Querying Parametric Temporal Logic Properties on Embedded Systems

Hengyi Yang, Bardh Hoxha, and Georgios Fainekos

School of Computing, Informatics and Decision Systems Engineering,
Arizona State University
{hyang67,bhoxha,fainekos}@asu.edu

Abstract. In Model Based Development (MBD) of embedded systems, it is often desirable to not only verify/falsify certain formal system specifications, but also to automatically explore the properties that the system satisfies. Namely, given a parametric specification, we would like to automatically infer the ranges of parameters for which the property holds/does not hold on the system. In this paper, we consider parametric specifications in Metric Temporal Logic (MTL). Using robust semantics for MTL, the parameter estimation problem can be converted into an optimization problem which can be solved by utilizing stochastic optimization methods. The framework is demonstrated on some examples from the literature.

1 Introduction

Software development for embedded control systems is particularly challenging. The software may be distributed with real time constraints and must interact with the physical environment in non trivial ways. Multiple incidents and accidents of safety critical systems [1,2] reinforce the need for design, verification and validation methodologies that provide a certain level of confidence in the system correctness and robustness.

Recently, there has been a trend to develop software for safety critical embedded control systems using the Model Based Design (MBD) paradigm. Among the benefits of the MBD approach is that it provides the possibility for automatic code generation. Based on a level of confidence on the automatic code generation process, some of the system verification and validation can be performed at earlier design stages using only models of the system. Due to the importance of the problem, there has been a substantial level of research on testing and verification of models of embedded and hybrid systems (see [3] for an overview).

In [4], we investigated a new approach for testing embedded and hybrid systems against formal requirements in Metric Temporal Logic (MTL) [5]. Our work was premised on the need to express complex design requirements in a formal logic for both requirements analysis and requirements verification. Based on the concept of robustness of MTL specifications [6], we were able to pose the property falsification/testing problem as an optimization problem. In particular, robust MTL semantics provide the user with an application depended

B. Nielsen and C. Weise (Eds.): ICTSS 2012, LNCS 7641, pp. 136–151, 2012.
© IFIP International Federation for Information Processing 2012

measure of how far a system behavior is from failing to satisfy a requirement. Therefore, the goal of an automatic test generator is to produce a sequence of tests by gradually reducing that positive measure until a system behavior with a negative robustness measure is produced. In other words, we are seeking to detect system behaviors that minimize the specification robustness measure.

Unfortunately, the resulting optimization problem is non-linear and non-convex, in general. Moreover, embedded system models frequently contain black boxes as subcomponents. Thus, only stochastic optimization techniques can be employed for solving the optimization problem and, in turn, for solving the initial falsification problem. In our previous research [7,8,4], we have explored the applicability of various stochastic optimization methods to the MTL falsification problem with great success.

In this work, we take the MTL falsification method one step further. Namely, not only would we like to detect a falsifying behavior if one exists, but also we would like to be able to explore and determine system properties. Such a property exploration framework can be of great help to the practitioner. In many cases, the system requirements are not well formalized or understood at the initial system design stages. Therefore, if the specification can be falsified, then it is natural to ask for what parameter values the system still falsifies the specification.

In more detail, given an MTL specification with an unknown or uncertain parameter [9], we automatically formulate an optimization problem whose solution provides a range of values for the parameter such that the specification does not hold on the system. In order to solve the resulting optimization problem, we utilize our MTL falsification toolbox S-TaLiRo [10], which contains a number of stochastic optimization methods [7,8,4]. Finally, we demonstrate our framework on a challenge problem from the industry [11] and we present some experimental results on a small number of benchmark problems.

2 Problem Formulation

In this work, we take a general approach in modeling real-time embedded systems that interact with physical systems that have non-trivial dynamics. In the following, we will be using the term *hybrid systems* or *Cyber-Physical Systems* (CPS) for such systems to stress the interconnection between the embedded system and the physical world.

We fix $N \subseteq \mathbb{N}$, where $\mathbb{N}$ is the set of natural numbers, to be a finite set of indexes for the finite representation of a system behavior. In the following, given two sets A and B, B^A denotes the set of all functions from A to B. That is, for any $f \in B^A$ we have $f : A \to B$.

We view a system Σ as a mapping from a compact set of *initial operating conditions* X_0 and *input signals* $\mathbf{U} \subseteq U^N$ to *output signals* Y^N and *timing* (or *sampling*) functions $\mathfrak{T} \subseteq \mathbb{R}_+^N$. Here, U is a compact set of possible input values at each point in time (input space), Y is the set of output values (output space), $\mathbb{R}$ is the set of real numbers and $\mathbb{R}_+$ the set of positive reals.

We impose three assumptions / restrictions on the systems that we consider:

1. The input signals (if any) must be parameterizable using a finite number of parameters. That is, there exists a function $\mathfrak{U}$ such that for any $u \in \mathbf{U}$, there exist two parameter vectors $\lambda = [\lambda_1 \ \dots \ \lambda_m]^T \in \Lambda$, where Λ is a compact set, and $t = [t_1 \ \dots \ t_m]^T \in \mathbb{R}_+^m$ such that $m << \max N$ and for all $i \in N$, $u(i) = \mathfrak{U}(\lambda, t)(i)$.
2. The output space Y must be equipped with a generalized metric $\mathbf{d}$ which contains a subspace Z equipped with a metric d.
3. For a specific initial condition x_0 and input signal u, there must exist a unique output signal $\mathbf{y}$ defined over the time domain R. That is, the system Σ is deterministic.

Further details on the necessity and implications of the aforementioned assumptions can be found in [12].

Under Assumption 3, a system Σ can be viewed as a function $\Delta_\Sigma : X_0 \times \mathbf{U} \to Y^N \times \mathfrak{T}$ which takes as an input an initial condition $x_0 \in X_0$ and an input signal $u \in \mathbf{U}$ and it produces as output a signal $\mathbf{y} : N \to Y$ (also referred to as *trajectory*) and a timing function $\tau : N \to \mathbb{R}_+$. The only restriction on the timing function τ is that it must be a monotonic function, i.e., $\tau(i) < \tau(j)$ for $i < j$. The pair $\mu = (\mathbf{y}, \tau)$ is usually referred to as a *timed state sequence*, which is a widely accepted model for reasoning about real time systems [13]. A timed state sequence can represent a computer simulated trajectory of a CPS or the sampling process that takes place when we digitally monitor physical systems. We remark that a timed state sequence can represent both the internal state of the software/hardware (usually through an abstraction) and the state of the physical system. The set of all timed state sequences of a system Σ will be denoted by $\mathcal{L}(\Sigma)$. That is,

$$\mathcal{L}(\Sigma) = \{(\mathbf{y}, \tau) \mid \exists x_0 \in X_0 \,.\, \exists u \in \mathbf{U} \,.\, (\mathbf{y}, \tau) = \Delta_\Sigma(x_0, u)\}.$$

Our high level goal is to explore and infer properties that the system Σ satisfies by observing its response (output signals) to particular input signals and initial conditions. We assume that the system designer has some partial understanding about the properties that the system satisfies or does not satisfy and he/she would like to be able to precisely determine these properties. In particular, we assume that the system developer can formalize the system properties in Metric Temporal Logic (MTL) [5], but some parameters are unknown. Such parameters could be unknown threshold values for the continuous state variables of the hybrid system or some unknown real time constraints.

