Synthesizing Switching Logic to Minimize Long-Run Cost

Susmit Kumar Jha
Sanjit A. Seshia
Ashish Tiwari

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2011-16
http://www.eecs.berkeley.edu/Pubs/TechRpts/2011/EECS-2011-16.html

March 4, 2011

Synthesizing Switching Logic to Minimize Long-Run Cost

Susmit Jha[1], Sanjit A. Seshia[1], and Ashish Tiwari[2]

[1] UC Berkeley (jha,sseshia@berkeley.edu)
[2] SRI International (tiwari@csl.sri.com)

Abstract. Given a multi-modal dynamical system, optimal switching logic synthesis involves generating the conditions for switching between the system modes such that the resulting hybrid system satisfies a quantitative specification. We formalize and solve the problem of optimal switching logic synthesis for quantitative specifications over long run behavior. Each trajectory of the system, and each state of the system, is associated with a cost. Our goal is to synthesize a system that minimizes this cost from each initial state. Our paper generalizes earlier work on synthesis for safety as safety specifications can be encoded as quantitative specifications. We present an approach for specifying quantitative measures using reward and penalty functions, and illustrate its effectiveness using several examples. We present an automated technique to synthesize switching logic for such quantitative measures. Our algorithm is based on reducing the synthesis problem to an unconstrained numerical optimization problem which can be solved by any off-the-shelf numerical optimization engines. We demonstrate the effectiveness of this approach with experimental results.

1 Introduction

One of the holy grails in the design of hybrid systems is to automatically synthesize models from safety and performance specifications. In general, automated synthesis is difficult to achieve, in part because synthesis often involves human insight and intuition, and in part because of system complexity. Nevertheless, in some contexts, it may be possible for automated tools to *complete partial designs* generated by a human designer, enabling the designer to efficiently explore the space of design choices whilst ensuring that the synthesized system meets its specification.

One such problem is to synthesize the mode switching logic for multi-modal dynamical systems (MDS). An MDS is a physical system (plant) that can operate in different modes. The dynamics of the plant in each mode is known. However, to achieve safe and efficient operation, it is often necessary to switch between the different operating modes. Designing correct and optimal switching logic can be tricky and tedious.

In this paper, we consider the problem of synthesizing the switching logic for an MDS so that the resulting system is optimal. We formulate a performance metric which can be used to specify quantitative objectives about long-term behavior of the system as minimizing some cost measure. Specifically, we formulate cost as penalty per unit reward motivated by similar cost measure in Economics. For a given initial

state, the optimal long-term behavior corresponds to a trajectory of infinite length with infinite number of mode switches which has minimum cost. So, discovering the optimal long-term behavior requires

- discovering this infinite chain of mode switches, and
- the switching states from one mode to another.

Thus, a naive approach would require searching the optimal behavior over unbounded set of parameters. We reduce this problem to optimization over bounded set of parameters representing the repetitive long-term behavior. The key insight is that the long-term cost is essentially the cost of the repetitive part of the behavior. We only require the user to provide the number of switches which are enough to reach the repetitive behavior from an initial state. The system stays in repetitive behavior after reaching it and hence, the user can pick any large number of switches which would be enough to reach the repetitive behavior. We consider the supersequence of all possible mode sequences with the given number of mode switches and use the times spent in each mode in this supersequence as the parameters for optimization. If the time spent in a particular mode is zero, the mode is taken out of the optimum mode sequence. The optimization problem is then formulated as an unconstrained numerical optimization problem which can be solved by off-the-shelf tools. Solving this optimization problem yields the time spent in each mode which can then be used to find the switching states. By pruning out modes in which zero time is spent, we also discover the optimum mode switching sequence. So, we discover both the optimum mode switching sequence and the switching states for each initial state. These switching states can then be combined for different initial states to yield the optimum switching logic for the hybrid system.

The contributions of this paper are as follows:

- We formalize the problem of synthesizing optimal switching logic by introducing the notion of long-run cost which needs to be minimized for optimality (Section 2).
- The synthesis problem requires optimization over trajectories and not just finite segments. We show how to reduce optimization over an infinite trajectory to an equivalent optimization over a bounded set of parameters representing the *limit* behavior. (Section 4);
- We present an algorithm to solve this optimization problem for a single initial state based on unconstrained numerical optimization (Section 5) and show how our technique can be applied with multiple initial states. Our algorithm makes no assumptions on the intra-mode continuous dynamics other than locally-Lipschitz continuity and relies only on the ability to accurately simulate the dynamics.

Experimental results demonstrate our approach on a range of examples, including a DC-DC Boost power converter circuit (Section 7).

2 Problem Definition

2.1 Multimodal and Hybrid Systems

We model a hybrid system as a combination of a multimodal dynamical system and a switching logic.

Definition 1. Multimodal Dynamical System (MDS). *A* multimodal dynamical system *is a tuple* $\langle Q, X, f, \texttt{Init}\rangle$*, where* $Q := \{1, \ldots, N\}$ *is a set of modes,* $X := \{x_1, \ldots, x_n\}$ *is a set of continuous variables,* $f : Q \times \mathbb{R}^X \mapsto \mathbb{R}^X$ *defines a vector field for each mode in* Q*, and* $\texttt{Init} \subseteq Q \times \mathbb{R}^X$ *is a set of initial states. The* state space *of such an MDS is* $Q \times \mathbb{R}^X$*. A function* $\mathbf{qx} : \mathbb{R}^+ \mapsto (Q \times \mathbb{R}^X)$ *is said to be a* trajectory *of this MDS with respect to a sequence* $t_1, t_2, \ldots$ *of* switching times *if*
(i) $\mathbf{qx}(0) \in \texttt{Init}$ *and*
(ii) for all i *and for all* t *such that* $t_i < t$ *and* $t < t_{i+1}$*, it is the case that* $\mathbf{q}(t) = \mathbf{q}(t_i)$ *and*

$$\frac{d\mathbf{x}(t)}{dt} = f(\mathbf{q}(t), \mathbf{x}(t)), \tag{1}$$

where $\mathbf{q}$ *and* $\mathbf{x}$ *denote the projection of* $\mathbf{qx}$ *into the mode and continuous state components. The function* $\mathbf{x}$ *is continuous. The* switching sequence is the sequence of modes $\mathbf{q}(t_1), \mathbf{q}(t_2), \ldots$.

If MDS is a multimodal dynamical system, then its semantics, denoted $[\![\texttt{MDS}]\!]$, is the set of its trajectories with respect to all possible switching time sequences.

Definition 2. Switching Logic (SwL). *A* switching logic *for a multimodal system* $\texttt{MDS} := \langle Q, X, f, \texttt{Init}\rangle$ *is a tuple* $\langle (\texttt{g}_{q_1 q_2})_{q_1, q_2 \in Q}\rangle$ *where* $\texttt{g}_{q_1 q_2} \subseteq \mathbb{R}^X$ *is the* guard *defining the switch from mode* q_1 *to mode* q_2*.*

Given a multimodal system and a switching logic, we can now define a hybrid system by considering only those trajectories of the multimodal system that are consistent with the switching logic.

Definition 3. Hybrid System (HS). *A* hybrid system *is a tuple* $\langle \texttt{MDS}, \texttt{SwL}\rangle$ *consisting of a multimodal system* $\texttt{MDS} := \langle Q, X, f, \texttt{Init}\rangle$ *and a switching logic* $\texttt{SwL} := \langle (\texttt{g}_{q_1 q_2})_{q_1, q_2 \in Q}\rangle$*. The* state space *of the hybrid system is the same as the state space of* MDS*. A function* $\mathbf{qx} : \mathbb{R}^+ \mapsto (Q \times \mathbb{R}^X)$ *is said to be a* trajectory *of this hybrid system if there is a a sequence* $t_1, t_2, \ldots$ *of* switching times *such that*
(a) $\mathbf{qx}$ *is a trajectory of* MDS *with respect to this switching time sequence and*
(b) setting $t_0 = 0$*, for all* t_i *in the switching time sequence with* $i \geq 1$*,* $\mathbf{x}(t_i) \in \texttt{g}_{\mathbf{q}(t_{i-1})\mathbf{q}(t_i)}$ *and for all* t *such that* $t_{i-1} < t < t_i$*,* $\mathbf{x}(t) \notin \cup_{q \in Q} \texttt{g}_{\mathbf{q}(t_{i-1})q}$*.*

Discrete jumps are taken as soon as they are enabled and they do not have non-identity resets. For the notion of a trajectory to be well-defined, guards are required to be closed sets. The semantics of a hybrid system HS, denoted $[\![\texttt{HS}]\!]$, is the collection of all its trajectories as defined above.

2.2 Quantitative Measures for Hybrid Systems

Our interest is in automatically synthesizing hybrid systems which are *optimal* in the long-run. Thus, we require a quantiative measure which can be used to define optimality. This is done by extending the hybrid system HS with new continuous state variables. The new continuous variables compute "rewards" or "penalties" that are accumulated over the course of a hybrid trajectory. We also allow the new variables to be updated during discrete transitions, which enables us to penalize or reward discrete mode switches.

