Sound Test-Suites for Cyber-Physical Systems

Morteza Mohaqeqi
Department of Information Technology
Uppsala University, Sweden
Email: morteza.mohaqeqi@it.uu.se

Mohammad Reza Mousavi
Centre for Research on Embedded Systems,
School of IT, Halmstad University, Sweden
Email: m.r.mousavi@hh.se

***Abstract*—Conformance testing is a formal and structured approach to verifying system correctness. We propose a conformance testing algorithm for cyber-physical systems, based on the notion of hybrid conformance by Abbas and Fainekos. We show how the dynamics of system specification and the sampling rate play an essential role in making sound verdicts. We specify and prove error bounds that lead to sound test-suites for a given specification and a given sampling rate.**

I. Introduction

A. Background

The idea of model-based testing (MBT) [1] is to systematically generate test-cases from a test model, i.e., a specification of system's correct behavior. A rigorous notion of MBT aims at establishing a *conformance relation* by running a number of such generated test-cases [2], [3], [4], [5], [6]. Conformance relation is typically defined on a common semantic domain of both the test model and the system under test. We refer to such a mathematically-founded notion of MBT as *conformance testing*. Figure 1 provides a schematic view of the notions of conformance relation, conformance testing and the relation between them.

The common semantic domain is used to facilitate specifying and reasoning about the conformance relation, but in practice, one does not typically have access to such a rigorous description of behavior, particularly of the system under test. Hence, conformance testing should ideally precisely characterize the conformance relation, denoted by the downward and upward arrows in Figure 1 representing soundness and exhaustiveness, respectively.

Some notions of conformance testing for ordinary reactive systems are both sound and exhaustive [7], [8]. However, for cyber-physical systems (CPSs), it is far from trivial to come up with a sound and exhaustive, yet practical, notion of conformance testing. Conformance testing of CPSs often involves discrete sampling of continuous signals and hence, for any realistic notion of conformance testing, error margins should be accommodated to allow for slight deviations (e.g., measurement errors) in time and value [9], [2], [3]. The interaction among the continuous dynamics, the sampling rates and the error margins is an intricate one and if the aforementioned parameters are not in sync, the resulting conformance testing method can be unsound. Exhaustiveness is even more intricate and requires detailed information about the continuous dynamics of the implementation under test (as well as the specification); we only address exhaustiveness briefly and in passing towards the end of this paper.

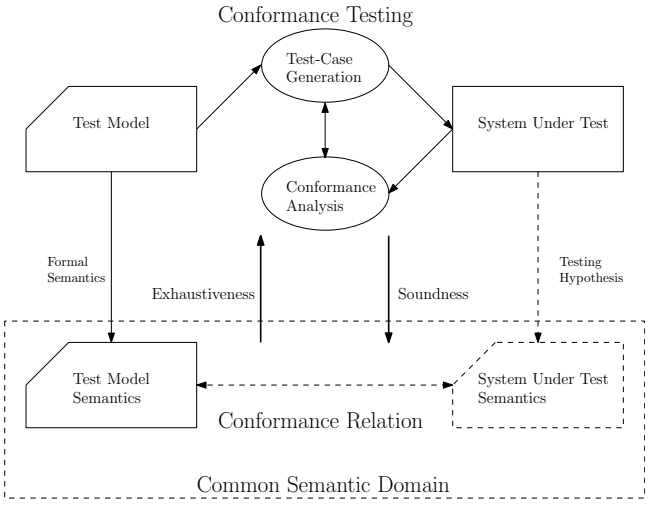


Fig. 1: Schematic View of Conformance Testing and Conformance Relation

B. Problem definition

Given a specification and a sampling rate, we seek sufficient conditions for a test-suite under which, conformance testing is sound with respect to a given conformance relation. That is, the test-suite only fails on non-conforming implementations.

To make this problem more specific, we take the notion of hybrid conformance by Abbas and Fainekos [9], [2], [3] as our conformance relation. We then define a straightforward conformance testing algorithm based on this notion and study its soundness. As it turns out, for all reasonable specifications such a conformance testing algorithm may in general result in unsound verdicts. Hence, we specify and prove soundness criteria, based on the error margins (in time and value), the properties of specification's continuous dynamics, and the sampling rate, that guarantee soundness of the test verdicts.

C. Running Example

To illustrate different notions, we consider the model of a thermostat [6] as our running example. The thermostat has two operation modes to control the temperature. In mode *ON*, a heater is turned on in order to warm up the environment.

During the mode *OFF*, the heater is turned off, which leads to a steady decrease in the environment temperature. There is a threshold for the minimum environment temperature, which triggers the thermostat to switch to mode *ON*. A similar threshold exists for the maximum temperature which makes the thermostat to switch to mode *OFF*.

D. Organization

The rest of this paper is organized as follows. In Section II, we review some notions from the literature concerning hybrid system specification and hybrid conformance. In Section III, we define our notion of conformance testing for hybrid conformance. In Section IV, we first show that in all practical cases, conformance testing can lead to unsound test verdicts and subsequently, define sufficient conditions on test-suites to produce sound verdicts. In Section V, we review the related work. Finally, we conclude the paper and present the directions of our ongoing research in Section VI.