Example 1. *As a motivating example, we will consider a slightly modified version of the Automatic Transmission model provided by Mathworks as a Simulink demo*[1]*. Further details on this example can be found in [14,15,12].*

The only input u to the system is the throttle schedule, while the break schedule is set simply to 0 for the duration of the simulation which is $T = 30$ sec.

[1] Available at: `http://www.mathworks.com/products/simulink/demos.html`

The physical system has two continuous-time state variables which are also its outputs: the speed of the engine ω (RPM) and the speed of the vehicle v, i.e., $Y = \mathbb{R}^2$ and $\mathbf{y}(t) = [\omega(t)\ v(t)]^T$ for all $t \in [0, 30]$. Initially, the vehicle is at rest at time 0, i.e., $X_0 = \{[0\ 0]^T\}$ and $x_0 = \mathbf{y}(0) = [0\ 0]^T$. Therefore, the output trajectories depend only on the input signal u which models the throttle, i.e., $(\mathbf{y}, \tau) = \Delta_\Sigma(u)$. The throttle at each point in time can take any value between 0 (fully closed) to 100 (fully open). Namely, $u(i) \in U = [0, 100]$ for each $i \in N$. The model also contains a Stateflow chart with two concurrently executing Finite State Machines (FSMs) with 4 and 3 states, respectively. The FSMs model the logic that controls the switching between the gears in the transmission system. We remark that the system is deterministic, i.e., under the same input u, we will always observe the same output $\mathbf{y}$.

In our previous work [12,10,7], on such models, we demonstrated how to falsify requirements like: "The vehicle speed v is always under 120km/h or the engine speed ω is always below 4500RPM." A falsifying system trajectory appears in Fig. 1. In this work, we provide answers to queries like "What is the fastest time that ω can exceed 3250 RPM" or "For how long can ω be below 4500 RPM".

Formally, in this work, we solve the following problem.

Problem 1 (Temporal Logic Parameter Estimation Problem). *Given an MTL formula $\phi[\theta]$ with a single unknown parameter $\theta \in \Theta = [\theta_m, \theta_M] \subseteq \mathbb{R}$, a hybrid system Σ, and a maximum testing time T, find an optimal range $\Theta^* = [\theta^*_m, \theta^*_M]$ such that for any $\zeta \in \Theta^*$, $\phi[\zeta]$ does not hold on Σ, i.e., $\Sigma \not\models \phi[\zeta]$.*

Ideally, by solving Problem 1, we would also like to have the property that for any $\zeta \in \Theta - \Theta^*$, $\phi[\zeta]$ holds on Σ, i.e., $\Sigma \models \phi[\zeta]$. However, even for a given ζ, the problem of algorithmically computing whether $\Sigma \models \phi[\zeta]$ is not easy to solve for the classes of hybrid systems that we consider in this work.

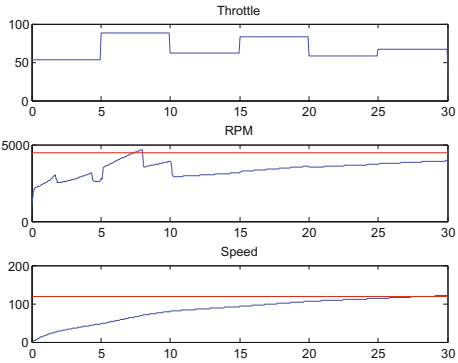

Fig. 1. Example 1: A piecewise constant input signal u parameterized with $\Lambda \in [0, 100]^6$ and $t = [0, 5, 10, 15, 20, 25]$ and the corresponding output signals that falsify the specification

An overview of our proposed solution to Problem 1 appears in Fig. 2. The sampler produces a point x_0 from the set of initial conditions, a parameter vector λ that characterizes the control input signal u and a parameter θ. The vectors x_0 and λ are passed to the system simulator which returns an execution trace (output trajectory and timing function). The trace is then analyzed by the MTL robustness analyzer which returns a robustness value representing the best estimate for the robustness found so far. In turn, the robustness score computed is used by the stochastic sampler to decide on a next input to analyze.

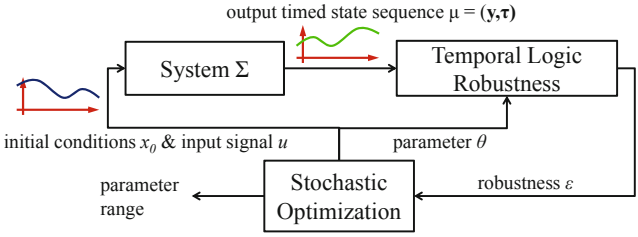

Fig. 2. Overview of the solution to the MTL parameter estimation problem on CPS

The process terminates after a maximum number of tests or when no improvement on the parameter estimate θ has been made after a number of tests.

3 Robustness of Metric Temporal Logic Formulas

Metric Temporal Logic (MTL) was introduced in [5] in order to reason about the quantitative timing properties of boolean signals. In the following, we present directly MTL in Negation Normal Form (NNF) since this is needed for the presentation of the new results in Section 5. We denote the extended real number line by $\overline{\mathbb{R}} = \mathbb{R} \cup \{\pm\infty\}$.

Definition 1 (Syntax of MTL in NNF). *Let $\overline{\mathbb{R}}$ be the set of truth degree constants, AP be the set of atomic propositions and $\mathcal{I}$ be a non-empty non-singular interval of $\overline{\mathbb{R}}_{\geq 0}$. The set MTL of all well-formed formulas (wff) is inductively defined using the following rules:*

- *Terms: True ($\top$), false ($\bot$), all constants $r \in \overline{\mathbb{R}}$ and propositions p, $\neg p$ for $p \in AP$ are terms.*
- *Formulas: if ϕ_1 and ϕ_2 are terms or formulas, then $\phi_1 \vee \phi_2$, $\phi_1 \wedge \phi_2$, $\phi_1 \mathcal{U}_{\mathcal{I}} \phi_2$ and $\phi_1 \mathcal{R}_{\mathcal{I}} \phi_2$ are formulas.*

The atomic propositions in our case label subsets of the output space Y. In other words, each atomic proposition is a shorthand for an arithmetic expression of the form $p \equiv g(y) \leq c$, where $g : Y \to \mathbb{R}$ and $c \in \mathbb{R}$. We define an observation map $\mathcal{O} : AP \to \mathcal{P}(Y)$ such that for each $p \in AP$ the corresponding set is $\mathcal{O}(p) = \{y \mid g(y) \leq c\} \subseteq Y$.

In the above definition, $\mathcal{U}_{\mathcal{I}}$ is the timed *until* operator and $\mathcal{R}_{\mathcal{I}}$ the timed *release* operator. The subscript $\mathcal{I}$ imposes timing constraints on the temporal operators. The interval $\mathcal{I}$ can be open, half-open or closed, bounded or unbounded, but it must be non-empty ($\mathcal{I} \neq \emptyset$) (and, practically speaking, non-singular ($\mathcal{I} \neq \{t\}$)). In the case where $\mathcal{I} = [0, +\infty)$, we remove the subscript $\mathcal{I}$ from the temporal operators, i.e., we just write $\mathcal{U}$, and $\mathcal{R}$. Also, we can define *eventually* ($\Diamond_{\mathcal{I}} \phi \equiv \top \mathcal{U}_{\mathcal{I}} \phi$) and *always* ($\Box_{\mathcal{I}} \phi \equiv \bot \mathcal{R}_{\mathcal{I}} \phi$).

