Ant Colonies for Temporal Logic Falsification of Hybrid Systems

Yashwanth Singh Rahul Annapureddy and Georgios E. Fainekos

Abstract— **The Extended Ant Colony Optimization (EACO) metaheuristic is applied to the problem of Metric Temporal Logic (MTL) falsification of nonautonomous hybrid systems. The goal of the MTL falsification problem is to detect operating conditions and input signals that will generate system trajectories that do not satisfy a user provided specification in MTL. A new method for parameterizing the input space using splines and Hermitian polynomials is introduced and the performance of the new algorithm is compared against Monte Carlo testing techniques. It is envisioned that this work will help practitioners in designing complex control or mixed-signal systems.**

I. Introduction

Hybrid systems include a wide variety of engineered and/or natural systems which require in their mathematical models both discrete and continuous components. Typical examples of such systems can range from aircraft collision avoidance controllers [1] to power electronics [2] to biological systems [3]. The design problem for hybrid systems is a challenging one where only special cases of system dynamics and user specifications can be handled [4] while the analysis and verification problem is undecidable [5], in general. Therefore, system designers still resort to manual component design, manual synthesis of the components and, finally, testing of the final system.

However, testing (or the very related problem of parameter estimation) can be cumbersome and time consuming for the test engineers. Especially since in non-linear systems with complex discrete system dynamics the bad behaviors cannot be found within the extreme or "corner" cases. Towards this goal several authors have proposed methods for the automatic testing of hybrid systems for safety properties [6]–[9] or more complex specifications [10]–[14].

In this paper, we continue along the lines of our previous work [10] on the problem of Metric Temporal Logic (MTL) falsification. MTL was introduced by Koymans [15] for expressing real-time requirements of systems. It is a formal language that can express most of the requirements of interest for hybrid systems and at the same time it is very closely related to natural language. Both attributes make MTL ideal for adoption by test engineers.

Informally, the problem of MTL falsification is defined as a search through the space of possible trajectories of a system in order to detect a system trajectory that does not satisfy the MTL specification. Even though the semantics of an MTL formula as defined by Koymans are Boolean, a multi-valued interpretation is possible as we have demonstrated in our earlier work [16]. Briefly, in [16], we introduced robust semantics for MTL, which not only indicate whether a trajectory satisfies a specification or not, but also how robustly it does so. In this context, robustness is simply defined as the magnitude of perturbations that a trajectory can tolerate without changing its Boolean truth value.

The notion of MTL robustness is a valuable tool in the falsification problem since it allows us to convert a decision problem into an optimization problem. In other words, an optimization algorithm can now be utilized in order to search for system trajectories that minimize the robustness of the system. Unfortunately, besides very simple cases, the robustness of a trajectory is not a convex function and, moreover, the trajectories of a system cannot be put in a closed form. Therefore, we have to resort to stochastic and/or heuristic nonlinear optimization algorithms.

Our contribution in this work is twofold. First, we modify the Extended Ant Colony Optimization (EACO) algorithm [17] to apply it to the MTL falsification problem. EACO is an extension to continuous domains of the classical discrete Ant Colony Optimization (ACO) algorithm [18]. Second, we parameterize the input signal space using interpolating polynomials, i.e., cubic spline and Hermite polynomials. Spline curves have been used in the past in control theory [19], but to the author's best knowledge never in the context of system falsification.

II. Problem Formulation

In this paper, we address the problem of temporal logic testing of non-autonomous hybrid systems with parametric uncertainties. The aforementioned class of systems can be thought as a generalization of the classic model of hybrid automata [5]. In brief, in this paper we will be using the following formal model.