Definition 4. Performance Metric. *A* performance metric *for a given multimodal system* $\texttt{MDS} := \langle Q, X, f, \texttt{Init}\rangle$ *is a tuple* $\langle \texttt{PR}, f_{\texttt{PR}}, \texttt{update}\rangle$*, where* $\texttt{PR} := P \cup R$ *is a finite set of continuous variables (disjoint from* X*), partitioned into penalty variables* P *and reward variables* R*,* $f_{\texttt{PR}} : Q \times \mathbb{R}^X \mapsto \mathbb{R}^{\texttt{PR}}$ *defines the vector field that determines the evolution of the variables* PR*, and* $\texttt{update} : Q \times Q \times \mathbb{R}^{\texttt{PR}} \mapsto \mathbb{R}^{\texttt{PR}}$ *defines the updates to the variables* PR *at mode switches.*

Given a trajectory $\mathbf{qx} : \mathbb{R}^+ \mapsto (Q \times \mathbb{R}^X)$ of a multimodal or hybrid system, and given a performance metric, we can define the *extended trajectory* $\mathbf{qx}^e : \mathbb{R}^+ \mapsto (Q \times \mathbb{R}^X \times \mathbb{R}^{\texttt{PR}})$ as follows. Let $t_1, t_2, \ldots$ be the mode-switching time sequence defined by the trajectory $\mathbf{qx}$. The extended trajectory $\mathbf{qx}^e$ is defined with respect to the same mode-switching time sequence and it is any function that satisfies $\mathbf{qx}^e(0) = (\mathbf{q}(0), \mathbf{x}(0), \mathbf{0})$ and $\mathbf{qx}^e(t) = (\mathbf{q}(t), \mathbf{x}(t), \mathbf{PR}(t))$, where $\mathbf{PR}$ satisfies: $\frac{d\mathbf{PR}(t)}{dt} = f_{\texttt{PR}}(\mathbf{qx}(t))$ for all $t : t_i < t < t_{i+1}$, and $\mathbf{PR}(t_i) = \texttt{update}(\mathbf{q}(t_{i-1}), \mathbf{q}(t_i), \lim_{t \to t_i^-} \mathbf{PR}(t))$.

We can now define the cost of a trajectory as follows. If $\mathbf{qx}$ is a trajectory of a multimodal or hybrid system and $\mathbf{qx}^e$ is the corresponding extended trajectory defined by a given performance metric, then

$$\texttt{cost}(\mathbf{qx}) := \lim_{t \to \infty} \sum_{i=1}^{|P|} \frac{\mathbf{P}_i(t)}{\mathbf{R}_i(t)} \tag{2}$$

where $\mathbf{P}_i$ and $\mathbf{R}_i$ are the projection of $\mathbf{qx}^e$ onto the i-th penalty variable and i-th reward variable, and $|P| = |R|$.

We are only interested in trajectories where the above limit exists and is finite. We will further define cost of a part of a trajectory from time instant t_1 to a time instant t_2 $(t_2 > t_1)$ as follows:

$$\texttt{cost}(\mathbf{qx}, t_1, t_2) := \sum_{i=1}^{|P|} \frac{\mathbf{P}_i(t_2) - \mathbf{P}_i(t_1)}{\mathbf{R}_i(t_2) - \mathbf{R}_i(t_1)} \tag{3}$$

where $\mathbf{P}_i$ and $\mathbf{R}_i$ are components of $\mathbf{PR}$ as before.

As the definition of cost indicates, we are interested in the *long-run average* (penalty per unit reward) cost rather than (penalty or reward) cost over some bounded/finite time horizon. Some examples of auxiliary performance variables (PR) and cost function are described below.

- the *number of switches* that take place in a trajectory can be tracked by defining an auxiliary variable p_1 that has dynamics $\frac{dp_1}{dt} = 0$ at all points in the state space, and that is incremented by 1 at every mode switch; that is,

$$p_1(t_i) = \texttt{update}(q, q', p_1(t_i^-)) = p_1(t_i^-) + 1$$

- the *time elapsed since start* can be tracked by defining an auxiliary variable r_1 that has dynamics $\frac{dr_1}{dt} = 1$ at all points and that is left unchanged at discrete transitions; that is,

$$r_1(t_i) = \texttt{update}(q, q', r_1(t_i^-)) = r_1(t_i^-)$$

– the *average switchings* (per unit time) can be observed to be $\frac{p_1}{r_1}$. If this cost becomes unbounded as the time duration of a trajectory increases, then this indicates *zeno* behavior. Thus, if we use p_1 and r_1 as the penalty and reward variables in the performance metric, then we are guaranteed that non-zeno systems will have "smaller" cost and thus be "better".
– the *power consumed* could change in different modes of a multimodal system and an auxiliary (penalty) variable can track the power consumed in a particular trajectory.
– the *distance from unsafe region* can be tracked by an auxiliary reward variable that evolves based on the distance of the current state from the closest unsafe state.

2.3 Optimal Switching Logic Synthesis

Definition 5. Optimal Switching Synthesis Problem. *Given a multimodal system* $\mathtt{MDS} = \langle Q, X, f, \mathtt{Init}\rangle$ *, and a performance metric, the* optimal switching logic synthesis problem *seeks to find a switching logic* $\mathtt{SwL}^*$ *such that the cost of a trajectory from any initial state in the resulting hybrid system* $\mathtt{HS}^* := \mathtt{HS}(\mathtt{MDS}, \mathtt{SwL}^*)$ *is no more than the cost of corresponding trajectory from the same initial state in an arbitrary hybrid system* $\mathtt{HS} := \mathtt{HS}(\mathtt{MDS}, \mathtt{SwL})$ *obtained using an arbitrary switching logic* $\mathtt{SwL}$*, that is,* $\forall(\mathbf{x}, q) \in \mathtt{Init}\ .\ \mathtt{cost}(\mathbf{qx}^*) \leq \mathtt{cost}(\mathbf{qx})$ *where* $\mathbf{qx}^*(0) = \mathbf{qx}(0) = (\mathbf{x}, q), \mathbf{qx} \in [\![\mathtt{HS}^*]\!], \mathbf{qx} \in [\![\mathtt{HS}]\!]$

We will assume, without loss of any generality, that we are given an over-approximation of the switching logic $\mathtt{SwL}^{over} := \langle(\mathtt{g}^{over}_{qq'})_{q,q'\in Q}\rangle$. In this case, the optimal synthesis problem seeks to find a switching logic $\mathtt{SwL}^* := \langle(\mathtt{g}^*_{qq'})_{q,q'\in Q}\rangle$ that also satisfies the constraint that $\mathtt{g}^*_{qq'} \subseteq \mathtt{g}^{over}_{qq'}$ for all $q, q' \in Q$, which is also written in short as $\mathtt{SwL}^* \subseteq \mathtt{SwL}^{over}$.

The over-approximation $\mathtt{SwL}^{over}$ of the switching set can be used to restrict the search space for switching conditions. The set $\mathtt{g}^{over}_{qq'}$ can be an empty set if switches are disallowed from q to q'. The set $\mathtt{g}^{over}_{qq'}$ can be $\mathbb{R}^X$ if there is no restriction on switching from q to q'.

2.4 Running Example

Let us consider a simple three mode thermostat controller as our running example. The multimode dynamical system describing this system is presented in Figure 1. The thermostat controller is described by the tuple $\langle Q, X, f, \mathtt{Init}\rangle$ where $Q = \{\mathtt{OFF}, \mathtt{HEAT}, \mathtt{COOL}\}$, $X = \{\mathtt{temp}, \mathtt{out}\}$, f is $f_{\mathtt{OFF}} : \dot{\mathtt{temp}} = -0.1(\mathtt{temp} - \mathtt{out})$ in mode $\mathtt{OFF}$, $f_{\mathtt{HEAT}} : \dot{\mathtt{temp}} = -0.1(\mathtt{temp} - \mathtt{out}) + 0.5(80 - \mathtt{temp})$ in mode $\mathtt{HEAT}$ and $f_{\mathtt{COOL}} : \dot{\mathtt{temp}} = -0.1(\mathtt{temp} - \mathtt{out}) + 0.5(80 - \mathtt{temp})$ in mode $\mathtt{COOL}$, and $\mathtt{Init} = \mathtt{OFF} \times [18, 20] \times [12, 26]$. For simplicity, we assume that the outside temperature $\mathtt{out}$ does not change.

The performance requirement is to keep the temperature as close as possible to the target temperature 20 and to consume as little fuel as possible in the long run. We also want to minimize the wear and tear of the heater caused by switching. The performance metric is given by the tuple $\langle \mathtt{PR}, f_{\mathtt{PR}}, \mathtt{update}\rangle$, where penalty variables

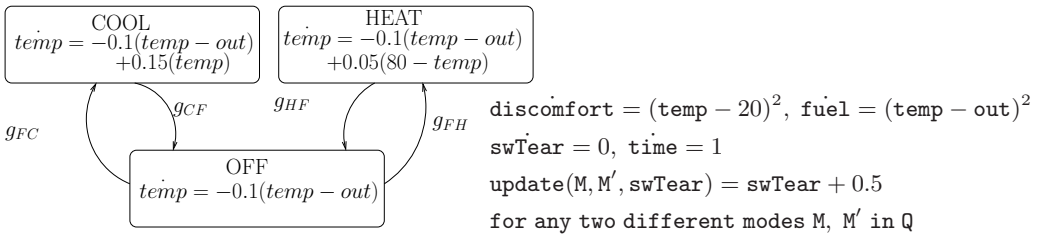

Fig. 1. Thermostat Controller

$P = \{\texttt{discomfort}, \texttt{fuel}, \texttt{swTear}\}$ denote the discomfort, fuel and wear-tear due to switching and reward variables $R = \{\texttt{time}\}$ denote the time spent. The evolution and update functions for the penalty and reward variables is shown in Figure 1. We need to synthesize the guards such that the following cost metric is minimized. Since the reward variable is the time spent, minimizing this metric means minimizing the *average* discomfort, fuel cost and wear-tear of the heater. We give a higher weight (10) to discomfort than fuel cost and wear-tear.

$$\lim_{t \to \infty} \frac{10 \times \texttt{discomfort}(t) + \texttt{fuel}(t) + \texttt{swTear}(t)}{\texttt{time}(t)}$$

3 Related Work

There is a lot of work on synthesis of controllers for hybrid systems, which can be broadly classified along several different dimensions. First, based on the *property of interest*, synthesis work broadly falls into one of two categories. The first category finds controllers that meet some *liveness* specifications, such as synthesizing a trajectory to drive a hybrid system from an initial state to a desired final state [15, 12], while also minimizing some cost metric [4]. The second category finds controllers that meet some safety specification; see [1] for detailed related work in this category. While our work does not directly consider only safety or only liveness requirements, both these requirements can be suitably incorporated into the definition of "reward" and "penalty" functions that define the cost that our approach then optimizes. While optimal control problems for hybrid systems have been formulated where cost is defined over some finite trajectory, we are unaware of any work in control of hybrid systems that attempts to formulate and solve the optimal control problem for *long-run* costs.