II. Preliminaries

In the remainder of this paper, $\mathbb{N}$, $\mathbb{R}$, and $\mathbb{R}_+$ denote the set of non-negative integers, real numbers, and non-negative real-numbers, respectively. Consider a set of real-valued variables V. A valuation of V is a function of type $V \mapsto \mathbb{R}$, which assigns a real number to each variable $v \in V$. The set of all valuations of V is denoted by $Val(V)$. Further, the domain of a function f is denoted by $\mathrm{dom}(f)$.

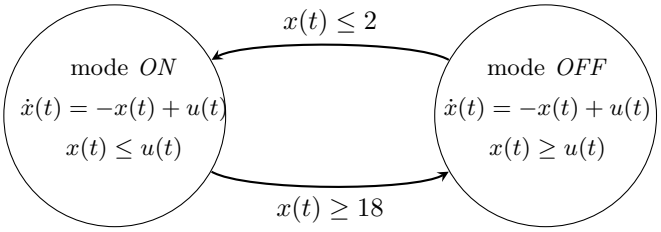
In the following subsections, we first formally specify the notion of hybrid systems and the corresponding concepts. Then, we elaborate a formal definition of a conformance relation for hybrid systems.

A. Hybrid System Specifications

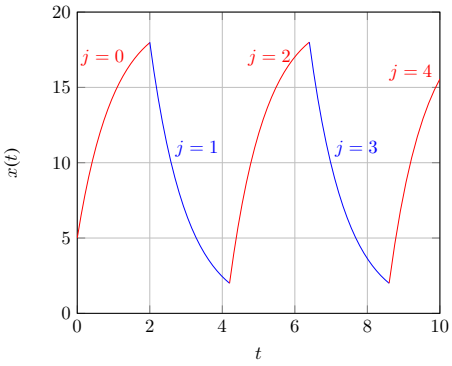
To specify test models for hybrid systems, we use the Hybrid Automata formalism, defined below.

Definition 1 (Hybrid Automata [10])**.** *A hybrid automaton is defined as a tuple (Loc, V, (l_0,v_0), $\rightarrow$, I, F), where*

- *Loc is the finite set of locations;*
- *$V = V_I \uplus V_O$ is the set of continuous variables, where V_I and V_O denote the disjoint sets of input variables and output variables, respectively;*
- *l_0 denotes the initial location and v_0 is an initial valuation of V;*
- *$\rightarrow \subseteq Loc \times \mathcal{B}(V) \times Reset(V) \times Loc$ is the set of jumps where:*
  - *$\mathcal{B}(V) \subseteq Val(V)$ indicates the guards under which the jump may be performed, and*
  - *$Reset(V) = \bigcup_{V' \subseteq V} Val(V')$ is the set of value assignments to the variables in V after the jump;*
- *$I : Loc \rightarrow \mathcal{B}(V)$ determines the allowed valuation of variables in each location (called the invariant of the location);*
- *$F : Loc \rightarrow \mathcal{B}\left(V \cup \dot{V}\right)$ describes some constraints on variables and their derivatives and specifies the allowed continuous behavior in each location.*



(a) Hybrid automaton of the thermostat



(b) A sample of the continuous dynamics of the system

Fig. 2: Thermostat example

We denote the set of all hybrid automata by $\mathbb{H}$. We typically write $l \xrightarrow{g,r} l'$ to denote $(l, g, r, l') \in \rightarrow$.

Example 1. *Fig. 2a shows the hybrid automaton of the thermostat described in Section I-C with $V_I = \{u\}$, $V_O = \{x\}$, and $(l_0, v_0) = (ON, 5)$.*

The evolution of a hybrid system is defined over a domain of *hybrid* time, defined below.

Definition 2 (Hybrid Time Domain [9])**.** *A hybrid time domain E is a subset of $\mathbb{R}_+ \times \mathbb{N}$ defined as*

$$E = \bigcup_{j=0}^{J-1} [t_j, t_{j+1}] \times \{j\} \tag{1}$$

where $0 = t_0 \leq t_1 \leq t_2 \leq ... \leq t_J$. We denote the set of all hybrid time domains by $\mathbb{T}$.

The hybrid time domain is used to model the evolution of a hybrid system regarding both evolution of system dynamics (using continuous time intervals $[t_j, t_{j+1}]$) and discrete jumps (using integer numbers j).The following notion of solution gives a semantics to hybrid automata using the notion of hybrid time domain.

Definition 3 (Solution)**.** *A solution to a hybrid automaton $HA = (Loc, V, (l_0, v_0), \rightarrow, I, F)$ is a function $s : \mathbb{T} \rightarrow Loc \times Val(V)$, where*

- *$s(0, 0) = (l_0, v_0)$;*
- *for each $(t, j) \in \mathrm{dom}(s)$: x satisfies $I(l)$ and $F(l)$, where $(l, x) = s(t, j)$; and*

- *for each* $(t_j, j) \in \mathrm{dom}(s)$ *with* $j > 0$*: there exists* $l \xrightarrow{g,r} l'$ *such that* x *satisfies* g *and* (x, x') *satisfies* r*, where* $(l, x) = s(t_j, j-1)$ *and* $(l', x') = s(t_j, j)$.

In order to capture the evolution of system dynamics and abstract away from the internal discrete states (i.e., locations), we use the following notion of *trajectory*.