Definition 2.1 (Nonautonomous Hybrid Automaton): A *hybrid automaton* $\mathcal{H}$ consists of tuple (H, H_0, U, P, $\mathfrak{F}$, E, $\mathfrak{G}$, $\mathfrak{R}$), where, $H = L \times X$ is the state space of the hybrid automaton with L a finite set of discrete locations (system operating modes) and $X \subseteq \mathbb{R}^n$ the continuous state space, $H_0 \subseteq H$ is a compact set of initial conditions for the system, $U \subseteq \mathbb{R}^m$ is a compact set that constraints the inputs to the system, $P \subseteq \mathbb{R}^q$ is a compact set where the uncertain or unknown parameters take values from, $\mathfrak{F} : L \to (X \times U \times P \to \mathbb{R}^n)$ describes the continuous system dynamics in each location, $E \subseteq L \times L$ is a set of edges with nodes from L which describes which transitions are possible between different system locations,

This work was supported in part by NSF CNS-1017074, NSF IIP-0856090, and by the industry partners of the NSF I/UCRC for Embedded Systems. The authors are with the School of Computing, Informatics and Decision Systems Engineering at Arizona State University. E-mail: {Yashwanthsingh.Annapureddy+,fainekos+}@asu.edu

$\mathfrak{G} : E \to 2^{X \times U}$ returns the conditions that will enable a transition from E, and, finally, $\mathfrak{R} : E \to (X \times U \to X)$ is a reset function.

We denote the reals by $\mathbb{R}$, the positive reals by $\mathbb{R}_+$, the natural numbers by $\mathbb{N}$ and the integers by $\mathbb{Z}$. We will also use $\overline{\mathbb{R}} = \mathbb{R} \cup \{\pm\infty\}$ and $\overline{\mathbb{Z}} = \mathbb{Z} \cup \{\pm\infty\}$.

Intuitively, the locations l model the behavior of the system under different dynamics or control laws. That is, in each location $l \in L$, the system dynamics are

$$\dot{x} = f_l(x, u, p), \quad u(t) \in U, p(t) \in P \tag{1}$$

where x is the continuous state vector of the system, u is an input signal and p an unknown parameter (possibly time varying). For brevity, when $f \in \mathfrak{F}$, we write $f_l(\cdot, \cdot, \cdot)$ instead of $f(l, \cdot, \cdot, \cdot)$. If f_l describes a closed loop system, then we can think of the inputs u as disturbances. We will make the assumption that the system $\dot{x} = f_l(x, u, p)$ has a unique solution $\xi_l(t)$. In detail, we assume that f_l is smooth for all $l \in L$, and $u : \mathbb{R}_+ \to U$ and $p : \mathbb{R}_+ \to P$ are piecewise continuous. Informally, the system trajectories are piecewise continuous functions. In each location l, the system evolves under the system dynamics f_l until it enters a set defined by the guard $\mathfrak{G}(e)$ of a transition $e = (l, l')$, i.e., $\xi_l(t_*) \in \mathfrak{G}(e)$ for some time t_*. At this point, the system changes its location to l' and it flows under the system dynamics $f_{l'}$ with initial conditions defined by $\xi_{l'}(0) = \mathfrak{R}_e(\xi_l(t_*), u(t_*))$. A hybrid system trajectory $h : \mathbb{N} \times \mathbb{R}_+ \to L \times \mathbb{R}^n$ consists of concatenations of the pairs (l, ξ_l).

As it is well known, in most of the cases, system (1) does not have a closed form solution. Even when it does - as in the case of linear systems - the solution is still bound to numerical errors. Therefore, in this work, we assume that we have an approximation $\hat{h} : \mathbb{N} \to L \times \mathbb{R}^n$ to the actual system trajectory h. In other words, we assume that there is a increasing (under the dictionary order) mapping $\tau : \mathbb{N} \to \mathbb{N} \times \mathbb{R}_+$ such that there exists some $\varepsilon^* > 0$ such that $\sup_i \|\hat{h}(i) - h(\tau(i))\| < \varepsilon^*$ where $\| \cdot \|$ is the usual Euclidean norm. A discussion regarding the properties of numerical simulators of hybrid systems can be found in [20]. In the following, we assume that the behavior of the hybrid system $\mathcal{H}$ is fully characterized by the solutions $\eta = (\hat{h}, \hat{\tau})$ as returned by a numerical simulator (e.g., Simulink or ode45 solver). Here, $\hat{\tau} : \mathbb{N} \to \mathbb{R}_+$ is the sampling function as returned by the simulator. We denote all such solutions η of $\mathcal{H}$ by $\mathcal{L}(\mathcal{H})$.