Before proceeding to the actual definition of the robust semantics, we introduce some auxiliary notation. A metric space is a pair (X, d) such that the

topology of the set X is induced by a metric d. Using a metric d, we can define the distance of a point $x \in X$ from a set $S \subseteq X$. Intuitively, this distance is the shortest distance from x to all the points in S. In a similar way, the depth of a point x in a set S is defined to be the shortest distance of x from the boundary of S. Both the notions of distance and depth will play a fundamental role in the definition of the robustness degree.

Definition 2 (Signed Distance). *Let $x \in X$ be a point, $S \subseteq X$ be a set and d be a metric on X. Then, we define the Signed Distance from x to S to be*

$$\mathbf{Dist}_d(x,S) := \begin{cases} -\mathbf{dist}_d(x,S) := -\inf\{d(x,y) \mid y \in S\} & \text{if } x \notin S \\ \mathbf{depth}_d(x,S) := \mathbf{dist}_d(x, X \backslash S) & \text{if } x \in S \end{cases}$$

We remark that we use the extended definition of the supremum and infimum, i.e., $\sup \emptyset := -\infty$ and $\inf \emptyset := +\infty$.

MTL formulas are interpreted over timed state sequences μ. In the past [6], we proposed multi-valued semantics for MTL where the valuation function on the predicates takes values over the totally ordered set $\overline{\mathbb{R}}$ according to a metric d operating on the output space Y. For this purpose, we let the valuation function be the depth (or the distance) of the current point of the signal $\mathbf{y}(i)$ in a set $\mathcal{O}(p)$ labeled by the atomic proposition p. Intuitively, this distance represents how robustly is the point $\mathbf{y}(i)$ within a set $\mathcal{O}(p)$. If this metric is zero, then even the smallest perturbation of the point can drive it inside or outside the set $\mathcal{O}(p)$, dramatically affecting membership.

For the purposes of the following discussion, we use the notation $[\![\phi]\!]$ to denote the robustness estimate with which the timed state sequence μ satisfies the specification ϕ. Formally, the valuation function for a given formula ϕ is $[\![\phi]\!] : (Y^N \times \mathfrak{T}) \times N \to \overline{\mathbb{R}}$. In the definition below, we also use the following notation : for $Q \subseteq R$, the *preimage* of Q under τ is defined as : $\tau^{-1}(Q) := \{i \in N \mid \tau(i) \in Q\}$.

Definition 3 (Robustness Estimate). *Let $\mu = (\mathbf{y}, \tau) \in \mathcal{L}(\Sigma)$, $r \in \overline{\mathbb{R}}$ and $i, j, k \in N$, then the robustness estimate of any formula MTL ϕ with respect to μ is recursively defined as follows*

$$[\![r]\!](\mu,i) := r \qquad [\![\top]\!](\mu,i) := +\infty \qquad [\![\bot]\!](\mu,i) := -\infty$$
$$[\![p]\!](\mu,i) := \mathbf{Dist}_d(\mathbf{y}(i), \mathcal{O}(p)) \qquad [\![\neg p]\!](\mu,i) := -\mathbf{Dist}_d(\mathbf{y}(i), \mathcal{O}(p))$$
$$[\![\phi_1 \vee \phi_2]\!](\mu,i) := \max([\![\phi_1]\!](\mu,i), [\![\phi_2]\!](\mu,i))$$
$$[\![\phi_1 \wedge \phi_2]\!](\mu,i) := \min([\![\phi_1]\!](\mu,i), [\![\phi_2]\!](\mu,i))$$
$$[\![\phi_1 \,\mathcal{U}_{\mathcal{I}} \phi_2]\!](\mu,i) := \sup_{j \in \tau^{-1}(\tau(i)+\mathcal{I})} \big(\min([\![\phi_2]\!](\mu,j), \inf_{i \le k < j} [\![\phi_1]\!](\mu,k))\big)$$
$$[\![\phi_1 \mathcal{R}_{\mathcal{I}} \phi_2]\!](\mu,i) := \inf_{j \in \tau^{-1}(\tau(i)+\mathcal{I})} \big(\max([\![\phi_2]\!](\mu,j), \sup_{i \le k < j} [\![\phi_1]\!](\mu,k))\big)$$

Recall that we use the extended definition of supremum and infimum. When $i = 0$, then we simply write $[\![\phi]\!](\mu)$.

The robustness of an MTL formula with respect to a timed state sequence can be computed using several existing algorithms [6,15,16].

4 Parametric Metric Temporal Logic over Signals

In many cases, it is important to be able to describe an MTL specification with unknown parameters and, then, infer the parameters that make the specification true/false. In [9], Asarin et. al. introduce Parametric Signal Temporal Logic (PSTL) and present two algorithms for computing approximations for parameters over a given signal. Here, we review some of the results in [9] while adapting them in the notation and formalism that we use in this paper.

We will restrict the occurrences of unknown parameters in the specification to a single parameter that may appear either in the timing constraints of a temporal operator or in the atomic propositions.

Definition 4 (Syntax of Parametric MTL (PMTL)). *Let λ be a parameter, then the set of all well formed PMTL formulas is the set of all well formed MTL formulas where either λ appears in an arithmetic expression, i.e., $p[\lambda] \equiv g(y) \leq \lambda$, or in the timing constraint of a temporal operator, i.e., $\mathcal{I}[\lambda]$.*

We will denote a PMTL formula ϕ with parameter λ by $\phi[\lambda]$. Given some value $\theta \in \Theta$, then the formula $\phi[\theta]$ is an MTL formula.

Since the valuation function of an MTL formula is a composition of minimum and maximum operations quantified over time intervals, a formula $\phi[\lambda]$ is monotonic with respect to λ.

Example 2. *Consider the PMTL formula $\phi[\lambda] = \Box_{[0,\lambda]} p$ where $p \equiv (\omega \leq 3250)$. Given a timed state sequence $\mu = (\mathbf{y}, \tau)$ with $\tau(0) = 0$, for $\theta_1 \leq \theta_2$, we have: $[0, \theta_1] \subseteq [0, \theta_2] \implies \tau^{-1}([0, \theta_1]) \subseteq \tau^{-1}([0, \theta_2])$. Therefore, $[\![\phi[\theta_1]]\!](\mu) = \inf_{i \in \tau^{-1}([0,\theta_1])}(-\mathbf{Dist}_d(\mathbf{y}(i), \mathcal{O}(p))) \geq \inf_{i \in \tau^{-1}([0,\theta_2])}(-\mathbf{Dist}_d(\mathbf{y}(i), \mathcal{O}(p))) = [\![\phi[\theta_2]]\!](\mu)$. That is, the function $[\![\phi[\theta]]\!](\mu)$ is non-increasing with θ. See Fig. 3 for an example using an output trajectory from the system in Example 1.*

The previous example can be formalized in the following result.