Definition 4 (Trajectory [9][1])**.** *Take a hybrid time domain* E *and a set of variables* V*. A trajectory over* E *is a function* $\phi : E \to Val(V)$*, where for every* j*,* $t \mapsto \phi(t, j)$ *is absolutely continuous in* t *over the interval* $I_j = \{t | (t, j) \in E\}$*. The set of all trajectories defined over the variable set* V *is denoted by* $Trajs(V)$.

Definition 5 (Trajectory for Hybrid Automata)**.** *Given a hybrid automaton HA, a trajectory* $\phi : E \to Val(V)$ *is a trajectory for HA, if there exists some solution* s *to HA for which* $\forall (t, j) \in E$*,* $\exists l \in Loc$ *such that* $(l, \phi(t, j)) = s(t, j)$.

To discriminate input trajectories from output trajectories in a solution, the notion of *solution pair* is defined next. To this aim, we first define the notion of trajectory restriction.

Definition 6 (Trajectory Restriction [5])**.** *Consider a set of variables* V*. The restriction of a valuation* $val \in Val(V)$ *to* $V' \subset V$*, denoted by* $val \downarrow V'$*, is a valuation* $val' \in Val(V')$ *such that* $val' \downarrow V'(v) = val(v)$*,* $\forall v \in V'$*. Further, the restriction of a trajectory* $\phi : E \to Val(V)$ *to* $V' \subset V$ *is a trajectory* $E \to Val(V')$*, denoted by* $\phi \downarrow V'$*, for which* $(\phi \downarrow V')(t, j) = \phi(t, j) \downarrow V'$*,* $\forall (t, j) \in \mathrm{dom}(\phi)$.

Example 2. *Figure 2b shows a trajectory to the thermostat hybrid automaton over a hybrid time domain of* $E = ([0, 2], 0) \cup ([2, 4.2], 1) \cup ([4.2, 6.4], 2) \cup ([6.4, 8.6], 3) \cup ([8.6, 10], 4)$ *after restriction to* $\{x\} \subset V$.

Definition 7 (Solution Pair [9])**.** *Let* u *and* y *be two trajectories;* (u, y) *is a solution pair to a hybrid automaton* $\mathcal{H}$ *if*

- $\mathrm{dom}(u) = \mathrm{dom}(y)$*, and*
- *there exists a trajectory* ϕ *to* $\mathcal{H}$ *such that* $\mathrm{dom}(\phi) = \mathrm{dom}(u)$*,* $u = \phi \downarrow V_I$*, and* $y = \phi \downarrow V_O$.

Note that by requiring $\mathrm{dom}(\phi) = \mathrm{dom}(u) = \mathrm{dom}(y)$, we make sure that the solution and its input and output pairs are all defined on the same hybrid time domain.

For an example, consider a trajectory u over the hybrid time domain specified in Example 2 where $u(t, j) = 20$ for $j \in \{0, 2, 4\}$ and $u(t, j) = 0$ for $j \in \{1, 3\}$. Also, let y be the trajectory described in that example. then, the pair (u, y) constitute a solution pair to the respective hybrid automaton.

To simplify the forthcoming developments in the current work, we focus on deterministic hybrid automata, defined below. The extension to the non-deterministic case is straightforward and requires iterating over all possible output solutions for a given input. Some initial ideas to this effect are provided in [3].

[1]In [9], the term *Hybrid Arc* is used to refer to a trajectory.

Definition 8 (Deterministic Hybrid Automata)**.** *A hybrid automaton* $\mathcal{H}$ *with the set of solution pairs* Φ *is deterministic if*

$$((u, y_1) \in \Phi \text{ and } (u, y_2) \in \Phi) \Rightarrow y_1 = y_2 \tag{2}$$

In this case, we write $y = out_{\mathcal{H}}(u)$ *to denote* $(u, y) \in \Phi$.

B. Conformance Relation

To define a conformance relation, we assume that both the specification and the purported underlying semantics of the implementation can be captured by some hybrid automata. We use the notion of (τ,ϵ)-closeness, which is defined on the continuous behavior (solution) associated to a hybrid automaton. (We abstract away from the number of discrete jumps, as we consider them irrelevant regarding the observable behavior of the system.)

Definition 9 ((τ,ϵ)-closeness [9])**.** *Consider a test duration* $T \in \mathbb{R}_+$*, a maximum number of jumps* $J \in \mathbb{N}$*, and* $\tau, \epsilon > 0$*; then two trajectories* y_1 *and* y_2 *are said to be* (τ,ϵ)*-close, denoted by* $y_1 \approx_{(\tau,\epsilon)} y_2$*, if*

1) *for all* $(t, i) \in \mathrm{dom}(y_1)$ *with* $t \leq T, i \leq J$*, there exists* $(s, j) \in \mathrm{dom}(y_2)$ *such that* $|t - s| \leq \tau$ *and* $\|y_1(t, i) - y_2(s, j)\| \leq \epsilon$*, and*
2) *for all* $(t, i) \in \mathrm{dom}(y_2)$ *with* $t \leq T, i \leq J$*, there exists* $(s, j) \in \mathrm{dom}(y_1)$ *such that* $|t - s| \leq \tau$ *and* $\|y_2(t, i) - y_1(s, j)\| \leq \epsilon$. [2]