In order to reason about functional properties of systems, we have to define certain sets of interest in $\mathbb{R}^n$. For example, we would like to know whether all trajectories of a system attain a value in the set $[10, +\infty)$. We do so by using a set of atomic propositions AP which label subsets of $\mathbb{R}^n$. In other words, we define an observation map $\mathcal{O} : AP \to 2^H$ such that for each $\pi \in AP$ the corresponding set is $\mathcal{O}(\pi) \subseteq H$.

Metric Temporal Logic (MTL) formulas are built over a set of propositions, the set AP in our case, using combinations of the traditional and temporal operators. Traditional logic operators are the *conjunction* ($\wedge$), *disjunction* ($\vee$), *negation* ($\neg$), *implication* ($\to$) and *equivalence* ($\leftrightarrow$). Some of the temporal operators, which we will be using here, are *eventually* ($\Diamond_{\mathcal{I}}$), *always* ($\Box_{\mathcal{I}}$), *until* ($\mathcal{U}_{\mathcal{I}}$) and *release* ($\mathcal{R}_{\mathcal{I}}$). The subscript $\mathcal{I}$ imposes timing constraints on the temporal operators. Metric Temporal Logic specifications can describe the usual properties of interest for control design problems such as (bounded time) **reachability**, e.g., $\Diamond_{[1,5)}\pi_1$ with $\mathcal{O}(\pi_1) = (-\infty, -10]$, and **safety**, e.g., $\Box_{[2,+\infty)}\neg\pi_1$. Beyond the usual properties, MTL can capture sequences of events (e.g., $\Diamond_{\mathcal{I}_1}(\pi_1 \wedge \Diamond_{\mathcal{I}_2}(\pi_2 \wedge \Diamond_{\mathcal{I}_3}\pi_3))$) and infinite behaviors (e.g., **periodic** behaviors : $\Box(\pi_1 \to \Diamond_{[0,2]}\pi_2)$).

In this work, we interpret MTL formulas over the approximate trajectories of a given system $\mathcal{H}$. We denote the satisfaction of an MTL formula φ under a map $\mathcal{O}$ over a trajectory $\eta = (\hat{h}, \hat{\tau})$ by $(\hat{h}, \hat{\tau}) \models \varphi$. For such systems and specifications, we formally solve the following problem.

Problem 2.1 (MTL System Falsification): Given an MTL specification φ built over the set AP, an observation map $\mathcal{O}$, a hybrid automaton $\mathcal{H}$ and a maximum simulation time T, find a trajectory $(\hat{h}, \hat{\tau})$ such that $(\hat{h}, \hat{\tau}) \not\models \varphi$.

In other words, we try to determine if there exists a behavior of $\mathcal{H}$ in the time domain $[0, T]$ which does not satisfy our system specification φ. The challenges in solving Problem 2.1 are twofold. First, we have an infinite number of initial conditions to verify, i.e., the set X_0, and, also, the parameters of the system are not known in advance, i.e., they belong to the uncountably infinite set P. In addition, we have unknown inputs to the system and, in most of the cases, the system dynamics are nonlinear and potentially high dimensional. Second, the satisfaction of the specification with respect to a system trajectory cannot be computed analytically and, moreover, MTL semantics are defined over Boolean truth values. The former problem, whose solution is one of the contributions of this paper (Section IV), is addressed by developing a new continuous optimization algorithm and by parameterizing the input and parameter search space using splines. The latter problem is addressed by utilizing the theory of MTL robustness that we have developed in our previous work [16].