Proposition 1. *Consider a PMTL formula $\phi[\lambda]$ such that it contains a subformula $\phi_1 Op_{\mathcal{I}[\lambda]} \phi_2$ where $Op \in \{\mathcal{U}, \mathcal{R}\}$. Then, given a timed state sequence $\mu = (\mathbf{y}, \tau)$, for θ_1, $\theta_2 \in \overline{\mathbb{R}}_{\geq 0}$, such that $\theta_1 \leq \theta_2$, and for $i \in N$, we have:*

1. *if (i) $Op = \mathcal{U}$ and $\sup \mathcal{I}[\lambda] = \lambda$ or (ii) $Op = \mathcal{R}$ and $\inf \mathcal{I}[\lambda] = \lambda$, then $[\![\phi[\theta_1]]\!](\mu, i) \leq [\![\phi[\theta_2]]\!](\mu, i)$, i.e., the function $[\![\phi[\lambda]]\!](\mu, i)$ is nondecreasing with respect to λ, and*
2. *if (i) $Op = \mathcal{R}$ and $\sup \mathcal{I}[\lambda] = \lambda$ or (ii) $Op = \mathcal{U}$ and $\inf \mathcal{I}[\lambda] = \lambda$, then $[\![\phi[\theta_1]]\!](\mu, i) \geq [\![\phi[\theta_2]]\!](\mu, i)$, i.e., the function $[\![\phi[\lambda]]\!](\mu, i)$ is non-increasing with respect to λ.*

Proof (Sketch). The proof is by induction on the structure of the formula and it is similar to the proofs that appear in [6].

For completeness, we present the case $[\![\phi_1 \mathcal{U}_{\langle \alpha, \lambda \rangle} \phi_2]\!](\mu, i)$, where $\langle \in \{[, (\}$ and $\rangle \in \{],)\}$. The other cases are either similar or they are based on the monotonicity

of the operators max and min. Let $\theta_1 \leq \theta_2$, then:

$$[\![\phi_1 \mathcal{U}_{\langle \alpha,\theta_1 \rangle} \phi_2]\!](\mu, i) \leq \max\left([\![\phi_1 \mathcal{U}_{\langle \alpha,\theta_1 \rangle} \phi_2]\!](\mu, i), [\![\phi_1 \mathcal{U}_{\overline{\langle}\theta_1,\theta_2\rangle} \phi_2]\!](\mu, i)\right)$$
$$= [\![\phi_1 \mathcal{U}_{\langle \alpha,\theta_2 \rangle} \phi_2]\!](\mu, i)$$

where $\overline{\langle} \in \{[, (\}$ such that $\langle \alpha, \theta_1\rangle \cap \overline{\langle}\theta_1, \theta_2\rangle = \emptyset$ and $\langle \alpha, \theta_1\rangle \cup \overline{\langle}\theta_1, \theta_2\rangle = \langle \alpha, \theta_2\rangle$. □

We can derive similar results when the parameter appears in the numerical expression of the atomic proposition.

Proposition 2. *Consider a PMTL formula $\phi[\lambda]$ such that it contains a parametric atomic proposition $p[\lambda]$ in a subformula. Then, given a timed state sequence $\mu = (\mathbf{y}, \tau)$, for θ_1, $\theta_2 \in \overline{\mathbb{R}}_{\geq 0}$, such that $\theta_1 \leq \theta_2$, and for $i \in N$, we have:*

1. *if $p[\lambda] \equiv g(x) \leq \lambda$, then $[\![\phi[\theta_1]]\!](\mu, i) \leq [\![\phi[\theta_2]]\!](\mu, i)$, i.e., the function $[\![\phi[\lambda]]\!](\mu, i)$ is nondecreasing with respect to λ, and*
2. *if $p[\lambda] \equiv g(x) \geq \lambda$, then $[\![\phi[\theta_1]]\!](\mu, i) \geq [\![\phi[\theta_2]]\!](\mu, i)$, i.e., the function $[\![\phi[\lambda]]\!](\mu, i)$ is non-increasing with respect to λ.*

Proof (Sketch). The proof is by induction on the structure of the formula and it is similar to the proofs that appear in [6].

For completeness, we present the base case $[\![p[\lambda]]\!](\mu, i)$ where $p[\lambda] \equiv g(x) \leq \lambda$. Since $\theta_1 \leq \theta_2$, $\mathcal{O}(p[\theta_1]) \subseteq \mathcal{O}(p[\theta_2])$. We will only present the case for which $\mathbf{y}(i) \notin \mathcal{O}(p[\theta_2])$. We have:

$$\mathcal{O}(p[\theta_1]) \subseteq \mathcal{O}(p[\theta_2]) \implies \mathbf{dist}_d(\mathbf{y}(i), \mathcal{O}(p[\theta_1])) \geq \mathbf{dist}_d(\mathbf{y}(i), \mathcal{O}(p[\theta_2])) \implies$$
$$\mathbf{Dist}_d(\mathbf{y}(i), \mathcal{O}(p[\theta_1])) \leq \mathbf{Dist}_d(\mathbf{y}(i), \mathcal{O}(p[\theta_2])) \implies [\![p[\theta_1]]\!](\mu, i) \leq [\![p[\theta_2]]\!](\mu, i) \square$$

The results presented in this section can be easily extended to multiple parameters. However, in this work, we will focus on a single parameter in order to derive a more tractable optimization problem.

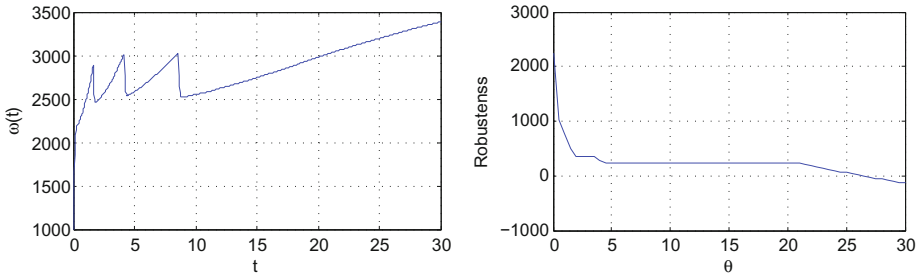

Fig. 3. Example 2. Left: Engine speed $\omega(t)$ for constant throttle $u(t) = 50$. Right: The robustness of the specification $\square_{[0,\theta]}(\omega \leq 3250)$ with respect to θ.

5 Temporal Logic Parameter Bound Computation

The notion of robustness of temporal logics will enable us to pose the parameter estimation problem as an optimization problem. In order to solve the resulting optimization problem, falsification methods and S-TaLiRo can be utilized in order to estimate Θ^* for Problem 1.

As described in the previous section, the parametric robustness functions that we are considering are monotonic with respect to the search parameter. Therefore, if we are searching for a parameter over an interval $\Theta = [\theta_m, \theta_M]$, we know that Θ^* is going to be either of the form $[\theta_m, \theta^*]$ or $[\theta^*, \theta_M]$. In other words, depending on the structure of $\phi[\lambda]$, we are either trying to minimize or maximize θ^* such that for all $\theta \in \Theta^*$, we have $[\![\phi[\theta]]\!](\Sigma) = \min_{\mu \in \mathcal{L}_\tau(\Sigma)} [\![\phi[\theta]]\!](\mu) \leq 0$.

