
  i = -0.0183i - 0.0663v - 0.0660 \\
  \dot{i} = 0.0711 * i - 0.0142 * v + 0.0158 
\end{cases}$$

Region of interest is $S = \{[i \ v]^T | i \in [-0.7 \ 0.45], v \in [-0.7 \ 0.7]\}$, $\sigma_I = 0.15$ and $\sigma_G = 0.02$.

System 10 The system is a heater for keeping several rooms warm [30]. There are 3 rooms $t_1$, $t_2$ and $t_3$ and heater can be in one of these room or it can be off. Therefore, there are four modes $(q_0, ..., q_3)$ with the following dynamics. The goal is to keep $t_i$ around 21 ($i \in \{1, 2, 3\}$).

$$q_0 \begin{cases} 
  \dot{t}_1 = 0.01(-10.5(t_1 + 21) + 5(t_2 + 21) + 5(t_3 + 21) + 5) \\
  \dot{t}_2 = 0.01(5(t_1 + 21) - 10.5(t_2 + 21) + 5(t_3 + 21) + 5) \\
  \dot{t}_3 = 0.01(5(t_1 + 21) + 5(t_2 + 21) - 10.5(t_3 + 21) + 5) 
\end{cases}$$

$$q_1 \begin{cases} 
  \dot{t}_1 = 0.01(-11.5(t_1 + 21) + 5(t_2 + 21) + 5(t_3 + 21) + 5) \\
  \dot{t}_2 = 0.01(5(t_1 + 21) - 10.5(t_2 + 21) + 5(t_3 + 21) + 5) \\
  \dot{t}_3 = 0.01(5(t_1 + 21) + 5(t_2 + 21) - 10.5(t_3 + 21) + 5) 
\end{cases}$$

$$q_2 \begin{cases} 
  \dot{t}_1 = 0.01(-10.5(t_1 + 21) + 5(t_2 + 21) + 5(t_3 + 21) + 5) \\
  \dot{t}_2 = 0.01(5(t_1 + 21) - 11.5(t_2 + 21) + 5(t_3 + 21) + 55) \\
  \dot{t}_3 = 0.01(5(t_1 + 21) + 5(t_2 + 21) - 10.5(t_3 + 21) + 5) 
\end{cases}$$

$$q_3 \begin{cases} 
  \dot{t}_1 = 0.01(-11.5(t_1 + 21) + 5(t_2 + 21) + 5(t_3 + 21) + 55) \\
  \dot{t}_2 = 0.01(5(t_1 + 21) - 11.5(t_2 + 21) + 5(t_3 + 21) + 55) \\
  \dot{t}_3 = 0.01(5(t_1 + 21) + 5(t_2 + 21) - 10.5(t_3 + 21) + 55) 
\end{cases}$$

Region $S = [-10 \ 10]^3$, $\sigma_I = 2.5$ and $\sigma_G = 1$. 
System 11 The system is similar to System 10 except that the number of dimensions is 4. See [36].

System 12 The system is similar to System 10 except that the number of dimensions is 5. See [36].

System 13 This system is a 6 variables version of System 10 and there are 6 rooms and 2 heaters and we only consider 4 modes. The heater is off for one mode \((q_0)\) and for mode \(q_1 (1 \leq i \leq 3)\), two heaters are on in rooms \(i\) and \(3+i\).

System 14 This system is adapted from [21]. There are two continuous variables \(x_1\) and \(x_2\) and the controller can choose between three different modes \((q_1, q_2)\). By setting \(x_1 = -0.75\) and \(x_2 = 1.75\) as the origin, the new dynamics for these modes are

\[
\begin{align*}
q_1: & \quad \dot{x}_1 = -x_2 - 1.5x_1 - 0.5x_1^3 \\
& \quad \dot{x}_2 = x_1 - x_2^2 + 2
\end{align*}
\]

\[
\begin{align*}
q_2: & \quad \dot{x}_1 = -x_2 - 1.5x_1 - 0.5x_1^3 \\
& \quad \dot{x}_2 = x_1 - x_2
\end{align*}
\]

\[
\begin{align*}
q_3: & \quad \dot{x}_1 = -x_2 - 1.5x_1 - 0.5x_1^3 + 2 \\
& \quad \dot{x}_2 = x_1 + 10
\end{align*}
\]