Definition 10 (Conformance Relation [9])**.** *Consider two hybrid automata* $\mathcal{H}_1$ *and* $\mathcal{H}_2$*. Given a test duration* $T \in \mathbb{R}_+$*, a maximum number of jumps* $J \in \mathbb{N}$*, and* $\tau, \epsilon > 0$*,* $\mathcal{H}_2$ conforms to $\mathcal{H}_1$*, denoted by* $\mathcal{H}_2 \approx_{(\tau,\epsilon)} \mathcal{H}_1$*, if and only if for all solution pairs* (u, y_1) *of* $\mathcal{H}_1$*, there exists a solution pair* (u, y_2) *of* $\mathcal{H}_2$ *such that the corresponding output trajectories* y_1 *and* y_2 *are* (τ,ϵ)*-close.*

III. Conformance Testing

The conformance relation defined in the previous section assumes access to the solutions of the hybrid automaton underlying the implementation. This is not a realistic assumption in practice. To remedy this, in this section, we define a notion of conformance testing that instead uses sampling of the specification solution to test the implementation outputs at various discrete points. Such a notion of conformance testing should at least be sound with respect to the conformance relation and ideally should coincide with (i.e., also be exhaustive).

To start with, we define below a sampling mechanism, which involves sampled sequences of the continuous (output) signals.

Definition 11 (Hybrid-Timed State Sequence (TSS) [2])**.** *Let* $N \in \mathbb{N}$ *and* V *be a set of variables. A hybrid-timed state sequence (TSS) is defined as function* $x : \mathbb{R}_+ \times \mathbb{N} \to Val(V)$*, with* $\mathrm{dom}(x) \in (\mathbb{R}_+ \times \mathbb{N})^N$*. The value of function* x *at a*

[2]Unlike [9], here we allow different jump numbers (i.e. $i \neq j$) in the definition of (τ,ϵ)-closeness.

specific point $(t, j) \in \mathrm{dom}(x)$ is denoted by $x(t, j)$. Also, we denote the set of all TSSs defined over the set of variables V by $\mathbb{TSS}(V)$.

Definition 12 (Sampling Function)**.** *Consider $N \in \mathbb{N}$. Any $P \in (\mathbb{R}_+ \times \mathbb{N})^N$ is called a set of sampling points. Take a set of trajectories Y with a set of variables V. Given a set of sampling points P, a sampling function over Y is defined as $\pi_P : Y \mapsto \mathbb{TSS}(V)$, for which, $y_s = \pi_P(y)$ only if*

- $\mathrm{dom}(y_s) = \mathrm{dom}(y) \cap P$
- $\forall (t, j) \in \mathrm{dom}(y_s) : y_s(t, j) = y(t, j)$

A sampling function is periodic when its sampling points are equally distanced. In other words, a sampling function with the set of sampling points P is periodic with period p if and only if $\forall (t_1, j), (t_2, k)$ with $t_1 < t_2$, we have

$$(\nexists (t, l) \in P : t_1 < t < t_2) \Rightarrow t_2 - t_1 = p \tag{3}$$

Further, in this situation, P is called a periodic set of sampling points with period p.

Next we define the notions of test-suite and test-case. They involve providing an input trajectory and a sampling function and providing the expected output valuations at the specified sampling points.

Definition 13 (Test-Suite and Test-Case)**.** *A test-suite is defined as a finite set $TS \subset Trajs(V_I) \times \mathbb{TSS}(V_O)$. A test-suite TS is a valid one for a given hybrid automaton $\mathcal{H}$ only if, for any $(u, y) \in TS$ there exists a sampling point set P such that $y = \pi_P(out_{\mathcal{H}}(u))$. Each member of a valid test-suite is called a test-case.*

It is worth noting that in the above definition, we provide continuous inputs and observe sampled (discrete) outputs. This is because in practice, the system receives a continuous input.[3] However, the output behavior of a system is usually observed using a sampled signal [6], which provides a sequence of values in a number of discrete instants.

Based on the definition of periodic sampling points, we define a class of *periodic* test cases.

Definition 14 (Periodic Test-Case)**.** *A test-case (u, y) is periodic with period p if $\mathrm{dom(y)}$ is a periodic set of sampling points with period p.*

In this paper, we treat with periodic test-cases. However, the presented results can be extended to the general case.

Execution of a test case $tc = (u, y) \in TS$ on a system $\mathcal{H}_I$ consists of applying u to $\mathcal{H}_I$, obtaining the system output at the same sampling points of the TSS y, and making a decision on the correctness of $\mathcal{H}_I$.

Definition 15 (Test Verdict)**.** *For a given hybrid system HA and a test-suite TS, a test verdict function is a mapping $(TS, HA) \mapsto \{Pass, Fail\}$.*

[3]Note that even in discrete-time systems (e.g. a system equipped with a digital controller) a *hold* circuit is used [11], which leads to a continuous input to the system.

A class of test verdict functions is defined by Algorithm 1. A test verdict function defined by this algorithm compares the expected outcomes of TS with the solutions $\mathcal{H}_I$ at the sampling points of TS within the neighborhood of T and E in time and value, respectively.