Example 3. *Let us consider again the automotive transmission example and the specification* $\phi[\lambda] = \Box_{[0,\lambda]} p$ *where* $p \equiv (\omega \leq 4500)$. *The specification robustness* $[\![\phi[\theta]]\!](\Delta_\Sigma(u))$ *as a function of* θ *and the input* u *appears in Fig. 4 (left) for constant input signals. The creation of the graph required* $100 \times 30 = 3,000$ *tests. The contour under the surface indicates the zero level set of the robustness surface, i.e., the* θ *and* u *values for which we get* $[\![\phi[\theta]]\!](\Delta_\Sigma(u)) = 0$. *From the graph, we can infer that* $\theta^* \approx 2.8$ *and that for any* $\theta \in [2.8, 30]$, *we have* $[\![\phi[\theta]]\!](\Sigma) \leq 0$. *The approximate value of* θ^* *is a rough estimate based on the granularity of the grid that we used to plot the surface.*

In summary, in order to solve Problem 1, we would have to solve the following optimization problem:

$$\begin{aligned} &\text{optimize} && \theta \\ &\text{subject to} && \theta \in \Theta \text{ and } [\![\phi[\theta]]\!](\Sigma) = \min_{\mu \in \mathcal{L}_\tau(\Sigma)} [\![\phi[\theta]]\!](\mu) \leq 0 \end{aligned} \tag{1}$$

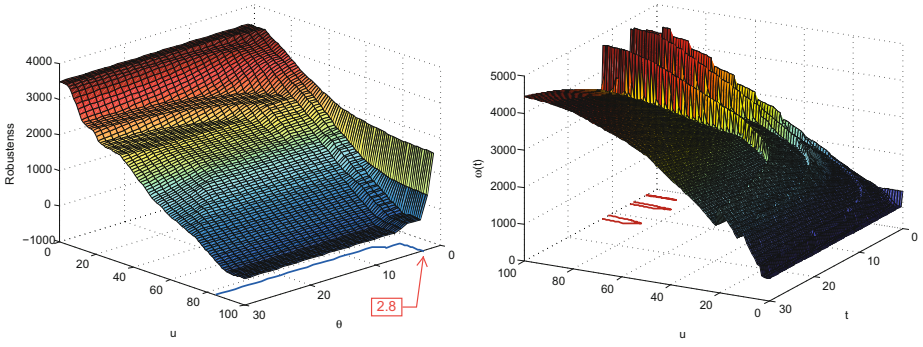

Fig. 4. Example 3: Left: Specification robustness as a function of the parameter θ and the constant input u. Right: Engine speed $\omega(t)$ as a function of the constant input u and time t. The contours indicate the u-t combinations for which $\omega(t) = 4500$.

However, $[\![\phi[\theta]]\!](\Sigma)$ neither can be computed using reachability analysis algorithms nor is known in closed form for the systems that we are considering. Therefore, we will have to compute an under-approximation of Θ^*.

Our focus will be to formulate an optimization problem that can be solved using stochastic search methods. In particular, we will reformulate optimization problem (1) into a new one where the constraints due to the specification are incorporated into the cost function:

$$\text{optimize}_{\theta\in\Theta}\left(\theta + \begin{cases} \gamma \pm [\![\phi[\theta]]\!](\Sigma) & \text{if } [\![\phi[\theta]]\!](\Sigma) \geq 0 \\ 0 & \text{otherwise} \end{cases}\right) \tag{2}$$

where the sign ($\pm$) and the parameter γ depend on whether the problem is a maximization or a minimization problem. The parameter γ must be properly chosen so that the optimum of problem (2) is in Θ if and only if $[\![\phi[\theta]]\!](\Sigma) \leq 0$. In other words, we must avoid the case where for some θ, we have $[\![\phi[\theta]]\!](\Sigma) > 0$ and $(\theta + [\![\phi[\theta]]\!](\Sigma)) \in \Theta$. Therefore, if the problem in Eq. (1) is feasible, then the optimum of equations (1) and (2) is the same.

5.1 Non-increasing Robustness Functions

First, we consider the case of non-increasing robustness functions $[\![\phi[\theta]]\!](\Sigma)$ with respect to the search variable θ. In this case, the optimization problem is a minimization problem.

To see why this is the case, assume that $[\![\phi[\theta_M]]\!](\Sigma) \leq 0$. Since for $\theta \leq \theta_M$, we have $[\![\phi[\theta]]\!](\Sigma) \geq [\![\phi[\theta_M]]\!](\Sigma)$, we need to find the minimum θ such that we still have $[\![\phi[\theta]]\!](\Sigma) \leq 0$. That θ will be θ^* since for all $\theta' \in [\theta^*, \theta_M]$, we will have $[\![\phi[\theta']]\!](\Sigma) \leq 0$.

We will reformulate the problem of Eq. (2) so that we do not have to solve two separate optimization problems. From (2), we have:

$$\begin{aligned} &\min_{\theta\in\Theta}\left(\theta + \begin{cases} \gamma + \min_{\mu\in\mathcal{L}_\tau(\Sigma)}[\![\phi[\theta]]\!](\mu) & \text{if } \min_{\mu\in\mathcal{L}_\tau(\Sigma)}[\![\phi[\theta]]\!](\mu) \geq 0 \\ 0 & \text{otherwise} \end{cases}\right) = \\ &\quad = \min_{\theta\in\Theta}\left(\theta + \min_{\mu\in\mathcal{L}_\tau(\Sigma)}\begin{cases} \gamma + [\![\phi[\theta]]\!](\mu) & \text{if } [\![\phi[\theta]]\!](\mu) \geq 0 \\ 0 & \text{otherwise} \end{cases}\right) = \\ &\quad = \min_{\theta\in\Theta}\min_{\mu\in\mathcal{L}_\tau(\Sigma)}\left(\theta + \begin{cases} \gamma + [\![\phi[\theta]]\!](\mu) & \text{if } [\![\phi[\theta]]\!](\mu) \geq 0 \\ 0 & \text{otherwise} \end{cases}\right) \end{aligned} \tag{3}$$

where $\gamma \geq \max(\theta_M, 0)$.

The previous discussion is formalized in the following result.

Proposition 3. *Let θ^* and μ^* be the parameters returned by an optimization algorithm that is applied to the problem in Eq. (3). If $[\![\phi[\theta^*]]\!](\mu^*) \leq 0$, then for all $\theta \in \Theta^* = [\theta^*, \theta_M]$, we have $[\![\phi[\theta]]\!](\Sigma) \leq 0$.*

Proof. If $[\![\phi[\theta^*]]\!](\mu^*) \leq 0$, then $[\![\phi[\theta^*]]\!](\Sigma) \leq 0$. Since $[\![\phi[\theta]]\!](\Sigma)$ is non-increasing with respect to θ, then for all $\theta \in [\theta^*, \theta_M]$, we also have $[\![\phi[\theta]]\!](\Sigma) \leq 0$.