The second dimension that differentiates work on controller synthesis for hybrid systems is the space of control inputs considered; that is, what is assumed to be controllable. The space of controllable inputs could consist of any combination of continuous control inputs, the mode sequence, and the dwell times within each mode. A recent paper by Gonzales et al. [9] consider all the three control parameters, whereas some other works either assume the mode sequence is not controllable [24, 21] or there are no continuous control inputs [2]. In our work, we assume there are no continuous control inputs and both the mode sequence and the dwell time within each mode are controllable entities.

The third dimension for placing work on controller synthesis of hybrid systems is the approach used for solving the synthesis problem. There are direct approaches for synthesis that compute the controlled reachable states in the style of solving a game [1, 23], and abstraction-based approaches that do the same, but on an abstraction or approximation of the system [18, 8, 22]. Some of these approaches are limited in the kinds of continuous dynamics they can handle. They all require some form of iterative fixpoint computation. The other class of approaches are based on using nonlinear optimization techniques and gradient descent [9, 2]. We also use similar techniques in this paper and reduce the controller synthesis problem to an optimization problem of a function that is computed by performing simulations of the dynamical system.

Notions of long-run cost similar to ours have appeared in other areas. The notion of long-run average cost is used in economics to describe the cost per unit output (reward) in the long-run. In computer science, long-run costs have been studied for graph optimization problems [11]. Long-run average objectives have also been studied for markov decision processes (MDPs) [6, 13]. However, MDPs do not have any continuous dynamics. There is some recent work on controller synthesis with budget contraints where the budget applies in the long-run [7].

In contrast to existing literature, we present an automated synthesis algorithm to synthesize switching logic $\texttt{SwL}$ for a given $\texttt{MDS}$ and performance metric such that all trajectories in the hybrid system $\texttt{HS}(\texttt{MDS}, \texttt{SwL})$ have minimum long-term cost with respect to the given performance metric.

4 Optimization Formulation

In this section, we formulate the problem of finding switching logic for minimum long-run cost from an initial state as an optimization problem. Given a multimodal system $\texttt{MDS} = \langle Q, X, f, d, Inv, Init\rangle$, an initial state $(q_0, \mathbf{x}_0) \in Init$ and the performance metric tuple $(Y, f_Y, \texttt{update})$, we need to find the switching times $t_1, t_2, \ldots$ and the mode switching sequence $\mathbf{q}$ such that the corresponding trajectory $\mathbf{qx}$ is of minimum cost.

$$\min_{\mathbf{q}, t_1, t_2, \ldots} \texttt{cost}(\mathbf{qx}) \quad \textit{subject to} \tag{4}$$

$$1(\text{Init}):\ \mathbf{qx}(0) = (q_0, \mathbf{x}_0) \qquad 2(\text{Time elapse}):\ \forall t\ .\ t_i < t < t_{i+1}\ .\ \mathbf{q}(t) = \mathbf{q}(t_i)\ \ ,\ i = 1, 2, \ldots;$$

$$3(\text{Guards}):\ \forall i\ \mathbf{x}(t_i) \in \mathbf{g}^{over}_{\mathbf{q}(i)\mathbf{q}(i+1)} \qquad 4(\text{Flow}):\ \forall t\ \frac{d\mathbf{x}(t)}{dt} = f(\mathbf{q}(t), \mathbf{x}(t))$$

A naive approach to solve the above optimization algorithm requires searching over all the mode sequences $\mathbf{q}$ and all the switching sequences $t_1, t_2, \ldots$ to discover the trajectory $\mathbf{qx}$ of minimum cost. Since trajectories are of infinite length and consequently the switching sequences have infinite number of switching times, such a naive search is not feasible. In the rest of the section, we formulate an equivalent optimization problem with finite number of switching times as variables.

Let *trajectory segment* $\mathbf{qx}_{[tf,te]}$ of a trajectory $\mathbf{qx}$ of length $L = tf - te$ be the restriction of the *trajectory* to $tf \le t \le te$, that is, $\mathbf{qx}_{[tf,te]} : T \mapsto (Q \times \mathbb{R}^X)$ where $T = [tf, te] \subseteq \mathbb{R}^+$ and $\mathbf{qx}_{[tf,te]}(t) = \mathbf{qx}(t)$ for $tf \le t \le te$. The *switching times* of

the trajectory segment is a finite subsequence $t_m, t_{m+1}, \ldots t_n$ of the switching times $t_1 \ldots, t_m, \ldots, t_n, \ldots$ of the trajectory $\mathbf{qx}$ and $t_{m-1} < tf \leq t_m$ and $t_n \leq te < t_{n+1}$. The special case of a trajectory segment is a *trajectory prefix* in which the trace starts at time $ts = 0$.

Our goal is to minimize the lifetime cost. The lifetime cost is dominated by the the cost of the *limit behavior* of the system. In Appendix A, we discuss different possible *limit behaviors*. We are only interested in the following stable limit behaviors when the lifetime cost is defined by the limit in Equation 2.

- *asymptotic:* for any ϵ, there exists a time t_ϵ after which the trajectory gets asymptotically ϵ-close to some state $(q_T, \mathbf{x}_T)$, $||\mathbf{qx}(t) - (q_T, \mathbf{x}_T)||^2 < \epsilon$ for all $t \geq t_\epsilon$ where $||.,.||$ denotes the Euclidean norm, or
- *converging:* there exists a time t_{conv} after which the trajectory converges, $\mathbf{qx}(t) = \mathbf{qx}(t_{conv})$ for all $t \geq t_{conv}$, or
- *cyclic:* there exists a time t_{cyc} after which the trajectory enters a cycle with period L, $\mathbf{qx}(t) = \mathbf{qx}(t + kP)$ for all $t \geq t_{cyc}$ and $k \geq 1$.

In all these cases, we can reason about the long-run cost by considering some finite, but arbitrarily long, trajectory prefixes. Suppose the trajectory $\mathbf{qx}$ is asymptotic to some hybrid state $\mathbf{qx}^\infty = (q^\infty, \mathbf{x}^\infty)$. In this case, we assume that the penalty and reward variables $\mathtt{PR}$ also asymptotically approach some values $\mathtt{PR}^\infty = (P^\infty, R^\infty)$. Now consider the trajectory prefix $\mathbf{qx}_{[0,te]}$. We have

$$\mathtt{cost}(\mathbf{qx}) = \sum_i \frac{P_i^\infty}{R_i^\infty} \quad \text{and} \quad \mathtt{cost}(\mathbf{qx}_{[0,te]}) = \sum_i \frac{P_i(te)}{R_i(te)} < \sum_i \frac{P_i^\infty + \epsilon}{R_i^\infty - \epsilon} < \mathtt{cost}(\mathbf{qx}) + \delta_\epsilon$$

Hence, by choosing te appropriately, we can find a trajectory prefix whose cost is arbitrarily close to the cost of the asymptotic trajectory.

Any repetitive trajectory $\mathbf{qx}$ can be decomposed into a finite prefix $\mathbf{qx}_{pref} = \mathbf{qx}_{[0,tp]}$ followed by a trajectory segment $\mathbf{qx}_{rep} = \mathbf{qx}_{[tp,tP]}$ repeated infinitely. We say

$$\mathbf{qx} = \mathbf{qx}_{pref} \cdot (\mathbf{qx}_{rep})^\omega$$

when $\forall t \leq tp \;.\; \mathbf{qx}(t) = \mathbf{qx}_{pref}(t)$ and $\forall t \geq tp \;.\; \mathbf{qx}(t) = \mathbf{qx}_{rep}(tp + r)$ where $r = (t - tp) \mod L$ and $L = tP - tp$.

If the trajectory converges to some hybrid state after t_{conv}, $tp = t_{conv}$ and tP is any time after convergence, that is, $tP > tp$. If the trajectory enters a cycle with a period L after time t_{cyc}, then $tp = t_{cyc}$ and $tP = tp + L$. In Lemma 1 and Theorem 1, we summarize how cost converges to a limit for trajectories with repetitive limit behavior.

Lemma 1. *For each repetition of the segment $\mathbf{qx}_{rep} = \mathbf{qx}_{[tp,tP]}$, the change in penalty and reward variables is constant, that is, for $P = tP - tp$.*

$$\forall k \geq 1 \;.\quad P_i(tp + kP) - P_i(tp + (k-1)P) = P_i(tp + P) - P_i(tp) = \Delta P_i$$
$$\forall k \geq 1 \;.\; R_i(tp + kP) - R_i(tp + (k-1)P) = R_i(tp + P) - R_i(tp) = \Delta R_i$$

Proof. The change in penalty and reward variables is given by the evolution function $f_{\mathtt{PR}}$ and the update on switch function `update`. We know that $\mathbf{qx}_{rep}$ is repetitive and so, $\mathbf{qx}(tp + kP + t) = \mathbf{qx}(tp + t)$ for all $t < tP - tp$ and $k \geq 1$ and hence,

$$f_{\mathtt{PR}}(\mathbf{qx}(tp + kP + t)) = f_{\mathtt{PR}}(\mathbf{qx}(tp + t))$$

Also, for any mode switch time $tp \leq t_i \leq tP$, $t'_i = t_i + kP$ is also a switch time because hybrid states at t_i and t'_i are the same. Further,

$$\mathtt{update}(q(t_{i-1}), q(t_i), \lim_{t \to t_i^-} \mathbf{PR}(t))$$

$$= \mathtt{update}(q(t'_{i-1}), q(t'_i), \lim_{t \to t_i'^-} \mathbf{PR}(t))$$

So, integrating f_{PR} over continuous evolution and applying `update` function at mode switches, we observe that

$$\begin{aligned} P_i(tp + kP) - P_i(tp + (k-1)P) &= P_i(tp + P) - P_i(tp) \\ &= \Delta P_i \\ R_i(tp + kP) - R_i(tp + (k-1)P) &= R_i(tp + P) - R_i(tp) \\ &= \Delta R_i \end{aligned}$$

Theorem 1. *For a trajectory* $\mathbf{qx}$ *which can be decomposed into* $\mathbf{qx}_{pref} \cdot (\mathbf{qx}_{rep})^{\omega}$*, the cost of the trajectory is equal to the cost of the repetitive segment* $\mathbf{qx}_{rep}$*, that is,* $\mathtt{cost}(\mathbf{qx}) = \mathtt{cost}(\mathbf{qx}_{rep})$.