In a test verdict, $Fail$ means that the system does not conform to the specification, while $Pass$ means that, using the considered test-suite, no evidence has been found to conclude that the system does not conform to the specification. We use Algorithm 1 as the the test verdict for conformance testing of cyber-physical systems.

Algorithm 1 Test Verdict

```
1: inputs: A test-suite TS; A hybrid automaton H_I; Conformance parameters T, E
2: output: Pass or Fail
3: for each (u, y) ∈ TS do
4:     y_I ← out_{H_I}(u)
5:     P ← dom(y)
6:     y_I^s ← π_P(y_I)
7:     for each (t, j) ∈ dom(y_I^s) do
8:         I_t = [t − T, t + T] ∩ {t | ∃j : (t, j) ∈ dom(y) }
9:         if ∃t' ∈ I_t s.t. ‖y(t', i) − y_I^s(t, k)‖ ≤ E then
10:            continue;
11:        else
12:            return Fail
13:        end if
14:    end for
15: end for
16: return Pass
```

Definition 16 (Soundness)**.** *Considering a specification $\mathcal{H}$, a test-suite TS is sound under a specified test verdict algorithm if the following proposition holds*

$$\forall \mathcal{H}_I : \left(\mathcal{H}_I \approx_{(\tau,\epsilon)} \mathcal{H}\right) \Rightarrow \mathcal{H}_I \ passes \ TS \tag{4}$$

IV. Sound Test-Suites

The aim of this section is to establish the conditions under which the soundness of a test-suite can be guaranteed when Algorithm 1 is used as the test verdict.

A. Unsoundness result

A straightforward method for (τ, ϵ)-conformance testing is to use Algorithm 1, considering τ and ϵ as the algorithm parameters T and E, respectively. However, the problem with this approach is that, for all practical specifications, it can produce unsound results, irrespective of the sampling period used in the test-suite. The following example illustrates this problem.

Example 3. *Consider the thermostat system described in Example I-C, with the hybrid automaton given in Figure 2a as its specification. Figure 3 shows an output trajectory obtained*

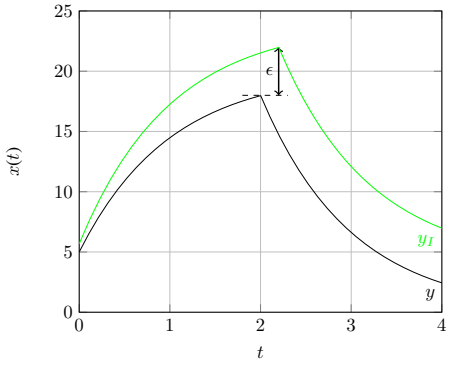


Fig. 3: The output trajectory of the thermostat specification (y) and that of a sample implementation (y_I)

from the specification (labeled as y), and an output trajectory of a conforming implementation (labeled as y_I). The trajectory y_I has been obtained by shifting y to the right by 0.2 and to the above by 4. Considering $\tau = 1 > 0.2$ and $\epsilon = 4$, it is seen that y_I satisfies the (τ, ϵ)-closeness condition. Assume that for (τ, ϵ)-conformance testing of the implementation, Algorithm 1 is used with $T = \tau$ and $E = \epsilon$. Also, assume a test-case containing a TSS obtained from y by a sampling function with sampling period of 0.03. If the set of sampling points P is selected such that $t = 2.2 \in P$, then $2 \notin P$ [4]. However, there is only one point in y, namely at time $t = 2$, which satisfies the closeness condition specified in line 9 of Algorithm 1. But under the described sampling function, this point is not included in the test-case; as a result, the implementation fails.

The following theorem generalizes the observation made in Example 3 to a large class of specifications (covering all thinkable practical cases). Namely, we define a general class of specifications $\mathbb{S}$ such that for each $\mathcal{H} \in \mathbb{S}$, there is no sampling period for which Algorithm 1 with parameter selection of $T = \tau$ and $E = \epsilon$ is guaranteed to be sound for all (τ, ϵ)-conforming implementations.

Theorem 1. *For a given $\tau > 0$, consider a class $\mathbb{S}$ of all specifications that exhibit at least one output trajectory, y, which satisfies the following property:*

- *There exists an interval $I = [s - \tau, s + \tau]$ in which y has a minimum (or maximum) at s and there are at most a countable number of minima (maxima) points in I.*

Also, for any $\mathcal{H} \in \mathbb{S}$, let $\mathbb{TS}_{\mathcal{H}}$ denote the set of all valid test suits for $\mathcal{H}$. Then, given $\tau, \epsilon > 0$, the following holds:

$$\forall \mathcal{H} \in \mathbb{S}, \forall\, TS \in \mathbb{TS}_{\mathcal{H}}, \exists \mathcal{H}_I \text{ such that}: \\ \mathcal{H}_I \approx_{\tau,\epsilon} \mathcal{H} \text{ and } \mathcal{H}_I \text{ fails } TS$$

Proof. We present a proof by construction. The intuition is based on a generalization of Example 3. Consider a test-suite containing a test-case with a sampling period of p and let M be the set of maxima (minima) described in the hypothesis of the theorem. Take a sample implementation which is obtained from y by shifting the values of its variables by ϵ to above (below), and its hybrid time axis by $\delta \leq \tau$ to the right such that

$$\forall (t', j) \in M : \frac{|t' + \delta - s|}{p} \notin \mathbb{N}, \tag{5}$$

As a result, when the sampling points are adjusted such that s is a sampling point, none of the mentioned maxima (minima) are included in the sampled TSS of the implementation output. Hence, the implementation is failed, because only the points in M could satisfy the closeness condition specified in line 9 of Algorithm 1. On the other hand, it can be easily seen that the described implementation conforms to the specification, which leads to the unsoundness of the test-suite. □

[4]We omit the jump index from a hybrid time instance for notation brevity

B. Reinstating Soundness

As mentioned before, the goal of this section is to guarantee the soundness of test-suites. To this end, we define the following measure on the specifications.