Since we are utilizing stochastic optimization methods [7,10,8,4] to solve problem (3), if $[\![\phi[\theta^*]]\!](\mu^*) > 0$, then we cannot infer that the system is correct for all parameter values in Θ.

Example 4. *Using Eq. (3) as a cost function, we can now compute the optimal parameter for Example 3 using our toolbox* S-TaLiRo *[10]. In particular, using Simulated Annealing as a stochastic optimization function,* S-TaLiRo *returns* $\theta^* \approx 2.45$ *as optimal parameter for constant input* $u(t) = 99.8046$. *The corresponding temporal logic robustness for the specification* $\square_{[0,2.45]}(\omega \leq 4500)$ *is* -0.0445. *The total number of tests performed for this example was* 500 *and, potentially, the accuracy of estimating* θ^* *can be improved if we increase the maximum number of tests. However, we remark that based on several tests the algorithm converges to a good approximation within* 200 *tests.*

5.2 Non-decreasing Robustness Functions

The case of non-decreasing robustness functions is symmetric to the case of non-increasing robustness functions. In particular, the optimization problem is a maximization problem. We will reformulate the problem of Eq. (2) so that we do not have to solve two separate optimization problems. From (2), we have:

$$\begin{aligned}
&\max_{\theta\in\Theta}\left(\theta + \begin{cases} \gamma - \min_{\mu\in\mathcal{L}_\tau(\Sigma)}[\![\phi[\theta]]\!](\mu) & \text{if } \min_{\mu\in\mathcal{L}_\tau(\Sigma)}[\![\phi[\theta]]\!](\mu) \geq 0 \\ 0 & \text{otherwise} \end{cases}\right) = \\
&= \max_{\theta\in\Theta}\left(\theta + \begin{cases} \gamma + \max_{\mu\in\mathcal{L}_\tau(\Sigma)}\left(-[\![\phi[\theta]]\!](\mu)\right) & \text{if } \max_{\mu\in\mathcal{L}_\tau(\Sigma)}\left(-[\![\phi[\theta]]\!](\mu)\right) \leq 0 \\ 0 & \text{otherwise} \end{cases}\right) = \\
&= \max_{\theta\in\Theta}\left(\theta + \max_{\mu\in\mathcal{L}_\tau(\Sigma)}\begin{cases} \gamma - [\![\phi[\theta]]\!](\mu) & \text{if } -[\![\phi[\theta]]\!](\mu) \leq 0 \\ 0 & \text{otherwise} \end{cases}\right) = \\
&= \max_{\theta\in\Theta}\max_{\mu\in\mathcal{L}_\tau(\Sigma)}\left(\theta + \begin{cases} \gamma - [\![\phi[\theta]]\!](\mu) & \text{if } [\![\phi[\theta]]\!](\mu) \geq 0 \\ 0 & \text{otherwise} \end{cases}\right) \qquad (4)
\end{aligned}$$

where $\gamma \leq \min(\theta_m, 0)$.

The previous discussion is formalized in the following result.

Proposition 4. *Let* θ^* *and* μ^* *be the parameters returned by an optimization algorithm that is applied to the problem in Eq. (4). If* $[\![\phi[\theta^*]]\!](\mu^*) \leq 0$, *then for all* $\theta \in \Theta^* = [\theta_m, \theta^*]$, *we have* $[\![\phi[\theta]]\!](\Sigma) \leq 0$.

Proof. If $[\![\phi[\theta^*]]\!](\mu^*) \leq 0$, then $[\![\phi[\theta^*]]\!](\Sigma) \leq 0$. Since $[\![\phi[\theta]]\!](\Sigma)$ is non-decreasing with respect to θ, then for all $\theta \in [\theta_m, \theta^*]$, we also have $[\![\phi[\theta]]\!](\Sigma) \leq 0$.

Again, if $[\![\phi[\theta^*]]\!](\mu^*) > 0$, then we cannot infer that the system is correct for all parameter values in Θ.

Example 5. *Let us consider the specification* $\phi[\lambda] = \square_{[\lambda,30]}(\omega \leq 4500)$ *on our running example. The specification robustness* $[\![\phi[\theta]]\!](\Delta_\Sigma(u))$ *as a function of* θ *and the input* u *appears in Fig. 5 (left) for constant input signals. The creation*

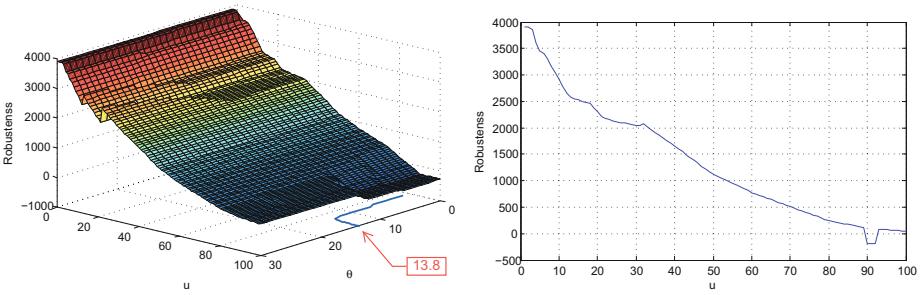

Fig. 5. Example 5. Left: Specification robustness as a function of the parameter θ and the constant input u. Right: The robustness function $[\![\Box_{[12.59,30]}(\omega \leq 4500)]\!](\Delta_\Sigma(u))$.

of the graph required $100 \times 30 = 3,000$ *tests. The contour under the surface indicates the zero level set of the robustness surface, i.e., the* θ *and* u *values for which we get* $[\![\phi[\theta]]\!](\Delta_\Sigma(u)) = 0$. *We remark that the contour is actually an approximation of the zero level set computed by a linear interpolation using the neighboring points on the grid. From the graph, we could infer that* $\theta^* \approx 13.8$ *and that for any* $\theta \in [0, 13.8]$, *we would have* $[\![\phi[\theta]]\!](\Sigma) \leq 0$. *Again, the approximate value of* θ^* *is a rough estimate based on the granularity of the grid.*

Using Eq. (4) as a cost function, we can now compute the optimal parameter for Example 3 using our toolbox S-TaLiRo *[10].* S-TaLiRo *returns* $\theta^* \approx 12.59$ *as optimal parameter for constant input* $u(t) = 90.88$ *within 250 tests. The temporal logic robustness for the specification* $\Box_{[12.59,30]}(\omega \leq 4500)$ *with respect to the input* u *appears in Fig. 5 (right). Some observations: (i) The* $\theta^* \approx 12.59$ *computed by* S-TaLiRo *is actually very close to the optimal value since for* $\theta^* \approx 12.79$ *the system does not falsify any more. (ii) The systematic testing that was used in order to generate the graph was not able to accurately compute a good approximation to the parameter unless even more tests (*> 3000*) are generated.*

6 Experiments and a Case Study

The parametric MTL exploration of embedded systems was motivated by a challenge problem published by Ford in 2002 [11]. In particular, the report provided a simple – but still realistic – model of a powertrain system (both the physical system and the embedded control logic) and posed the question whether there are constant operating conditions that can cause a transition from gear two to gear one and then back to gear two. Such a sequence would imply that the transition was not necessary in the first place.