Proof.

$$\mathtt{cost}(\mathbf{qx}) := \lim_{t \to \infty} \sum_{i=1}^{|P|} \frac{\mathbf{P}_i(t)}{\mathbf{R}_i(t)} \text{ [Equation 2]} = \lim_{t \to \infty} \sum_{i=1}^{|P|} \frac{\mathbf{P}_i(tp) + \mathbf{P}_i(t) - \mathbf{P}_i(tp)}{\mathbf{R}_i(tp) + \mathbf{R}_i(t) - \mathbf{R}_i(tp)}$$

$$= \lim_{k \to \infty} \sum_{i=1}^{|P|} \frac{\mathbf{P}_i(tp) + k\Delta\mathbf{P}_i}{\mathbf{R}_i(tp) + k\Delta\mathbf{R}_i} \text{ [Lemma 1]} = \frac{\Delta\mathbf{P}_i}{\Delta\mathbf{R}_i} \text{ [}\mathbf{P}_\mathrm{i}(tp), \mathbf{R}_\mathrm{i}(tp) \text{ are finite]} = \sum_{i=1}^{|P|} \frac{\mathbf{P}_i(tP) - \mathbf{P}_i(tp)}{\mathbf{R}_i(tP) - \mathbf{R}_i(tp)}$$

$$= \mathtt{cost}(\mathbf{qx}, tp, tP) \text{ [Equation 3]} = \mathtt{cost}(\mathbf{qx}_{rep}) \text{ [Definition of } \mathbf{qx}_{\mathrm{rep}}]$$

□

Using Theorem 1, the optimization problem in Equation 4 is equivalent to the following optimization problem. Intuitively, if the repetitive part of the trajectory and the finite prefix before the repetitive part have finite cost, then the long run cost of a trajectory in the limit is the cost of the repetitive part of the trajectory. More generally, to also handle the case when the (optimal) trajectory is asymptotic, we can replace the cyclicity requirement, $\mathbf{qx}(tp) = \mathbf{qx}(tP)$, in the optimization problem by the weaker requirement that the state $\mathbf{qx}(tP)$ at time tP be very "close" to the state $\mathbf{qx}(tp)$ at time tp; see also Section 5.1.

$$\min_{\mathbf{q},t_1,t_2,\ldots,t_n} \texttt{cost}(\mathbf{qx}_{rep}) \quad \textit{subject to} \tag{5}$$

$$1(\text{Init}):\ \mathbf{qx}(0) = (q_0, \mathbf{x}_0) \qquad 2(\text{Time elapse}):\ \forall t\ .\ t_i < t < t_{i+1}\ .\ \mathbf{q}(t) = \mathbf{q}(t_i)\ ,\ i = 1, 2, \ldots;$$

$$3(\text{Guards}):\ \forall i\ \mathbf{x}(t_i) \in \mathbf{g}^{over}_{\mathbf{q}(i)\mathbf{q}(i+1)} \qquad 4(\text{Flow}):\ \forall t\ \frac{d\mathbf{x}(t)}{dt} = f(\mathbf{q}(t), \mathbf{x}(t))$$

$$5(\text{Repetitive Trajectory}):\ \mathbf{qx} = \mathbf{qx}_{pref}\ .\ (\mathbf{qx}_{rep})^{\omega}$$

$$6(\text{Repetitive Time}):\ \mathbf{qx}_{pref} = \mathbf{qx}_{[0,tp]},\ \mathbf{qx}_{rep} = \mathbf{qx}_{[tp,tP]}$$

$$\text{where } 0 \leq t_1 \leq \ldots t_n \leq tP,\ 0 \leq tp < tP$$

5 Optimization Algorithm

In this section, we present an algorithm to solve the above optimization problem. The key idea is to construct a scalar function $F(\mathbf{q}, t_1, t_2, \ldots, t_n, tp, tP)$ where $\mathbf{q}$ denotes the switching mode sequence; $t_1, t_2, \ldots, t_n$ denote the switching times, and tp, tP denote the times denoting repetitive behavior, such that the minimum value of F is attained when the switching mode sequence and switching times correspond to the trajectory $\mathbf{qx}$ with minimum long-run cost, and $\mathbf{qx}_{[tp,tP]}$ is the repetitive part of the trajectory.

For any given value of the arguments, the function F would be computed using numerical simulation [3] of the multimodal system. If F is a scalar function of several variables which can be computed for given values of the arguments, minimization of F can be done using *unconstrained nonlinear numerical optimization* techniques [3].

Our algorithm for solving the constrained optimization function in Equation 5 is based on defining the function F and then using unconstrained nonlinear optimization techniques to minimize F. We use standard techniques for unconstrained nonlinear optimization. The novelty of our technique lies in formulating the function F such that minimizing F yields the solution for the optimization problem in Equation 5. For ease of presentation, we present our solution in two steps. Firstly, we consider the problem with fixed switching sequence of modes and then, we show how our technique can be used to discover the switching sequence as well.

5.1 Finding Switching States with Fixed Mode Sequence

Let $F_{\mathbf{q}}(t_1, t_2, \ldots, t_n, tp, tP)$ denote the function F with fixed mode sequence $\mathbf{q}$. Minimizing $F_{\mathbf{q}}$ over its arguments yields switching times $t_1, t_2, \ldots, t_n$ and tp, tP such that the trajectory $\mathbf{qx}$ starting from the initial state $(q_0, \mathbf{x}_0)$ enters repetitive behavior at tp and the trajectory repeats with a period of $tP - tp$, that is, $\mathbf{qx}(tp) = \mathbf{qx}(tP)$, and the corresponding trajectory segment $\mathbf{qx}_{rep} = \mathbf{qx}_{[tp,tP]}$ is of minimum cost. So, minimizing F would yield the solution to the optimization problem in Equation 5.

The optimization problem in Equation 5 is a constrained optimization problem. The constraint $\mathbf{qx} = \mathbf{qx}_{pref}\ .\ (\mathbf{qx}_{rep})^{\omega}$ requires identifying a trajectory $\mathbf{qx}$ starting

[3] We rely on simulating continuous behavior described by ODEs in a single mode for a fixed time period and accurate simulation of ODEs is a well-studied problem.

from the given initial state $(q_0, \mathbf{x}_0)$ such that it enters repetitive behavior at time tp, and $q(tp) = q(tP)$ and $\mathbf{x}(tp) = \mathbf{x}(tP)$ where $tp < tP$. We call this constraint the repetition constraint. A standard technique for solving some constrained optimization problems is to translate it into an unconstrained optimization problem by modifying the optimization objective such that optimization automatically enforces the constraint. This is done by quantifying the violation using some metric and then minimizing the sum of the earlier minimization objective and the weighted violation measure. A simple example of this approach is presented in Appendix B. In order to enforce the repetition constraint by suitably modifying the optimization objective, we introduce a distance function between the hybrid states. Let d be the distance function between two hybrid states such that

$$d((q_1, \mathbf{x}_1), (q_2, \mathbf{x}_2)) = ||\mathbf{x}_1 - \mathbf{x}_2||^2 \text{ if } q_1 = q_2 \text{ and } \infty \text{ } o.w.$$

where $||\mathbf{x}_1 - \mathbf{x}_2||$ is the Euclidean norm. $(Q \times X, d)$ forms a metric space. So, the distance between the hybrid states is 0 if and only if $q_1 = q_2$ and $\mathbf{x}_1 = \mathbf{x}_2$.

$$\text{Let } F(\mathbf{q}, t_1, \ldots, t_n, tp, tP) = \begin{cases} \texttt{cost}(\mathbf{qx}_{[tp,tP]}) + M \times d(\mathbf{qx}(tp), \mathbf{qx}(tP)) \\ \texttt{if } (a)\ 0 \leq t_1 \leq \ldots t_n \leq tP,\ tp < tP \text{ and } (b)\ \forall i\ \mathbf{x}(t_i) \in \mathtt{g}^{over}_{\mathbf{q}(i)\mathbf{q}(i+1)} \\ \infty \quad \texttt{otherwise} \end{cases}$$

where M is any positive constant and $\mathbf{qx}$ is a trajectory starting from the given initial state, that is, $\mathbf{qx}(0) = (q_0, \mathbf{x}_0)$; $\forall t\ t_i < t < t_{i+1}\ \mathbf{q}(t) = \mathbf{q}(t_i)$, $i = 1, 2, \ldots$ and $\forall t\ \frac{d\mathbf{x}(t)}{dt} = f(\mathbf{q}(t), \mathbf{x}(t))$.