Example 2.1 (Δ-Σ Modulation): We consider a Simulink model of a third-order Δ-Σ modulator. The modulator is shown in Fig. 1 and its numerical values are set as described in [21]. The three state variables $x = [x_1 \; x_2 \; x_3]^T$ model the state of the three integrators. The integrators saturate if their values exceed the range $[-1, 1]$. The initial conditions $x(0)$ belong to the box $[-0.1, 0.1]^3$. We need to verify that the integrators never saturate when the input signal is within the range $[-0.5, 0.5]$. In other words, we wish to find initial values and control inputs that falsify the MTL formula $\varphi = \Box(x \in [-1, 1]^3)$. The time interval of interest is $[0, 9.0]$.

III. MTL FALSIFICATION

Metric Temporal Logic (MTL), which was introduced by Koymans [15], is a popular formalism for expressing real-time requirements. However, the semantics of MTL are by definition Boolean and, thus, it does not capture how robustly a system satisfies a specification. In our previous work [16], we have introduced a notion of robustness for MTL that quantifies how robustly a trajectory satisfies a specification.

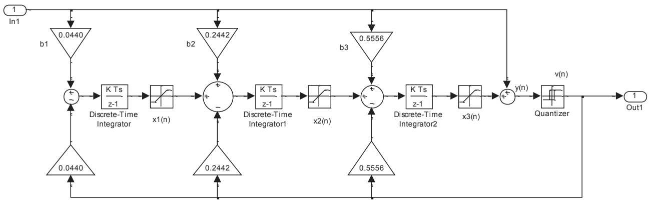

Fig. 1. A Simulink model of a third-order Δ-Σ modulator.

Intuitively, a robust trajectory should be able to tolerate perturbations and still satisfy the same specification.

Such a notion of robustness provides a natural guide for system exploration. Recall that our goal in Problem 2.1 is to falsify the system. It is intuitive that incorrect system behaviors are most probably near operating regions that generate less robust behaviors. Based on that observation, in our previous work [10], we utilized robustness as a cost function for a global non-convex nonlinear optimization problem. In this section, we review some results from [10], [16] and we introduce a parameterization of the search space using spline functions in order to convert the falsification problem into an optimization problem.

A. Metrics

We first consider state-spaces equipped with a metric function that provides a notion of distance between states.

Definition 3.1 (Metric): A *metric* d, defined over a state-space X is a function $d : X \times X \to \mathbb{R}_+$. The metric d maps pairs of states to non-negative extended real numbers, satisfying the following properties:

1) $d(x_1, x_2) = 0$ iff $x_1 = x_2$,
2) $d(x_1, x_2) = d(x_2, x_1)$, and
3) $d(x_1, x_3) \le d(x_1, x_2) + d(x_2, x_3)$.

If the second condition is dropped from the definition, then d is termed a *quasi-metric*.

Given a metric d, a radius $\varepsilon > 0$ and a point $x \in X$, the open ε-ball centered at x is defined as $B_d(x, \varepsilon) = \{y \in X \mid d(x, y) < \varepsilon\}$. Finally, if η and η' are two system trajectories that take values in a metric space with metric d, we will use ρ_d to denote the metric $\rho_d(\eta, \eta') = \sup_t\{d(\eta(t), \eta'(t))\}$.

In this work, we will be using the Euclidean metric $d_e(x_1, x_2) = \|x_1 - x_2\|^2$ when we are interested only on the continuous state trajectory. When the specifications are with respect to the hybrid state space, we will be using the generalized hybrid metric d_h that we introduced in [10]:

$$d_h(v, v') = \begin{cases} (0, d_e(x, x')) & \text{if } l = l' \\ (\pi(l, l'), +\infty) & \text{otherwise} \end{cases}$$

where $v = (l, x) \in H$, $v' = (l', x') \in H$ and π is the shortest path metric (a quasi-metric) on the graph $G = (L, E)$.

B. Robustness of Trajectories

We briefly present the robust interpretation of MTL formulas. Details are available in our previous work [16].