Definition 17 (Specification Maximum Change)**.** *Given a specification $\mathcal{H}$, a periodic test-case (u, y) for it with period $p > 0$, and a test duration T, the maximum change of $\mathcal{H}$ with respect to (u, y) and T is defined as $\Delta_p = \max_{t \in T} \Delta_p(t)$, where*

$$\Delta_p(t) = \max_{s \in [t - p/2, t + p/2]} y(s) - \min_{s \in [t - p/2, t + p/2]} y(s).$$

The following lemma paves the way for the subsequent soundness result. Namely, it states that if the maximum changes of the specification are confined, then by extending the error margins for output values, one can always obtain sound results.

Lemma 1. *Given $p \geq 0$, a test duration T, a specification $\mathcal{H}$, and an arbitrary output trajectory $y \in \mathbb{Y}$, we always have*

$$\forall \epsilon > 0, \forall c \in \mathbb{R}, \forall s_2 > s_1 \geq 0 : \text{ if } |s_1 - s_2| \leq p \text{ then} \\ (\|y(s_1) - c\| > \epsilon + \Delta_p \text{ and } \|y(s_2) - c\| > \epsilon + \Delta_p) \Rightarrow \\ \forall s \in (s_1, s_2) : \|y(s) - c\| > \epsilon \tag{6}$$

Proof. The proof is by contradiction. Assume that the left side of (6) holds, but the right side does not hold, namely $\exists s \in (s_1, s_2) : \|y(s) - c\| \leq \epsilon$, or equivalently,

$$-\|y(s) - c\| \geq -\epsilon \tag{7}$$

Summing up (7) with $\|y(s_1) - c\| > \epsilon + \Delta_p$ from (6) yields

$$\|y(s_1) - c\| - \|y(s) - c\| > \Delta_p \tag{8}$$

Besides, using the general rule $\|a - b\| \geq \|a\| - \|b\|$, we can conclude

$$\|y(s_1) - y(s)\| > \Delta_p \tag{9}$$

which implies in a maximum change of y larger than Δ_p in an interval of smaller than p. But this is inconsistent with the definition of Δ_p, which contradicts the initial assumption. □

Definition 18 (Robust Test-Suites)**.** *Given $\tau, \epsilon > 0$, assume that we use Algorithm 1 with parameter assignment of $T = \tau$*

and $E = \epsilon + \Delta$, where $\Delta > 0$. Then, given a specification $\mathcal{H}$, a test-case $tc = (u, y) \in \mathbb{TS}$ with a sampling period p is said to be robust if

$$\Delta \geq \Delta_p, \tag{10}$$

Theorem 2. *A robust test-case is always sound, according to the definition of soundness in Definition 16.*

Proof. The proof is by contradiction. Assume the theorem does not hold. Thus, there exists a system $\mathcal{H}_I$ conforming to a specification $\mathcal{H}$ which fails a robust test-case $tc = (u, y)$ with respect to a sampling period p. Let y_I be the sampled output trajectory of the system for the input u. Further, let $\phi = out_{\mathcal{H}}(u)$. Based on the mentioned assumption, Algorithm 1 returns *Fail* for y and y_I. As a result,

$$\exists t \in \mathrm{dom}(y_I) \ such\ that\ \forall t' \in [t-\tau, t+\tau] \cap dom(y) : \\ \|y_I(t) - y(t)\| > \epsilon + \Delta_p \tag{11}$$

(see lines 8-13 in Algorithm 1). According to Lemma 1, and based on the definition of Δ_p, (11) yields that

$$\forall t' \in [t-\tau, t+\tau] : \|y_I(t) - \phi(t)\| > \epsilon \tag{12}$$

which means that the implementation is not conforming to the specification. However, this result is in conflict with our assumption that $\mathcal{H}_I$ conforms to the specification $\mathcal{H}$, proving the theorem. □

It is worth noting that, in practice, an appropriate value for Δ_p, and thus, for Δ can be determined based on the formal specification of the system. For instance, the evolution of the system may be specified using a hybrid automaton with a number of differential equations without a discrete jump in the values. In this case, if such equations yield a bounded derivative for the respective functions, then the maximum value of the derivative can be used as a valid value for Δ_p.

C. Towards Exhaustiveness

In order for a test-suite to be exhaustive, it should fail each and every nonconforming implementation.

Definition 19 (Exhaustiveness)**.** *A test-suit TS is said to be exhaustive if*

$$\forall \mathcal{H}_I : \left(\mathcal{H}_I \not\approx_{(\tau,\epsilon)} \mathcal{H}\right) \Rightarrow \mathcal{H}_I \ fails \ TS$$

To this end, an implementation must pass a test suit only if it is conforming. However, the approach discussed up to now has the following two shortcomings to achieve this goal.