$F_{\mathbf{q}}$ is the restriction of the function F where the mode switching sequence is fixed to $\mathbf{q}$. The function $F_{\mathbf{q}}$ evaluates to a finite value only if the arguments satisfy conditions (a) and (b). The first condition is that the switching times are non-decreasing and the times for possible repetitive behavior satisfy $tp < tP$. The second condition is that the switching states $\mathbf{x}(t_i)$ at switching times t_i lie in the user specified over-approximation of the guards, that is, $\mathbf{x}(t_i) \in \mathtt{g}^{over}_{\mathbf{q}(i)\mathbf{q}(i+1)}$. If any of these two conditions are not satisfied by the arguments, then the function $F_{\mathbf{q}}$ evaluates to infinite. If the two conditions are satisfied, $F_{\mathbf{q}}$ is the sum of the cost of the trajectory segment $\mathbf{qx}_{[tp,tP]}$ and the distance between the hybrid states at time tp and tP weighted with a very large constant M. The minimum value of the function $F_{\mathbf{q}}$ is attained when the hybrid states at time tp and tP are the same, that is, the trajectory segment $\mathbf{qx}_{[tp,tP]}$ is the repetitive part of the trajectory and the cost of this segment is minimum. With the switching states given by $\mathbf{qx}(t_i)$, the trajectory starting with the given initial state would be $\mathbf{qx}_{[0,tp]}(\mathbf{qx}_{[tp,tP]})^{\omega}$. Since, the cost of repetitive segment $\mathbf{qx}_{[tp,tP]}$ is minimized, this trajectory is of minimum long-run cost. Thus, the optimization problem in Equation 5 of Section 4 can be reduced to the following unconstrained multivariate numerical optimization problem

$$\min_{t_1,\ldots,t_n,tp,tP} \quad F_{\mathbf{q}}(t_1, \ldots, t_n, tp, tP) \tag{6}$$

For given values of $\langle t_1, \ldots, t_n, tp, tP \rangle$, $F_{\mathbf{q}}(t_1, \ldots, t_n, tp, tP)$ can be computed using a numerical simulator. We simulate starting from initial state $(q_0, \mathbf{x}_0)$ till time tP with the switching times $t_1, \ldots, t_n$ and the mode switching sequence $\mathbf{q}$. Starting

from some initial state $(q_0, \mathbf{x}_0)$, we simulate the continuous dynamics in different modes till time tP. We simulate the continuous dynamics in the first mode in the given mode sequence till time t_1 and then the following mode in the given sequence for time $t_2 - t_1$ time and so on. We simulate the last mode in the sequence for time $tP - t_n$. During simulation, we also compute the reward and penalty by recording the extended trajectory $\mathbf{qx}^e$ as defined in Section 2. The cost of the repetitive part is computed using Equation 3 as

$$\texttt{cost}(\mathbf{qx}_{[tp,tP]}) = \texttt{cost}(\mathbf{qx}, tp, tP) = \frac{\mathbf{P}(tP) - \mathbf{P}(tp)}{\mathbf{R}(tP) - \mathbf{R}(tp)}$$

We also record the states at times tp and tP to compute the distance between them $d(\mathbf{qx}(tp), \mathbf{qx}(tP))$. If the two conditions required for $F_{\mathbf{q}}$ to be finite are satisfied, then $F_{\mathbf{q}}(t_1, \ldots, t_n, tp, tP)$ is the sum of the cost of the repetitive part and the weighted distance. We use a numerical nonlinear optimization engine to find the minimum value of the function $F_{\mathbf{q}}$.

Running Example We illustrate our technique for the running example with a fixed sequence of modes say $\mathbf{q} = \texttt{OFF}, \texttt{HEAT}, \texttt{OFF}$ starting from the initial state $(\texttt{OFF}, \texttt{temp} = 22, \texttt{out} = 16)$. The outside temperature `out` does not change with time and remains the same as the initial state. Only the room temperature `temp` changes with time. The switching time sequence is t_1, t_2. Let tp denote the time when the thermostat enters the repetitive behavior and tP be the time such that $\texttt{temp}(tp) = \texttt{temp}(tP)$. When $t_1 \leq t_2 \leq tp \leq tP$ and $tp < tP$, the function

$$F_{\mathbf{q}}(t_1, t_2, tp, tP) = \texttt{cost}(\mathbf{qx}_{[tp,tP]}) + 1000(\texttt{temp}(tp) - \texttt{temp}(tP))^2$$

and it is set to 2000 otherwise (approximating infinity in the formulation with a very high constant). We use *ode*45 function in MATLAB [17] for numerically simulating the ordinary differential equations representing continuous dynamics in each mode. In order to find the minimum value of $F_{\mathbf{q}}$ and the corresponding arguments that minimize the function, we use the implementation of Nelder-Mead simplex algorithm [19] available in MATLAB as $fminsearch$ function [16]. The minimum value of $F_{\mathbf{q}}$ is obtained at

	t_0	t_1	t_2	tp	tP
t	0	5.02	5.24	3.54	5.24
$temp$	22.0	19.6	20.2	20.02	20.2

So, the switch states corresponding to the minimum long-run cost for the given initial state $(\texttt{OFF}, \texttt{temp} = 22, \texttt{out} = 16)$ and given switching sequence of modes $\texttt{OFF}, \texttt{HEAT}, \texttt{OFF}$ is $\texttt{g}_{HF} = \{20.2\}$ and $\texttt{g}_{FH} = \{19.6\}$.

We repeat the experiments with different initial states but with the same mode switching sequence. Even with different initial states $(\texttt{OFF}, \texttt{temp} = 20.5, \texttt{out} = 16)$, $(\texttt{OFF}, \texttt{temp} = 21, \texttt{out} = 16)$ and $(\texttt{OFF}, \texttt{temp} = 21.5, \texttt{out} = 16)$, we obtain the same switching states in this example: $\texttt{g}_{HF} = \{20.2\}$ and $\texttt{g}_{FH} = \{19.6\}$.

When we change the mode switching sequence to OFF, HEAT, OFF, HEAT, OFF, we discover the optimal switching sequence to be

	t_0	t_1	t_2	t_3	t_4	tp	tP
t	0	5.02	5.24	6.73	6.95	3.54	6.95
$temp$	22.0	19.6	20.2	19.6	20.2	20.02	20.2

$t_1 = 5.02, t_2 = 5.24, t_3 = 6.73, t_4 = 6.95, tp = 3.54, tP = 6.95$ which again yields the same optimal switching states $\mathtt{g}_{HF} = \{20.2\}$ and $\mathtt{g}_{FH} = \{19.6\}$.

We observe that the optimal behavior with respect to the given cost metric would be to switch from OFF mode to HEAT mode at temp $= 19.6$ and then switch from HEAT to OFF mode at temp $= 20.2$ regardless of the initial room temperature as long as the outside temperature out $= 16$. The optimal mode cycle is between OFF and HEAT modes.

For an initial state with outside temperature higher than the outside room temperature out > 20, the optimal cycle would be between OFF and COOL modes. With the mode sequence OFF, COOL, OFF and the initial state (OFF, temp $= 20.5$, out $= 26$), we discover the optimal switching states to be $\mathtt{g}_{CF} = \{20\}$ and $\mathtt{g}_{FC} = \{20.3\}$.

5.2 Finding Optimal Mode Sequence

The algorithm presented above can be easily adapted to automatically discover the switching mode sequence that corresponds to the minimum long-run cost trajectory. Any mode sequence starting in mode 1 and with atmost k switches in a system with N modes $Q = \{1, 2, \ldots N\}$ is a subsequence of $1(2\ \ldots\ N\ 1)^k$, that is, mode 1 followed by $(2\ \ldots\ N\ 1)$ repeated k times. Let dwell-time of a mode i be the time spent in the mode $t_{i+1} - t_i$. Given the switching times $t_1, t_2, ...t_{Nk}$ and tp, tP, we define the NZ function which removes the switch times and modes from the switching sequence with zero dwell-times, that is,

$$NZ(\overline{\mathbf{q}}, t_1, t_2, \ldots, t_{Nk}, tp, tP) = (\mathbf{q}, t_{i_1}, t_{i_2}, \ldots, t_{i_K}, tp, tP) \text{ where}$$

$\mathbf{q} = q_{i_1}, q_{i_2}, \ldots, q_{i_K}, 0 < t_{i_1} < t_{i_2} < \ldots < t_{i_K} < tP$ and $t_m = t_{i_j}$ for all $i_j < m < i_{j+1}$

For example, given the sequence of switching times $5, 6, 6,\ 11, 12, 12$ and $tp = 6.5, tP = 12.5$ with the switching mode sequence $\overline{\mathbf{q}} = 1, 2, 3, 1, 2, 3, 1$,

$$NZ(\overline{\mathbf{q}}, 5, 6, 6, 11, 12, 12, 6.5, 12.5) = (\mathbf{q}, 5, 6, 11, 12, 6.5, 12.5) \text{ where } \mathbf{q} = 1, 2, 1, 2, 1$$

Given a guess on the number of mode switches k such that k or less switches are needed to reach the optimal repetitive behavior, we can use $\overline{\mathbf{q}} = 1(2\ \ldots\ N\ 1)^k$ as the over-approximate switching mode sequence and then find the optimal switching subsequence corresponding to the minimal long-run cost behavior using the following modified optimization formulation.

$$\min_{t_1,\ldots,t_{Nk},tp,tP} \quad F(NZ(\overline{\mathbf{q}}, t_1, \ldots, t_{Nk}, tp, tP)) \tag{7}$$

If the optimal value returned by minimizing the above function is attained with the arguments $t_1^*, \ldots, t_{Nk}^*, tp^*, tP^*$, then the optimal switching sequence $\mathbf{q}$ and the optimal switching time sequence is given by

$$(\mathbf{q}, t_{i_1}, \ldots, t_{i_k}, tp, tP) = NZ(\overline{\mathbf{q}}, t_1^*, \ldots, t_{Nk}^*, tp^*, tP^*)$$

Running Example We illustrate the above technique on the running example below. Let us guess that reaching the optimal repetitive behavior from the initial state $\texttt{OFF}, \texttt{temp} = 22, \texttt{out} = 16$ takes atmost 2 switches. We consider the mode sequence $\texttt{OFF}, \texttt{HEAT}, \texttt{COOL}, \texttt{OFF}, \texttt{HEAT}, \texttt{COOL}, \texttt{OFF}$ which would contain all mode sequences with 2 switches (it also contains some mode sequences with more than 2 switches). We try to minimize the corresponding function $F(NZ(t_1, t_2, \ldots, t_6, tp, tP))$.