Definition 3.2 (MTL Syntax): Let AP be the set of atomic propositions and $\mathcal{I}$ be any non-empty interval of $\mathbb{R}_+$. The set MTL of all well-formed MTL formulas is inductively defined as $\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi\,\mathcal{U}_{\mathcal{I}}\varphi$, where $p \in AP$ and $\top$ is *true*.

We provide semantics that maps an MTL formula φ and a trajectory η to a value drawn from a subset of the linearly ordered set $\overline{\mathbb{Z}} \times \overline{\mathbb{R}}$ (see [10] for details). The semantics for the atomic propositions evaluated for $\eta(i)$ consists of the distance between $\eta(i)$ and the set $\mathcal{O}(p)$ labeling atomic proposition p. Intuitively, this distance represents how robustly the point $\eta(i)$ lies within (or outside) the set $\mathcal{O}(p)$.

Definition 3.3 (Signed Distance): Let $x \in X$ be a point, $S \subseteq X$ be a set and d be a metric on X. We define the signed distance from x to S to be

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

If this distance is zero, then the smallest perturbation of the point x can affect the outcome of $x \in \mathcal{O}(p)$. We denote the robust valuation of the formula φ over the trajectory η at time i by $[\![\varphi]\!]_{d_h}(\eta, i)$. Formally, $[\![\cdot]\!]_{d_h} : (MTL \times (2^H)^{AP}) \to (\mathcal{L}(\mathcal{H}) \times \mathbb{N} \to \overline{\mathbb{Z}} \times \overline{\mathbb{R}})$.

Definition 3.4 (Robust Semantics): Let $\eta \in \mathcal{L}(\mathcal{H})$ and $\mathcal{O} \in (2^H)^{AP}$, then the robust semantics of any formula $\varphi \in MTL$ with respect to η is recursively defined as follows

$$\begin{aligned}
[\![\top]\!]_{d_h}(\eta, i) &:= (+\infty, +\infty) \\
[\![p]\!]_{d_h}(\eta, i) &:= \mathbf{Dist}_{d_h}(\hat{h}(i), \mathcal{O}(p)) \\
[\![\neg\varphi_1]\!]_{d_h}(\eta, i) &:= -\,[\![\varphi_1]\!]_{d_h}(\eta, i) \\
[\![\varphi_1 \vee \varphi_2]\!]_{d_h}(\eta, i) &:= \max([\![\varphi_1]\!]_{d_h}(\eta, i), [\![\varphi_2]\!]_{d_h}(\eta, i)) \\
[\![\varphi_1\,\mathcal{U}_{\mathcal{I}}\varphi_2]\!]_{d_h}(\eta, i) &:= \max_{i' \in (\hat{\tau}(i) + \mathcal{I})} \min\big([\![\varphi_2]\!]_{d_h}(\eta, i'), \\
&\qquad \min_{i < i'' < i'} [\![\varphi_1]\!]_{d_h}(\eta, i'')\big)
\end{aligned}$$

where $i \in \mathbb{N}$ and $\hat{\tau}(i) + \mathcal{I} = \{j \mid \exists t \in \mathcal{I} \,.\, \hat{\tau}(j) = \hat{\tau}(i) + t\}$. If $i = 0$, then we simply write $[\![\varphi]\!]_{d_h}(\eta)$.

Note that Boolean MTL satisfiability reduces to an application of Def. 3.4 wherein the range of the valuation function is the Boolean set $\mathbb{B} = \{\mathbf{T}, \mathbf{F}\}$ instead of $\overline{\mathbb{Z}} \times \overline{\mathbb{R}}$. It is easy to show that if the trajectory satisfies the property, then its robustness is non-negative and, similarly, it the trajectory does not satisfy the property, then its robustness is non-positive. The following result holds [16].