First, according to Definition 9, two criteria must be satisfied by an implementation to make sure that it conforms to the specification. However, the test verdict algorithm only checks one of them (lines 7 to 14 in Algorithm 1). In fact, for each sampled point in the implementation, the method checks the possibility of the satisfaction of the closeness condition, and the implementation fails if such possibility is not established. For an exhaustive test, however, we need to check also other way around. For this goal, a similar information (namely maximum change as defined in Definition 17) should be available for the implementation trajectory. However, in this manner, the conformance testing cannot be seen as a black box technique any more and some inside information about the implementation will be required.

Second, Definition 9 calls for the conditions over all points in a continuous interval. However, it is not possible to concretely check all the points as there are uncountably many points in a given continuous interval. To resolve this issue, we need to check a *restrictive* condition on the tested points such that we can conclude some result for the other points in their vicinity, including the untested ones. For this goal, our conjecture is that, using Algorithm 1 with a parameter assignment of $T = \tau - p/2$ and $E = \epsilon - \Delta_{p/2}$ provides an exhaustive test verdict method. This is based on the following observation.

Observation 1. *Let y and y_I be two trajectories. Then, the following holds for any $t \in \mathrm{dom}(y)$:*

$$\begin{aligned} &if\ \|y_I(t_I) - y(t)\| \leq \epsilon - \Delta_{\frac{p}{2}} \\ &\qquad for\ some\ t_I \in [t - \tau + \frac{p}{2}, t + \tau - \frac{p}{2}], \\ &\qquad then\ \ \forall t' \in [t - \frac{p}{2}, t + \frac{p}{2}] : \|y(t') - y_I(t_I)\| \leq \epsilon \end{aligned}$$

This observation states that if the two trajectories are sufficiently close to each other in two points, then they would be so for the other neighboring points (provided that the variation of the dynamics in the implementation is bounded in the same manner as the specification).

To summarize, once the two sets of criteria are met, we require two runs of conformance testing (possibly run simultaneously) with two different bounds in the two respective directions to guarantee soundness and completeness.

V. Related work

Conformance testing for cyber-physical systems has been approached from different perspectives. We refer to [5], [12] for comparisons of some of these approaches. Inspired by the notion of input/output conformance relation (ioco) [13], which is used for discrete-event systems, van Osch [14], [6] proposed a notion of hybrid input output conformance (hioco). He uses the hybrid labeled transition systems [15] formalism in order to specify the hioco relation. Intuitively, a system under test is conforming to a specification based on hioco if all possible behaviors of the implementation is a subset of behaviors allowed by the specification.

From a control-theoretic perspective, Abbas et al. proposed a more practical notion of conformance relation for cyber-physical systems [9], [2], which is used in this paper. Their conformance relation is defined based on the notion of time and value closeness. Their conformance relation allows for the implementation to deviate from the specification behavior to some extent, while it is still regarded as a conforming implementation. In [16], we take the first step towards unifying the approach of Abbas et al. with that of van Osch by interpreting discrete actions in the framework of Abbas.

We are not aware of any study devoted to soundness (or robustness) of test suites for the above-mentioned approximate conformance relations. In a slightly different context, Fainekos and Pappas [17] studied the robust sampling of a continuous trajectory (called signal in that context). For this purpose, a notion of *robustness estimate* is defined for a given signal in [17]. In summary, for a given Metric Interval Temporal Logic (MITL) formula and a given point in the signal, the robustness estimate specifies how deeply the formula is satisfied by that point. For example, consider the simple formula $t < 10$; then, the robustness of a signal point with value 7 is 3. This is done using the notions of *depth* and *distance*, already defined as the distance between a point and the boundary of a given set. Our soundness criteria bear close similarly to their notion of robustness for MITL. We not only require bounds on the depth of conformance at the sampling points, but also require them in their vicinity (by assuming an upper bound on the signal variation).

Different notions of robustness have been proposed within the control theory, e.g., the notion of input to state stability [18]; these notions have been recently considered in the context of cyber-physical systems [19]. We would like to study these notions of robustness and investigate their possible application to our setting for generating sound test cases.

VI. Conclusions

In this paper, we have defined a straightforward notion of conformance testing for cyber-physical systems. This notion is supposed to capture the conformance relation proposed by Abbas and Fainekos. To this end, we have formulated a soundness requirement on test suites and we have shown that test suites are not generally sound. To remedy this, we defined some criteria on the test suite (relative to the sampling points and the system dynamics around those sampling points) and have proven that they indeed guarantee soundness. We have also explored some preliminary ideas regarding exhaustiveness.

As a first immediate step, we would like to provide a formal proof for our conjecture regarding exhaustiveness. Subsequently, we would like to find practical ways of checking our criteria (particularly regarding soundness) and implement them in our prototype tool [20]. To this end, we may restrict ourselves to a particular class of system dynamics and reduce the criteria to syntactic checks (on the sampling rate and the derivatives).

Acknowledgment

The comments of the anonymous reviewers of TASE 2016 are gratefully acknowledged.