The minimum value obtained for the function F with the starting state ($\texttt{OFF}, \texttt{temp} = 22, \texttt{out} = 16$) by our optimization engine corresponds to the following trajectory.

	t_0	t_1	t_2	t_3	t_4	t_5	t_6	tp	tP
t	0	5.08	5.32	5.32	6.97	7.23	7.23	4.87	8.66
$temp$	22.0	19.6	20.2	20.2	19.6	20.2	20.2	19.7	19.7

The optimal mode sequence and the switching times points are obtained as

$$NZ(\overline{\mathbf{q}}, 5.08, 5.32, 5.32, 6.97, 7.23, 7.23, 4.87, 8.66) = (\texttt{OFF}, \texttt{HEAT}, \texttt{OFF}, \texttt{HEAT}, \texttt{OFF}, 5.08, 5.32, 6.97, 7.23, 4.87, 8.66)$$

Since $tp = 4.87$ and $tP = 8.66$, the repetitive part of the mode sequence is $\texttt{HEAT}, \texttt{OFF}$. The switch from mode $\texttt{OFF}$ to $\texttt{HEAT}$ occurs at times t_1 and t_4. We observe that $\texttt{temp}(t_1) = \texttt{temp}(t_4) = 19.6$. So, the optimal trajectory switches from $\texttt{OFF}$ to $\texttt{HEAT}$ at $\texttt{temp} = 19.6$. The switches from $\texttt{HEAT}$ to $\texttt{COOL}$ and then to $\texttt{OFF}$ occur at the same times: $t_2 = t_3$ and $t_5 = t_6$. So, the dwell-time in the mode $\texttt{COOL}$ is 0 and it needs to be removed from the optimal switching sequence. The switch into mode $\texttt{OFF}$ occurs at times t_3 and t_6 with $\texttt{temp}(t_3) = \texttt{temp}(t_6) = 20.2$. Thus, the optimal mode sequence is $\texttt{OFF}, (\texttt{HEAT}, \texttt{OFF})^\omega$ and the guards discovered from this trajectory are $\texttt{g}_{FH} = 19.6$ and $\texttt{g}_{HF} = 20.2$. □

Thus, the approach presented so far can be used to synthesize switching conditions for minimum cost long-run behavior for a given initial state. We need a guess on the number of switches k such that the optimal behavior has atmost k switches. Our technique can automatically synthesize the optimal mode sequence as well as the switching states corresponding to the minimum long-run cost trajectory for the given initial state. With a perfect numerical simulator and a numerical optimization engine capable of finding the true minimum value of the function F, our technique would find the behavior with minimum cost. But finding the globally minimum cost is not guaranteed by existing optimization techniques. It can be observed from the definition of the function F that it is discontinuous and hence, minimizing F is difficult. The adaptation to find the mode switching sequence further adds discontinuity to the function F since, eliminating modes when the dwell time is 0 also removes the switch cost of switching into and out of this mode. Since F is a discontinuous function, standard numerical gradient based optimization techniques are not very effective. We employ the Nelder-Mead simplex algorithm as described by Lagarias et al [14, 19] for minimizing F since it is a derivative-free method and it can better handle discontinuity. We use its implementation available as the $fminsearch$ [16] function in MATLAB.

6 Multiple Initial States

The approach presented in Section 5 can find the switching states corresponding to optimal long-run behavior *for a given initial state*. However, multimodal systems generallly have more than one initial state.

In this case, we use the approach of Section 5 on each initial state and then define the optimal switching guards as follows. The guard $\mathtt{g}(q_i, q_j)$ for the transition from Mode i to Mode j contains $\mathbf{x}_k$ if for some initial state, the optimal trajectory had a jump from q_i to q_j at state $\mathbf{x}_k$.

Let this procedure of finding optimal switching states described in Section 5 for a given single initial state $(\mathbf{x}, q) \in \mathtt{Init}$ be called $getSS((\mathbf{x}, q))$ which returns the switching sequence $\mathbf{q} = q_{i_1}, q_{i_2}, \ldots$ and switching states $\mathtt{SwL}_{(\mathbf{x},q)}$ containing $(q_i, \mathbf{x}_i) \in \mathtt{g}^{over}_{q_i q_j}$ for each switch (q_i, q_j) in $\mathbf{q}$. Algorithm 1 can be used to find optimal switching logic for a set of initial states.

Input: Set of initial state $\mathtt{Init}$
Output: Switching Logic $\mathtt{SwL}$
foreach $(q_i, q_j) \in Q \times Q$ **do**
  $\mathtt{g}_{q_i q_j} = \{\}$ // Initialize guards to empty sets
end
foreach $(\mathbf{x}, q) \in \mathtt{Init}$ **do**
  $(\mathbf{q}, \mathtt{SwL}_{(\mathbf{x},q)}) = getSS((\mathbf{x}, q))$;
  foreach *switch* (q_i, q_j) *in* $\mathbf{q}$ **do**
    $\mathtt{g}_{q_i q_j} = \mathtt{g}_{q_i q_j} \cup \{(q_i, \mathbf{x}_i)\}$ where $(q_i, \mathbf{x}_i) \in \mathtt{SwL}_{(\mathbf{x},q)}$
  end
end
return $\mathtt{SwL} = \{\mathtt{g}_{q_i q_j} | (q_i, q_j) \in Q \times Q\}$

Algorithm 1: Switching Logic Synthesis for Multiple Initial States

The correctness of this approach for handling multiple initial state relies on the following *non-interference* theorem.

Theorem 2. *From any two initial states, either the system enters a common repetitive behavior* $\mathbf{qx}_{rep}$ *with minimum long-run cost starting from both initial states or their trajectories have no common state.*

Proof. There can be three possible scenarios for optimal trajectories from two different initial states illustrated in Figure 2.

- One of the initial state is reachable from another along the optimal trajectory of the later.
- There is some state $(q', \mathbf{x}')$ reachable from both initial states along their optimal trajectories.
- There is no state $(q', \mathbf{x}')$ reachable from both initial states along their optimal trajectories.

In the first two cases, since the optimal trajectory must be deterministic, it follows that the trajectories after their common state are same and hence, the trajectories

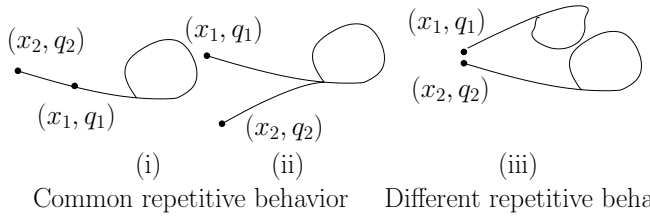

Fig. 2. Optimal trajectories from two initial states

have common repetitive part. In the third case, there is no state reachable in both optimal trajectories.

Nonlinear and hybrid systems show the phenomenon of *bifurcation* and can have multiple equilibria and different limit cycles from different initial states [20]. The above algorithm enumerates over all initial states. So, it is useful only when there are finite number of initial states. If the set of initial states is infinite, then we either sufficiently sample the initial states and generate the guards by *generalizing* the finitely many points we find on the optimal guards, or consider a finite partition of the initial states.

A common example of generalization is taking the convex hull of the points found. This approach works whenever the human designer can inform the synthesis algorithm about the *form* of the optimal guards.

Running Example Given the set of initial states $16 \leq \texttt{temp} \leq 26$ and $\texttt{out} \in \{16, 26\}$. The set of initial states is partitioned into subsets where each subset is a 0.1 interval of room temperature `temp` and the outside temperature is 16 or 26. The guards discovered are: $\mathtt{g}_{HF} : \texttt{temp} \geq 20.2 \wedge \texttt{out} = 16$, $\mathtt{g}_{FH} : \texttt{temp} \leq 19.6 \wedge \texttt{out} = 16$, $\mathtt{g}_{CF} : \texttt{temp} \leq 20.0 \wedge \texttt{out} = 26$, $\mathtt{g}_{FC} : \texttt{temp} \geq 20.3 \wedge \texttt{out} = 26$.

7 Case Studies

Apart from the running example of Thermostat controller, we applied our technique to two other case studies - Oil Pump Controller and Control Scheme for the DC-DC Buck-Boost Converter Circuit.

7.1 Thermostat Controller

If we change the cost metric in the thermostat controller to $\lim_{t\to\infty} \frac{\texttt{discomfort}(t)+\texttt{fuel}(t)+\texttt{swTear}(t)}{\texttt{time}(t)}$ giving equal weight to all the three penalties (instead of 10 : 1 : 1 weight ratio used earlier) the optimal switching logic discovered with this cost metric are: $\mathtt{g}_{HF} : \texttt{temp} \geq 20.0 \wedge \texttt{out} = 16$, $\mathtt{g}_{FH} : \texttt{temp} \leq 18.8 \wedge \texttt{out} = 16$, $\mathtt{g}_{CF} : \texttt{temp} \leq 21.9 \wedge \texttt{out} = 26$, $\mathtt{g}_{FC} : \texttt{temp} \geq 22.7 \wedge \texttt{out} = 26$. We observe that the room temperature oscillates closer to the target temperature when the discomfort penalty is given relatively higher weight in the cost metric. This case-study illustrates that a

designer can suitably define a cost metric which reflects their priorities and, then, our technique can be used to automatically synthesize switching logic for the given cost metric.