Theorem 3.1: Given a hybrid automaton $\mathcal{H}$, formula $\varphi \in MTL$, an observation map $\mathcal{O} \in (2^H)^{AP}$ and a trajectory $\eta \in \mathcal{L}(\mathcal{H})$, the following hold: (1) If $\eta \models \varphi$, then $[\![\varphi]\!]_{d_h}(\eta) \ge (0, 0)$. In other words, η satisfies the formula φ if its distance valuation is non-negative. Conversely, if $[\![\varphi]\!]_{d_h}(\eta) > (0, 0)$, then $\eta \models \varphi$. (2) If $\varepsilon = [\![\varphi]\!]_{d_h}(\eta) \neq (0, 0)$, then for all $\eta' \in B_{\rho_{d_h}}(\eta, |\varepsilon|)$, we have $\eta \models \varphi$ iff $\eta' \models \varphi$. I.e, ε defines a *robustness tube* around the trajectory such that other "nearby" trajectories lying inside this tube also satisfy φ.

Theorem 3.1 establishes the robust semantics of MTL as a natural measure of trajectory robustness. Namely, a trajectory is ε robust with respect to an MTL specification φ, if it can tolerate perturbations up to size ε and still maintain its current Boolean truth value. Alternatively, a trajectory with the opposite outcome for φ, if it exists, has a distance of at least ε away.

C. Optimization Problem

Using Theorem 3.1, we can reduce Problem 2.1 to the following optimization problem.

Problem 3.1 (MTL Robustness Minimization): Given an MTL specification ϕ built over the set AP, an observation map $\mathcal{O}$, a hybrid automaton $\mathcal{H}$ and a time T, solve the optimization problem $\inf_{\eta \in \mathcal{L}(\mathcal{H})} [\![\varphi]\!]_{d_h}(\eta)$.

If we find some η such that $[\![\varphi]\!]_{d_h}(\eta) < (0,0)$, then we have effectively falsified the system. However, even if we do not find a trajectory with negative robustness, a robustness value that is close to $(0,0)$ will indicate that the system is *not robust enough*. This could also be valuable information to the system designer.

In general, it is not straightforward how to search for a solution in the trajectory space of $\mathcal{H}$ unless the system is deterministic and the input signals and unknown parameters are constant. When considering time-varying input signals (or parameters) to a system $\mathcal{H}$, then we need to define the value of the m-dimensional input signal (q-dimensional parameter value) for any point in time in the interval $[0,T]$. In our previous work [10], we experimented with piecewise constant signals in order to reduce the dimensionality of the search space of the input signals. However, such a function approximation is still not fine enough when we consider the size of the integration steps of the simulators and, thus, not realistic.

In this work, we propose to use cubic spline and Hermite functions to parameterize the input signal space, as well as possible time varying system parameters. For example, assume that we would like to approximate an input signal $u \in U^{[0,T]}$. A one dimensional cubic spline function [19] is a piecewise polynomial function which is continuous and matches $u^{(j)}$ (the j-th component of u) at a number of control points (*knots* in the spline terminology). In detail, assume that the time interval $[0,T]$ is divided into r subintervals of equal length. Then, we have a sequence of time points $t_0, t_1, \ldots, t_r$ with $t_0 = 0$ and $t_r = T$ such that if we define $T_i = [t_i, t_{i+1}]$, then $\cup_{i=0}^{r-1} T_i = [0,T]$. A cubic spline $s^{(j)} : [0,T] \to U_j$ approximating function $u^{(j)} : [0,T] \to U_j$ must satisfy the following conditions: (1) $\forall i = 0,1,\ldots,r-1 \;.\; \forall t \in T_i \;.\; s^{(j)}(t) = s_i^{(j)}(t) = \sum_{k=0}^{3} c_{ik}(t-t_i)^k$ for some $c_{ik} \in \mathbb{R}$ (2) $\forall i = 0,1,\ldots,r \;.\; s^{(j)}(t_i) = u^{(j)}(t_i)$ (3) $\forall i = 0,1,\ldots,r-1 \;.\; s_i^{(j)}(t_i) = s_{i+1}^{(j)}(t_i)$ (4) $\forall i = 0,1,\ldots,r-1 \;.\; \dot{s}_i^{(j)}(t_i) = \dot{s}_{i+1}^{(j)}(t_i)$ (5) $\forall i = 0,1,\ldots,r-1 \;.\; \ddot{s}_i^{(j)}(t_i) = \ddot{s}_{i+1}^{(j)}(t_i)$, and (6) $\ddot{s}^{(j)}(0) = \ddot{s}^{(j)}(T) = 0$