The work of M. R. Mousavi has been partially supported by the Swedish Research Council (Vetenskapsradet) award number: 621-2014-5057 (Effective Model-Based Testing of Concurrent Systems) and the Swedish Knowledge Foundation (Stiftelsen for Kunskaps- och Kompetensutveckling) in the context of the AUTO-CAAS HoG project (number: 20140312).

References

[1] M. Broy, B. Jonsson, J.-P. Katoen, M. Leucker, and A. Pretschner, *Model-Based Testing of Reactive Systems*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2005, vol. 3472.

[2] H. Abbas, B. Hoxha, G. E. Fainekos, J. V. Deshmukh, J. Kapinski, and K. Ueda, "WiP abstract: Conformance testing as falsification for cyber-physical systems," in *Proceedings of the ACM/IEEE 5th International Conference on Cyber-Physical Systems (ICCPS 2014)*. IEEE CS, 2014, p. 211, available online: http://arxiv.org/abs/1401.5200.

[3] H. Abbas, H. Mittelmann, and G. E. Fainekos, "Formal property verification in a conformance testing framework," in *12th ACM-IEEE International Conference on Formal Methods and Models for System Design (MEMOCODE 2014)*. IEEE, 2014, pp. 155–164.

[4] T. Dang, "Model-based testing of hybrid systems," *Monograph in Model-Based Testing for Embedded Systems, CRC Press*, 2010.

[5] N. Khakpour and M. R. Mousavi, "Notions of conformance testing for cyber-physical systems: Overview and roadmap (invited paper)," in *Proc. of the 26th International Conference on Concurrency Theory, CONCUR 2015*, ser. LIPIcs, vol. 42. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015, pp. 18–40.

[6] M. van Osch, "Automated model-based testing of hybrid systems," Ph.D. dissertation, Eindhoven University of Technology, The Netherlands, 2009.

[7] J. Tretmans, "Model based testing with labelled transition systems," in *Formal Methods and Testing, An Outcome of the FORTEST Network, Revised Selected Papers*, ser. Lecture Notes in Computer Science, vol. 4949. Springer, 2008, pp. 1–38.

[8] M. Yannakakis and D. Lee, "Testing of finite state systems," in *Computer Science Logic*, ser. Lecture Notes in Computer Science, vol. 1584. Springer Berlin Heidelberg, 1999, pp. 29–44.

[9] H. Y. Abbas, "Test-based falsification and conformance testing for cyber-physical systems," Ph.D. dissertation, Arizona State University, 2015. [Online]. Available: http://hdl.handle.net/2286/R.A.150686

[10] R. Goebel, R. Sanfelice, and A. Teel, "Hybrid dynamical systems," *IEEE Control Systems*, vol. 29, no. 2, Month 2009, note. [Online]. Available: http://arxiv.org/abs/1009.3306

[11] K. J. Aström and B. Wittenmark, *Computer-controlled systems*. Prentice Hall Englewood Cliffs, NJ, 1997.

[12] M. Mohaqeqi, M. R. Mousavi, and W. Taha, "Conformance testing of cyber-physical systems: A comparative study," in *Post-Proceedings of the 14th International Workshop on Automated Verification of Critical Systems (AVOCS 2014)*, ser. ECEASST, vol. 70, 2014. [Online]. Available: http://journal.ub.tu-berlin.de/eceasst/article/view/982

[13] J. Tretmans, "Test generation with inputs, outputs and repetitive quiescence," *Software - Concepts and Tools*, vol. 17, no. 3, pp. 103–120, 1996.

[14] M. van Osch, "Hybrid input-output conformance and test generation," in *Formal Approaches to Software Testing and Runtime Verification*, ser. Lecture Notes in Computer Science. Springer, 2006, vol. 4262, pp. 70–84.

[15] P. J. L. Cuijpers, M. A. Reniers, and W. P. M. H. Heemels, *Hybrid transition systems*. Computer Science Reports 02-12, Technische Universiteit Eindhoven, Department of Mathematics and Computer Science, 2002.

[16] M. Mohaqeqi and M. R. Mousavi, "Approximate conformance for hybrid I/O automata," in *Proceedings of the First International Workshop on Verification and Validation of Cyber-Physical Systems (VVCPS 2016)*, ser. Electronic Proceedings in Theoretical Computer Science, 2016.

[17] G. Fainekos and G. Pappas, "Robust sampling for MITL specifications," in *Formal Modeling and Analysis of Timed Systems*, ser. Lecture Notes in Computer Science, J.-F. Raskin and P. Thiagarajan, Eds., vol. 4763. Springer Berlin Heidelberg, 2007, pp. 147–162.

[18] E. D. Sontag, "Input to state stability: Basic concepts and results," in *Nonlinear and optimal control theory*. Springer, 2008, pp. 163–220.

[19] M. Rungger and P. Tabuada, "A notion of robustness for cyber-physical systems," *IEEE Transactions on Automatic Control*, 2016, to appear.

[20] A. Aerts, M. R. Mousavi, and M. Reniers, "A tool prototype for model-based testing of cyber-physical systems," in *Proceedings of the 12th International Colloquium on Theoretical Aspects of Computing (ICTAC 2015)*, ser. Lecture Notes in Computer Science, M. Leucker, C. Rueda, and F. D. Valencia, Eds., vol. 9399. Springer International Publishing, 2015, pp. 563–572.