7.2 Oil Pump Controller

Our second case study is an Oil Pump Controller. We use a simplified model of the industrial case study in [5]. The example consists of three components - a machine which consumes oil in a periodic manner, a reservoir containing oil, an accumulator containing oil and a fixed amount of gas in order to put the oil under pressure, and a pump. The simplification we make is to use a periodic function to model the machine's oil consumption and we do not model any noise (stochastic variance) in oil consumption.

The state variable is the volume V of oil in the accumulator. The system has two modes: mode `ON` when the pump is ON and mode `OFF` when the pump is OFF. Let the rate of consumption of oil by the machine be given by $m = 3 * (cos(t) + 1)$ where t is the time. The rate at which oil gets filled in the accumulator is p. $p = 4$ when the pump is on and $p = 0$ when the pump is off. The change in volume of oil in the accumulator is given by the following equation $\dot{V} = p - m$ where p and m take different values depending on the mode of operation of the pump. For synthesis, we consider two different sets of requirements [5].

In the first set of requirements, the volume of oil in the tank must be within some safe limit, that is, $1 \leq V \leq 8$ and the average volume of oil in the accumulator should be minimized. The gas pressure in the accumulator is directly proportional to the volume of oil and so, minimizing the average oil volume minimizes the average gas pressure in the accumulator. In order to model these requirements using our cost definition, we define one penalty variable p_1 and one reward variable r_1. Let the evolution of penalty p_1 be $\dot{p}_1 = V$ if $1 \leq V \leq 8$, M otherwise where M is a very large ($M \geq 10^5 p_1$) constant (10^6 in our experiments) and that of reward r_1 be $\dot{r}_1 = 1$. Minimizing the cost function $cost1 = \lim_{t\to\infty} \frac{p_1(t)}{r_1(t)}$ minimizes the average volume $\lim_{t\to\infty} \frac{\int_0^t V(t)}{t}$ and also enforces the safety requirement $\forall t \,.\, 1 \leq V(t) \leq 8$.

In the second set of requirements, we add an additional requirement to those in the first set. We require that the the oil volume is mostly below some threshold $V_{high} = 4.5$ in the long run. Practically, this models minimizing the cost of extra safety features that need to be activated when the volume is above this threshold. We model this requirement by adding an additional penalty and an additional reward variable p_2 and r_2 with evolution functions: $\dot{p}_2 = 1$ if $V > V_{high}$, 0 otherwise and $\dot{r}_2 = 1$ if $V < V_{high}$, 0 otherwise. The new cost function is $cost2 = \lim_{t\to\infty}(\frac{p_1(t)}{r_1(t)} + \frac{p_2(t)}{r_2(t)})$. Let t_{high} be the total duration when the volume is above V_{high} and t_{low} be the duration that it is below V_{high}. Minimizing $p_2/r_2 = t_{high}/t_{low}$ would ensure that we spend more time with volume less than V_{high} in the accumulator.

The guards: $\mathtt{g}_{FN}$ from `OFF` to `ON` and $\mathtt{g}_{NF}$ from `ON` to `OFF` obtained for the above $cost1$ objective are $\mathtt{g}_{FN} : V \leq 3.71$ $\mathtt{g}_{NF} : V \geq 4.62$ and for $cost2$ objective are $\mathtt{g}_{FN} : V \leq 4.07$ $\mathtt{g}_{NF} : V \geq 4.71$.

We simulate from an initial state $V = 4$ and the behavior for both objectives is presented in Figure 3. In both cases, the behavior satisfies the safety property that

the volume is within 1 and 8. Since, we minimize oil volume, the volume is close to the lower limit of 1. We also observe that using the second cost metric causes decrease in duration of time when oil volume is higher than the 4.5 but the average volume of oil increases. This illustrates how designers can use different cost metrics to better reflect their requirements.

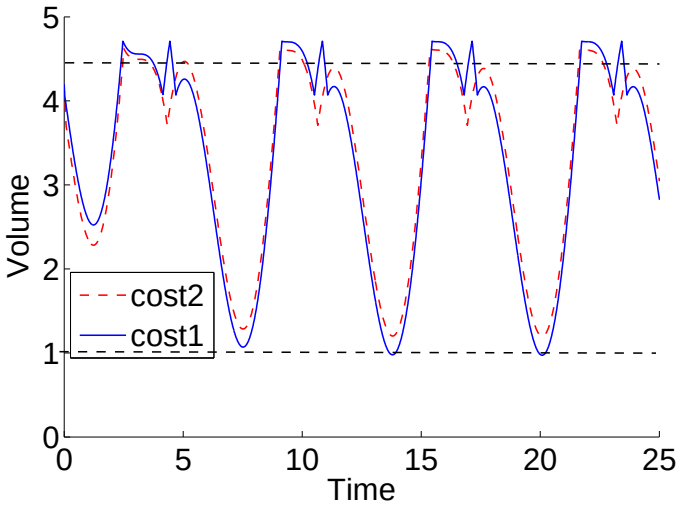

Fig. 3. Volume in accumulator

7.3 DC-DC Buck-Boost Converter

In this case study, we synthesize switching logic for controlling DC-DC buck-boost converter circuits described in [10]. The circuit diagram for the DC-DC converter is presented in Figure 4. The parameters used in the experiments are as follows: $V_d = 5V, C = 3.3\mu F, L = 47\mu H, rC = 0.06$ ohms, $rL = 0.1$ ohms, $rd = 0.05$ ohms, $rs = 0.05$ ohms. Input E applied is 10 volts and the target voltage at the load R is 5 volts. The load resistance periodically varies between $R = 100$ and $R = 200$ at intervals of 0.6 milliseconds.

A constant voltage E is applied to the converter which is expected to maintain the output voltage V_R across the variable load R at some target voltage V_d. The converter can be modeled as a hybrid system with three modes of operation. The state space of the system is $X = \langle i_L \ u_C \rangle$ where i_L is the current through the inductor and u_C is the voltage across the capacitor. The output voltage V_R can be computed from the values of i_L and u_C for a given mode using the circuit parameters. The load changed periodically between $R = 100$ and $R = 200$ at intervals of 0.6 milliseconds. The dynamics in the three modes are given by the

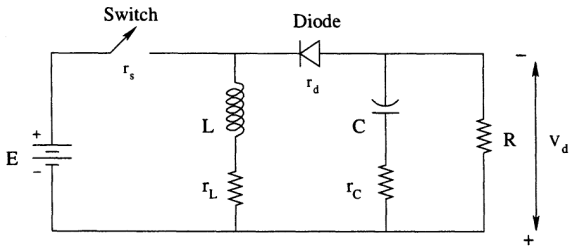

Fig. 4. Circuit Diagram for DC-DC Boost Converter

state space equation $\dot{X} = A_k X + B_k E$ where $k = 1, 2, 3$ is the mode and E is the input voltage. The coefficients of the equations are

$$A_1 = \begin{bmatrix} \frac{-rL-rs}{L} & 0 \\ 0 & \frac{-1}{C*(R+rC)} \end{bmatrix}, \quad B_1 = \begin{bmatrix} \frac{1}{L} \\ 0 \end{bmatrix}, \quad B_2 = \begin{bmatrix} \frac{1}{L} \\ \frac{R}{(R+rC)*\frac{rC}{L}} \end{bmatrix}, \quad A_3 = \begin{bmatrix} 0 & 0 \\ 0 & \frac{-1}{(R+rC)*C} \end{bmatrix},$$

$$A_2 = \begin{bmatrix} \frac{-rL-rd}{L} & \frac{-1}{L} \\ \frac{R}{R+rC} * (\frac{1}{C} - \frac{rC*(rL+rd)}{L}) & \frac{-R}{R+rC} * (\frac{rC}{L} + \frac{1}{R*C}) \end{bmatrix}, \quad B_3 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

The inductor acquires and stores the energy from the source in the first mode. In the second mode, the stored energy is transferred to the capacitor and the load. In the third mode, the excess energy of the capacitor is transferred to the load. We mention two key performance requirements of the DC-DC Boost Converter described in [10]. The first requirement is that the converter is resilient to load variations so that the transient response to change in load is within some limits and the system does not enter some unstable or chaotic region. The second requirement is to keep the variance of the voltage across the load V_R from the target voltage. This variance is called the ripple voltage. We define penalty variable p_1 with the following evolution functions: $\dot{p}_1 = (V_R - V_d)^2$. We want to minimize the average deviation from the target voltage. So, we define the reward variable r_1 with $\dot{r}_1 = 1$. The cost function is $\texttt{cost} = \lim_{t\to\infty} \frac{p_1(t)}{r_1(t)}$. This minimizes the average variance of V_R from the target voltage V_d. This corresponds to minimizing the ripple voltage. Since the load also changes periodically, it also minimizes the transient variance in voltage.

Given the dynamics in each of the three modes and the cost function, the synthesis problem is to automatically synthesize the guards $\mathtt{g}_{12}, \mathtt{g}_{23}, \mathtt{g}_{31}$ which minimizes the cost. We are given the over-approximation of the guard $\mathtt{g}_{23}^{over} : i_l = 0$. The guards obtained are as follows: $\mathtt{g}_{12} : i_L > 1.9$, $\mathtt{g}_{23} : i_L = 0$ and $\mathtt{g}_{23} : v_C > 4.6$. The system remains in the first mode until the inductor current reaches the reference current I_{ref}. The system remains in the second mode until the inductor current becomes 0. Then, the system switches to the third mode where it remains as long as the capacitor voltage remains over the reference voltage V_{ref}. We simulate the synthesized system and the behavior is shown in Figure 5.