Therefore, given a sequence of knots $\{t_i\}_{i \le r}$ and the corresponding values of the desired input signal $\{u(t_i)\}_{i \le r}$, we can generate a function $s = [s^{(1)}\ s^{(2)}\ \ldots\ s^{(m)}]^T$ which can approximate u at any point in time $t \in [0,T]$. Thus, the optimization problem over the input signal space reduces to a search space of $r * m$ continuous variables. Similarly, if we have $k \le q$ time varying parameters, we will need $r * k$ additional continuous variables.

If we now consider a continuous vector $y \in Y = X_0 \times U_* \times P_*$, then we can generate a unique trajectory η from $\mathcal{H}$, which we denote by $\eta = \mathcal{H}(y)$. Here, U_* and P_* capture the constraints on the inputs and the unknown parameters. In detail, we assume that all the m-input signals are time varying. Therefore, $U_* = U^r$. Let $P = P_1 \times P_2 \times \ldots \times P_q$. If $k \le q$ parameters are constant corresponding to the sets $P_1, P_2, \ldots, P_k$, then $P_* = \prod_{i=1}^{k} P_i \times (\prod_{i=k+1}^{q} P_i)^r$

Problem 3.2 (MTL Robustness Minimization): Given an MTL specification ϕ built over the set AP, an observation map $\mathcal{O}$, a hybrid automaton $\mathcal{H}$ and a time T, solve the optimization problem $\inf_{y \in X_0 \times U_* \times P_*} [\![\varphi]\!]_{d_h}(\mathcal{H}(y))$.

Besides a few very simple cases, the optimization problem above is non-convex and nonlinear. Therefore, we have to utilize stochastic continuous optimization techniques. This is the subject of the next section.

IV. Extended Ant Colony Optimization

a) ACO/TSP: Ant Colony Optimization (ACO) [18] is a metaheuristic optimization algorithm inspired by the foraging behavior of ants. ACO can be best conceptualized when applied to the Traveling Salesman Problem (TSP). Recall that the goal in TSP is to find the shortest path connecting a number of nodes (or towns) on a graph without visiting a node twice (besides the initial node). In ACO/TSP, artificial ants are moving from town to town guided by pheromone trails and local distance based decisions and searching for the optimal (i.e. the shortest) route. An ant chooses an edge on the graph with probability that is proportional to the amount of pheromone on that edge. A cycle is defined as the ensemble of tours made by simultaneously moving ants. Upon completion of each cycle, the pheromone trails over the links between towns are updated through evaporation and deposition. New pheromone is added only to the links used by the ants in a quantity which is proportional to the total tour length of the corresponding ant. In general, shorter paths accumulate more pheromone, which in turn attracts more ants and, thus, the algorithm converges quickly to an optimal path on the graph.

b) EACO/MTL: In [17], we introduced a continuous optimization version of ACO, namely the Extended Ant Colony Optimization (EACO) algorithm and applied it to the problem of inverse design of airfoils EACO/IDAS. In this paper, we modify EACO/IDAS so it can be applied to the problem of MTL falsification of hybrid systems. In EACO/MTL, each ant should visit a number of territories in a fixed sequence. In our case, the territories correspond to the components of the vector y and the visiting sequence is defined by the natural ordering of the components of y. Each territory captures the constraints on the initial values X_0 as well as the constraints U and P on the possible values of the input space and the parameter space. Since the sets X_0, U and P are compact, each territory is a closed and bounded one-dimensional set (interval).