Region \(S\) is defined as \(S = \{[x_1, x_2]^T | x_1 \in [-2.25, 2.75], v \in [-3.25, 3.25]\} \), \(\sigma_I = 1.0\) and \(\sigma_G = 0.2\). Notice that this region is a little different from the one introduced in [21].

System 15 The system is a linear switched system, adapted from [39]. There are three continuous variables \(x, y, z\) in this system and the dynamics for 3 modes \((q_1, q_2, q_3)\) are

\[
\begin{align*}
q_1: & \quad \dot{x} = 1.8631x - 0.0053y + 0.9129z \\
& \quad \dot{y} = 0.2681x - 6.4962y + 0.0370z \\
& \quad \dot{z} = 2.2497x - 6.7180y + 1.6428z
\end{align*}
\]

\[
\begin{align*}
q_2: & \quad \dot{x} = -2.4311x - 5.1032y + 0.4565z \\
& \quad \dot{y} = -0.0869x + 0.0869y + 0.0185z \\
& \quad \dot{z} = 0.0369x - 5.9869y + 0.8214z
\end{align*}
\]

\[
\begin{align*}
q_3: & \quad \dot{x} = 0.0372x - 0.0821y - 2.7388z \\
& \quad \dot{y} = 0.1941x + 0.2904y - 0.1110z \\
& \quad \dot{z} = -1.0360x + 3.0486y - 4.9284z
\end{align*}
\]

Region \(S = [-1, 1]^3\), \(\sigma_I = 0.3\) and \(\sigma_G = 0.01\).

System 16 This system is a switched system adopted from [37]. There are three continuous variables \(x, y, z\) in this system and the dynamics for 3 modes \((q_1, q_2, q_3)\) are
$y, z$ and $5$ modes ($q_1, ..., q_5$) the dynamics of each mode is described below

\[
\begin{align*}
q_1: & \begin{cases}
\dot{x} = 0.1764x + 0.8192y - 0.3179z \\
\dot{y} = -1.8379x - 0.2346y - 0.7963z \\
\dot{z} = -1.5023x - 1.6316y + 0.6908z
\end{cases} \\
q_2: & \begin{cases}
\dot{x} = -0.0420x - 1.0286y + 0.6892z \\
\dot{y} = 0.3240x + 0.0994y + 1.8833z \\
\dot{z} = 0.5065x - 0.1164y + 0.3254z
\end{cases} \\
q_3: & \begin{cases}
\dot{x} = -0.0952x - 1.7313y + 0.3868z \\
\dot{y} = 0.0312x + 0.4788y + 0.0540z \\
\dot{z} = -0.6138x - 0.4478y - 0.4861z
\end{cases} \\
q_4: & \begin{cases}
\dot{x} = 0.2445x + 0.1338y + 1.1991z \\
\dot{y} = 0.7183x - 1.0062y - 2.5773z \\
\dot{z} = 0.1535x + 1.3065y - 2.0863z
\end{cases} \\
q_5: & \begin{cases}
\dot{x} = -1.4132x - 1.4928y - 0.3459z \\
\dot{y} = -0.5918x - 0.0867y + 0.9863z \\
\dot{z} = 0.5189x - 0.0126y + 0.6433z
\end{cases}
\]

Region $S = [-3, 3]^3$, $\sigma_I = 1$ and $\sigma_G = 0.1$.