The system is modeled in Checkmate [17]. It has 6 continuous state variables and 2 Stateflow charts with 4 and 6 states, respectively. The Stateflow chart for the shift scheduler appears in Fig. 6. The system dynamics and switching conditions are linear. However, some switching conditions depend on the inputs to the system. The latter makes the application of standard hybrid system verification tools not a straightforward task.

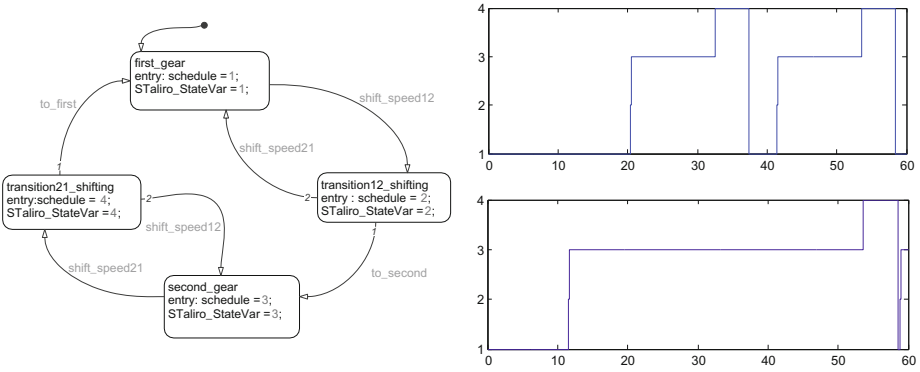

Fig. 6. Left: The shift scheduler of the powertrain challenge problem. Right: Shift schedules. The numbers on the y-axis correspond to the variables in the states of the shift scheduler. Right Top: The shift schedule falsifying requirement ϕ_{e1}. Right Bottom: The shift schedule falsifying requirement $\phi_{e3}[0.4273]$.

In [15], we demonstrated that S-TaLiRo [10] can successfully solve the challenge problem (see Fig. 6) by formalizing the requirement as an MTL specification $\phi_{e1} = \neg\Diamond(g_2 \wedge \Diamond(g_1 \wedge \Diamond g_2))$ where g_i is a proposition that is true when the system is in gear i. Stochastic search methods can be applied to solve the resulting optimization problem where the cost function is the robustness of the specification. Moreover, inspired by the success of S-TaLiRo on the challenge problem, we tried to ask a more complex question. Namely, does a transition exists from gear two to gear one and back to gear two in less than 2.5 sec? An MTL specification that can capture this requirement is $\phi_{e2} = \Box((\neg g_1 \wedge X g_1) \rightarrow \Box_{[0,2.5]}\neg g_2)$.

The natural question that arises is what would be the smallest time for which such a transition can occur? We can formulate a parametric MTL formula to query the model of the powertrain system: $\phi_{e3}[\lambda] = \Box((\neg g_1 \wedge X g_1) \rightarrow \Box_{[0,\lambda]}\neg g_2)$. We have extended S-TaLiRo to be able to handle parametric MTL specifications. The total simulation time of the model was 60 sec and the search interval was $\Theta = [0, 30]$. S-TaLiRo returned $\theta^* \approx 0.4273$ as the minimum parameter found (See Fig. 6) using about 300 tests of the system.

In Table 6, we present some experimental results. Since no other technique can solve the parameter estimation problem for MTL formulas over hybrid systems, we compare our method with the falsification methods that we have developed in the past [12,7]. A detailed description of the benchmark problems can be found in [12,7] and the benchmarks can be downloaded with the S-TaLiRo distribution[2]. In order to be able to compare the two methods, when performing parameter estimation, we regard a parameter value less than the constant in the MTL formula as falsification. Notably, for benchmark problems that are easier to falsify, the parameter estimation method incurs additional cost in the sense of reduced number of falsifications. On the other hand, on hard problem instances,

[2] `https://sites.google.com/a/asu.edu/s-taliro/`

Table 1. Experimental Comparison of Falsification (FA) vs. Parameter Estimation (PE). Each instance was run for 100 times and each run was executed for a maximum of 1000 tests. Legend: **#Fals.**: the number of runs falsified, **Parameter Estimate**: ⟨min, average, max⟩ of the parameter value computed, **dnf**: did not finish.

Benchmark Problem		#Fals.		Parameter Estimate
Specification	Instance	FA	PE	PE
$\phi_2^{AT}[\lambda] = \neg\Diamond(p_1^{AT} \wedge \Diamond_{[0,\lambda]} p_2^{AT})$	$\phi_2^{AT}[10]$	96	84	$\langle 7.7, 9.56, 16.84\rangle$
$\phi_3^{AT}[\lambda] = \neg\Diamond(p_1^{AT} \wedge \Diamond_{[0,\lambda]} p_3^{AT})$	$\phi_3^{AT}[10]$	51	0	$\langle 10.00, 10.22, 14.66\rangle$
$\phi_4^{AT}[\lambda] = \neg\Diamond(p_1^{AT} \wedge \Diamond_{[0,\lambda]} p_2^{AT})$	$\phi_4^{AT}[7.5]$	0	0	$\langle 7.57, 7.7, 8.56\rangle$
$\phi_5^{AT}[\lambda] = \neg\Diamond(p_1^{AT} \wedge \Diamond_{[0,\lambda]} p_2^{AT})$	$\phi_5^{AT}[5]$	0	0	$\langle 7.56, 7.74, 9.06\rangle$
	$\phi_{e3}[2.5]$	dnf	93	$\langle 1.28, 2.26, 6.82\rangle$

the parameter estimation method provides us with parameter ranges for which the system fails the specification. Moreover, on the powertrain challenge problem, the parameter estimation method actually helps in falsifying the system. We conjecture that the reason for this improved performance is that the timing requirements on this problem are more important than the state constraints.

7 Related Work

The topic of testing embedded software and, in particular, embedded control software is a well studied problem that involves many subtopics well beyond the scope of this paper. We refer the reader to specialized book chapters and textbooks for further information [18,19]. Similarly, a lot of research has been invested on testing methods for Model Based Development (MBD) of embedded systems [3]. However, the temporal logic testing of embedded and hybrid systems has not received much attention [20,21,4,22].

Parametric temporal logics were first defined over traces of finite state machines [23]. In parametric temporal logics, some of the timing constraints of the temporal operators are replaced by parameters. Then, the goal is to develop algorithms that will compute the values of the parameters that make the specification true under some optimality criteria. That line of work has been extended to real-time systems and in particular to timed automata [24] and continuous-time signals [9]. The authors in [25,26] define a parametric temporal logic called quantifier free LTL over real valued signals. However, they focus on the problem of determining system parameters such that the system satisfies a given property rather than on the problem of exploring the properties of a given system.

Another related research topic is the problem of Temporal Logic Queries [27,28]. In detail, given a model of the system and a temporal logic formula ϕ, a subformula in ϕ is replaced with a special symbol ?. Then, the problem is to determine a set of Boolean formulas such that if these formulas are placed into the placeholder ?, then ϕ holds on the model.