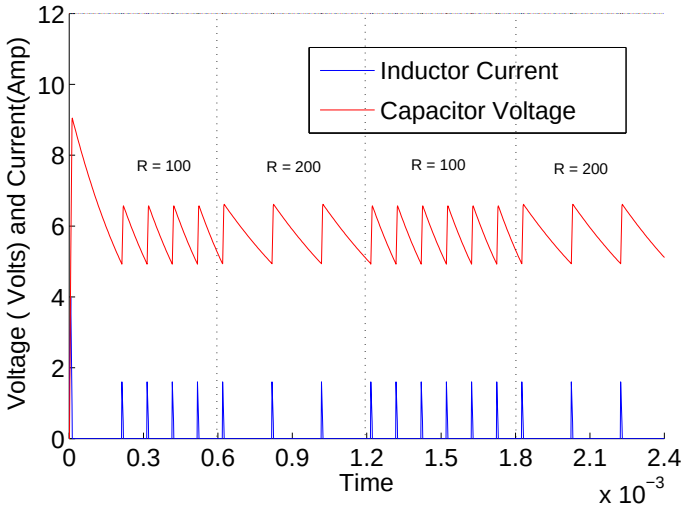

Fig. 5. DC-DC Boost Coverter

8 Conclusion

In this paper, we present an algorithm for automated synthesis of switching logic in order to achieve minimum long-run cost. Our algorithm is based on reducing the switching logic synthesis problem to an unconstrained numerical optimization problem which can then be solved by existing optimization techniques.

References

1. E. Asarin, O. Bournez, T. Dang, O. Maler, and A. Pnueli. Effective synthesis of switching controllers for linear systems. *Proc. of the IEEE*, 88(7):1011–1025, 2000.
2. H. Axelsson, Y. Wardi, M. Egerstedt, and E. Verriest. Gradient descent approach to optimal mode scheduling in hybrid dynamical systems. *J. Optimization Theory and Applications*, 136(2):167–186, 2008.
3. J. Bonnans, J. Gilbert, C. Lemaréchal, and C. Sagastizábal. *Numerical Optimization – Theoretical and Practical Aspects, Second Edition*. 2006.
4. M. Branicky, V. Borkar, and S. Mitter. A unified framework for hybrid control: Model and optimal control theory. *IEEE trans. on automatic control*, 43(1):31–45, 1998.
5. F. Cassez, J. J. Jessen, K. G. Larsen, J.-F. Raskin, and P.-A. Reynier. Automatic synthesis of robust and optimal controllers - an industrial case study. In *HSCC*, pages 90–104, 2009.
6. K. Chatterjee. Markov decision processes with multiple long-run average objectives. In *FSTTCS*, pages 473–484, 2007.
7. K. Chatterjee, R. Majumdar, and T. A. Henzinger. Controller synthesis with budget constraints. In *HSCC*, pages 72–86, 2008.
8. J. Cury, B. Brogh, and T. Niinomi. Supervisory controllers for hybrid systems based on approximating automata. *IEEE Trans. Automatic Control*, 43:564–568, 1998.

9. H. Gonzalez, R. Vasudevan, M. Kamgarpour, S. S. Sastry, R. Bajcsy, and C. J. Tomlin. A descent algorithm for the optimal control of constrained nonlinear switched dynamical systems. In *HSCC*, pages 51–60, 2010.
10. P. Gupta and A. Patra. Super-stable energy based switching control scheme for dc-dc buck converter circuits. In *ISCAS (4)*, pages 3063–3066, 2005.
11. R. M. Karp. A characterization of the minimum cycle mean in a digraph. *Discrete Mathematics*, 23:309–311, 1978.
12. T. Koo and S. Sastry. Mode switching synthesis for reachability specification. In *Proc. HSCC 2001*, LNCS 2034, pages 333–346, 2001.
13. A. Kucera and O. Strazovsky. On the controller synthesis for finite-state markov decision processes. *Fundamenta Informaticae*, 82(1-2):141–153, 2008.
14. J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright. Convergence properties of the nelder-mead simplex method in low dimensions. *SIAM Journal of Optimization*, 9:112–147, 1998.
15. P. Manon and C. Valentin-Roubinet. Controller synthesis for hybrid systems with linear vector fields. In *Proc. IEEE Intl. Symp. on Intelligent Control/Intell. Systems and Semiotics*, pages 17–22, 1999.
16. MathWorks. Find minimum of unconstrained multivariable function using derivative-free method. `http://www.mathworks.com/help/techdoc/ref/fminsearch.html`.
17. MathWorks. Solve initial value problems for ordinary differential equations. `http://www.mathworks.com/help/techdoc/ref/ode23.html`.
18. T. Moor and J. Raisch. Discrete control of switched linear systems. In *Proc. Eur. Control Conf. ECC'99*, 1999.
19. J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, January 1965.
20. S. S. Sastry. *Nonlinear Systems: Analysis, Stability, and Control*. Springer-Verlag, 1999.
21. M. Shaikh and P. Caines. On the optimal control of hybrid systems: Optimization of trajectories, switching times, and location schedules. In *HSCC*, pages 466–481, 2003.
22. P. Tabuada. Controller synthesis for bisimulation equivalence. *Systems and Control Letters*, 57(6):443–452, 2008.
23. C. Tomlin, L. Lygeros, and S. Sastry. A game-theoretic approach to controller design for hybrid systems. *Proc. of the IEEE*, 88(7):949–970, 2000.
24. X. Xu and P. Antsaklis. Optimal control of switched systems via nonlinear optimization based on direct differentiation of value functions. *Intl. journal of control*, 75(16):1406–1426, 2002.

A Limit Behaviors

We first discuss the possible limit behavior of a bivariate nonlinear system using a simple example of a violin string [20]. Using a bivariate system allows us to represent the limit behavior using 2-dimensional phase plots. We use linearization around the equilibrium point to analyze the limit behavior. The limit behaviors of nonlinear systems and hybrid systems are more diverse than linear systems. The stable limit behavior can be cyclic or converging.

The violin string can be described as a second order system using the following equation of motion.

$$M\ddot{x} + kx + F_b(\dot{x}) = 0$$

The state of the system is described using 2 variables: $x_1 = x, x_2 = \dot{x}$ and the evolution of the system is described as follows:

$$\dot{x}_1 = x_2 = f_1(x_1, x_2)$$
$$\dot{x}_2 = \frac{1}{M}(-kx_1 - F_b(x_2)) = f_2(x_1, x_2)$$

where $F_b(x_2) = -((x_2 - b) + c)^2 - d$ and the constant values are chosen as

$$M = 3, k = 3, b = 1, c = 2, d = 3$$

At equilibrium point (x_1^*, x_2^*), the left-hand side of the evolution should be zero. So,

$$\dot{x}_1^* = x_2^* = 0$$
$$\dot{x}_2^* = \frac{1}{M}(-kx_1^* - F_b(x_2^*)) = 0$$

Using the given values, $x_1^* = \frac{4}{3}, x_2^* = 0$.

Jacobean linearization around (x_1^*, x_2^*) can be used to analyze the nature of the limit behavior at equilibrium.

$$Df = \begin{bmatrix} \partial f_1/\partial x_1 & \partial f_1/\partial x_2 \\ \partial f_2/\partial x_1 & \partial f_2/\partial x_2 \end{bmatrix}$$

To analyze the limit behavior,

$$|Df|_{x^*} - \lambda I| = 0$$

, the nature of the eigen-values (λ) can be used to accurately predict the limit behavior for *linear systems*. The following cases are possible for a linear system:

- If both eigen-values are real and negative, the equilibrium is a *STABLE NODE*, that is, the system converges to the equilibrium.
- If both eigen-values are real and positive, the equilibrium is a *UNSTABLE NODE*, that is, the system will diverge.
- If real part of one eigen-value is positive and another is negative, the equilibrium is a *SADDLE*, that is, it diverges from all initial states except a line.

– If the eigen-values are complex and both real parts are positive, the equilibrium is a *STABLE FOCUS*, that is the system converges to the equilibrium going through a damping cycle.
– If the eigen-values are complex and both real parts are negative, the equilibrium is a *UNSTABLE FOCUS*, that is the system diverges through increased oscillations.
– If the eigen-values are purely imaginary, then the equilibrium is a *CENTER* and the system enters a cycle.

Thus, the *stable* limit behavior for a linear system is either converging or cycling. For a non-linear system, linearization only provides a hint regarding the limit behavior. It is possible to enter limit cycles for a non-linear system even if the linearization predicts an *UNSTABLE FOCUS*. Analytical techniques are not sufficient to predict limit behaviors of nonlinear and hybrid systems. Simulation is used to find the limit behaviors. But all *STABLE* behaviors are always either converging or cyclic.

In our example of voilin strings, the eigen values for the given parameters are

$$\lambda = \frac{1}{3} \pm \frac{2\sqrt{2}}{3} j$$

Thus, linearization predicts that the system would enter an unstable focus but simulation shows that the system enters a limit cycle from any initial state other than $(\frac{4}{3}, 0)$. If the system starts from $(\frac{4}{3}, 0)$, it stays at the initial state. This illustrates the phenomenon of *BIFURCATION*, that is, different initial states show different limit behaviors in non-linear and hybrid systems.

B Constrained Optimization Example

We recall a standard procedure to solve constrained optimization problem with a small example. This technique is used to solve the optimization problem with the Equation 5. Consider the following example,

$$\begin{aligned} \min_{x,y} \quad & \frac{1}{x^2 + y^2} \\ \textit{subject to} \quad & x + y = 10 \end{aligned}$$

The above optimization problem can be transformed into the following equivalent optimization problem without constraints by suitably modifying the optimization objective.

$$\begin{aligned} \min_{x,y} \quad & \frac{1}{x^2 + y^2} + M(x + y - 10)^2 \\ \text{where } M & \text{ is any positive number,} \\ & \texttt{say } M = 100 \end{aligned}$$