**System 17** This system is adopted from [33]. There are three continuous variables $x, y, z$ and the dynamics are

\[
\begin{align*}
\dot{x} &= -10x + 10y + u \\
\dot{y} &= 28x - y - xz \\
\dot{z} &= xy - 2.6667z
\end{align*}
\]

, where $u \in \{-100, 100\}$. And region $S = [-5, 5]^3$, $\sigma_I = 1.2$ and $\sigma_G = 0.3$.

**System 18** This system is a radiant system in building adopted from [21, 40] which is a switched linear system with three continuous variables ($T_c$, $T_1$ and $T_2$) and two modes ($q_1, q_2$). By setting $T_c = 24$ and $T_1 = T_2 = 23$ as the new origin, the dynamics obtained are

\[
\begin{align*}
q_1: & \begin{cases}
\dot{T}_c = 2.25T_1 + 2.25T_2 - 9.26T_c - 14.54 \\
\dot{T}_1 = 2.85T_2 - 7.13T_1 + 4.04T_c + 4.04 \\
\dot{T}_2 = 2.85T_1 - 7.13T_2 + 4.04T_c + 4.04
\end{cases} \\
q_2: & \begin{cases}
\dot{T}_c = 2.25T_1 + 2.25T_2 - 4.5T_c + 4.5 \\
\dot{T}_1 = 2.85T_2 - 7.13T_1 + 4.04T_c + 4.04 \\
\dot{T}_2 = 2.85T_1 - 7.13T_2 + 4.04T_c + 4.04
\end{cases}
\]

Region $S = [-6, 6]^3$, $\sigma_I = 3$ and $\sigma_G = 1$.

**System 19** The original system is a switched control system with inputs from [44]. There are 4 variables ($w, x, y$ and $z$) and 4 original modes. After converting the discrete system into a continuous one, the dynamics are
\[
q_1 \begin{aligned}
\dot{w} &= -0.693 w - 1.099 x + 2.197 y + 3.296 z - 7.820 u \\
\dot{x} &= -1.792 x + 2.197 y + 4.394 z - 8.735 u \\
\dot{y} &= -1.097 x + 1.504 y + 2.197 z - 2.746 u \\
\dot{z} &= 0.406 z + 3.244 u
\end{aligned}
\]
\[
q_2 \begin{aligned}
\dot{w} &= -1.792 w - 1.099 x + 2.197 y + 6.696 u \\
\dot{x} &= 0.406 x - 2.197 y + 4.734 u \\
\dot{y} &= -0.693 y + 2.773 u \\
\dot{z} &= -2.197 w - 1.099 x + 2.197 y + 1.504 z + 4.263 u
\end{aligned}
\]
\[
q_3 \begin{aligned}
\dot{w} &= 0.406 w + 0.811 u \\
\dot{x} &= 1.099 w - 0.144 x + 0.549 y - 0.549 z + 1.910 u \\
\dot{y} &= 0.549 x - 0.144 y - 0.549 z + 3.871 u \\
\dot{z} &= 1.099 w - 0.693 z + 4.970 u
\end{aligned}
\]
\[
q_4 \begin{aligned}
\dot{w} &= -0.693 w + 2.000 x + 1.863 u \\
\dot{x} &= -0.693 x + 4.159 u \\
\dot{y} &= -0.693 y + 2.773 u \\
\dot{z} &= 4.000 x - 4.000 y - 0.693 z - 1.069 u
\end{aligned}
\]
, where \(u \in \{-1, 1\}\) and Region of interest is \(S = [-1, 1]^4, \sigma_I = 0.1\) and \(\sigma_G = 0.1\).

**System 20** This system is a Tora system and the equations are adopted from [42]. There are 4 variables in this system with the following dynamics

\[
\begin{aligned}
\dot{w} &= x \\
\dot{x} &= -w + 0.1 \sin(y) \\
\dot{y} &= z \\
\dot{z} &= u
\end{aligned}
\]
, where \(u \in [-10, 10]\) and region \(S = [-1, 1]^4, \sigma_I = 0.2\) and \(\sigma_G = 0.1\).