8 Conclusions

An important stage in Model Based Development (MBD) of embedded control software is the formalization of system requirements. We advocate that Metric Temporal Logic (MTL) is an excellent candidate for formalizing interesting design requirements. In this paper, we have presented a solution on how we can explore system properties using Parametric MTL (PMTL) [9]. Based on the notion of robustness of MTL [6], we have converted the parameter estimation problem into an optimization problem which we solve using S-TaLiRo [10]. Even though this paper presents a method for estimating the range for a single parameter, the results can be easily extended to multiple parameters as long as the robustness function has the same monotonicity with respect to all the parameters. Finally, we have demonstrated that the our method can provide interesting insights to the powertrain challenge problem [11].

Acknowledgments. This work was partially supported by a grant from the NSF Industry/University Cooperative Research Center (I/UCRC) on Embedded Systems at Arizona State University and NSF awards CNS-1116136 and CNS-1017074.

References

1. Lions, J.L., Lbeck, L., Fauquembergue, J.L., Kahn, G., Kubbat, W., Levedag, S., Mazzini, L., Merle, D., O'Halloran, C.: Ariane 5, flight 501 failure, report by the inquiry board. Technical report, CNES (1996)
2. Hoffman, E.J., Ebert, W.L., Femiano, M.D., Freeman, H.R., Gay, C.J., Jones, C.P., Luers, P.J., Palmer, J.G.: The near rendezvous burn anomaly of december 1998. Technical report, Applied Physics Laboratory, Johns Hopkins University (1999)
3. Tripakis, S., Dang, T.: Modeling, Verification and Testing using Timed and Hybrid Automata. In: Model-Based Design for Embedded Systems, pp. 383–436. CRC Press (2009)
4. Nghiem, T., Sankaranarayanan, S., Fainekos, G.E., Ivancic, F., Gupta, A., Pappas, G.J.: Monte-carlo techniques for falsification of temporal properties of non-linear hybrid systems. In: Proceedings of the 13th ACM International Conference on Hybrid Systems: Computation and Control, pp. 211–220. ACM Press (2010)
5. Koymans, R.: Specifying real-time properties with metric temporal logic. Real-Time Systems 2, 255–299 (1990)
6. Fainekos, G.E., Pappas, G.J.: Robustness of temporal logic specifications for continuous-time signals. Theoretical Computer Science 410, 4262–4291 (2009)
7. Sankaranarayanan, S., Fainekos, G.: Falsification of temporal properties of hybrid systems using the cross-entropy method. In: ACM International Conference on Hybrid Systems: Computation and Control (2012)
8. Annapureddy, Y.S.R., Fainekos, G.E.: Ant colonies for temporal logic falsification of hybrid systems. In: Proceedings of the 36th Annual Conference of IEEE Industrial Electronics, pp. 91–96 (2010)
9. Asarin, E., Donzé, A., Maler, O., Nickovic, D.: Parametric Identification of Temporal Properties. In: Khurshid, S., Sen, K. (eds.) RV 2011. LNCS, vol. 7186, pp. 147–160. Springer, Heidelberg (2012)

10. Annapureddy, Y.S.R., Liu, C., Fainekos, G.E., Sankaranarayanan, S.: S-TaLiRo: A Tool for Temporal Logic Falsification for Hybrid Systems. In: Abdulla, P.A., Leino, K.R.M. (eds.) TACAS 2011. LNCS, vol. 6605, pp. 254–257. Springer, Heidelberg (2011)
11. Chutinan, A., Butts, K.R.: Dynamic analysis of hybrid system models for design validation. Technical report, Ford Motor Company (2002)
12. Abbas, H., Fainekos, G.E., Sankaranarayanan, S., Ivancic, F., Gupta, A.: Probabilistic temporal logic falsification of cyber-physical systems. ACM Transactions on Embedded Computing Systems (2011) (in press)
13. Alur, R., Henzinger, T.A.: Real-Time Logics: Complexity and Expressiveness. In: Fifth Annual IEEE Symposium on Logic in Computer Science, pp. 390–401. IEEE Computer Society Press, Washington, D.C (1990)
14. Zhao, Q., Krogh, B.H., Hubbard, P.: Generating test inputs for embedded control systems. IEEE Control Systems Magazine, 49–57 (August 2003)
15. Fainekos, G., Sankaranarayanan, S., Ueda, K., Yazarel, H.: Verification of automotive control applications using s-taliro. In: Proceedings of the American Control Conference (2012)
16. Donze, A., Maler, O.: Robust Satisfaction of Temporal Logic over Real-Valued Signals. In: Chatterjee, K., Henzinger, T.A. (eds.) FORMATS 2010. LNCS, vol. 6246, pp. 92–106. Springer, Heidelberg (2010)
17. Silva, B.I., Krogh, B.H.: Formal verification of hybrid systems using CheckMate: a case study. In: Proceedings of the American Control Conference, vol. 3, pp. 1679–1683 (2000)
18. Conrad, M., Fey, I.: Testing automotive control software. In: Automotive Embedded Systems Handbook. CRC Press (2008)
19. Koopman, P.: Better Embedded System Software. Drumnadrochit Education LLC (2010)
20. Plaku, E., Kavraki, L.E., Vardi, M.Y.: Falsification of LTL Safety Properties in Hybrid Systems. In: Kowalewski, S., Philippou, A. (eds.) TACAS 2009. LNCS, vol. 5505, pp. 368–382. Springer, Heidelberg (2009)
21. Tan, L., Kim, J., Sokolsky, O., Lee, I.: Model-based testing and monitoring for hybrid embedded systems. In: Proceedings of the 2004 IEEE International Conference on Information Reuse and Integration, pp. 487–492 (2004)
22. Zuliani, P., Platzer, A., Clarke, E.M.: Bayesian statistical model checking with application to simulink/stateflow verification. In: Proceedings of the 13th ACM International Conference on Hybrid Systems: Computation and Control, pp. 243–252 (2010)
23. Alur, R., Etessami, K., La Torre, S., Peled, D.: Parametric temporal logic for model measuring. ACM Trans. Comput. Logic 2, 388–407 (2001)
24. Di Giampaolo, B., La Torre, S., Napoli, M.: Parametric Metric Interval Temporal Logic. In: Dediu, A.-H., Fernau, H., Martín-Vide, C. (eds.) LATA 2010. LNCS, vol. 6031, pp. 249–260. Springer, Heidelberg (2010)
25. Fages, F., Rizk, A.: On temporal logic constraint solving for analyzing numerical data time series. Theor. Comput. Sci. 408, 55–65 (2008)
26. Rizk, A., Batt, G., Fages, F., Soliman, S.: On a Continuous Degree of Satisfaction of Temporal Logic Formulae with Applications to Systems Biology. In: Heiner, M., Uhrmacher, A.M. (eds.) CMSB 2008. LNCS (LNBI), vol. 5307, pp. 251–268. Springer, Heidelberg (2008)
27. Chan, W.: Temporal-Logic Queries. In: Emerson, E.A., Sistla, A.P. (eds.) CAV 2000. LNCS, vol. 1855, pp. 450–463. Springer, Heidelberg (2000)
28. Chechik, M., Gurfinkel, A.: TLQSolver: A Temporal Logic Query Checker. In: Hunt Jr., W.A., Somenzi, F. (eds.) CAV 2003. LNCS, vol. 2725, pp. 210–214. Springer, Heidelberg (2003)