Following EACO/IDAS, we divide each territory i into a number N_i of finite regions which an ant can visit. All regions j inside a territory i have the same length and are represented by their middle point μ_{ij}. At cycle c, each region j in a territory i contains some amount of pheromone which we denote by $\tau_{ij}(c)$. An ant chooses a region j to visit in a territory i with probability

$$p_c(j|i) = \frac{\tau_{ij}(c)}{\sum_{j=1}^{N_i} \tau_{ij}(c)} \tag{2}$$

and visits all territories in each cycle. Currently, for each ant k the exact location $\theta_{ij}^{(k)}(c)$ within each region j of territory i is chosen at random with uniform probability distribution. The intuition behind this approach is that we want to detect promising regions in the state space which should be explored further.

At the end of each cycle, i.e., when all M ants have finished their tours and have produced a set of candidate solutions $\{y^{(k)}\}_{k\leq M}$, we need to update the pheromone distribution on each territory. First, some amount of pheromone evaporates and, then, the ants deposit new pheromone to the regions that they visited. At any territory i and region j, the pheromone at the next cycle $c+1$ is updated as follows:

$$\tau_{ij}(c+1) = (1-\sigma)\tau_{ij}(c) + \sum_{k=1}^{M} \Delta\tau_{ij}^{(k)}(c) \tag{3}$$

where σ is the rate that the pheromone evaporates, and

$$\Delta\tau_{ij}^{(k)}(c) = \frac{\exp(-\|\theta_{ij}^{(k)}(c) - \mu_{ij}\|^2/e)}{a + [\![\varphi]\!](\mathcal{H}(y^{(k)}))} \tag{4}$$

where e is a parameter defining the decay rate of the exponential distribution and a is a user-defined constant parameter.

Additional heuristics on the pheromone updating rule can be applied in order to increase the speed of convergence to a good solution. Some common heuristics are (a) the use of upper and lower bounds, (b) smoothing, and (c) the regular re-initialization of pheromone distributions. Further details can be found in [17] and the references therein.

Example 4.1 ($\Delta-\Sigma$ modulator from Example 2.1): For this example, we used $M = 10$ ants each starting in a different territory, the initial pheromone value was set to $\tau_{ij}(0) = 10^{-10}$, the evaporation parameter was set to $\sigma = 0.9$ and the constants e and a were set to the following values: $e = 0.05$ and $a = 0.001$. We have chosen to let only the ant that has computed the best solution in each cycle to deposit pheromone. For this example, we have fixed the initial state to be $x_0 = [0\,0\,0]^T$ and the sampling rate to be 0.05. The input space was parameterized using 10 territories (knots) and each knot was placed at i time units for $i = 0, 1, \ldots, 9$ (recall that the simulation time is 9). In other words, we have an optimization problem with 10 continuous variables. Each territory was divided into 31 regions, i.e. $N_1 = N_2 = \ldots = N_{10} = 31$. The maximum number of cycles was set 200, i.e., we allow for a maximum of 2000 individual tests. The knots for this particular example are constraint within the interval $[-0.45, 0.45]$. We keep only input signals whose value is constraint in the interval $[-0.5, 0.5]$ and we penalize any spline function that exceeds those bounds. We leave it as future research to generate interpolating curves that satisfy other constraints, too. The falsifying input signal appears in Figure 2, while the system trajectory that falsifies the specification appears in Figure 3.

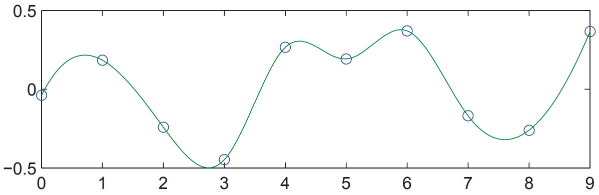

Fig. 2. The input signal and spline knots of Example 4.1.

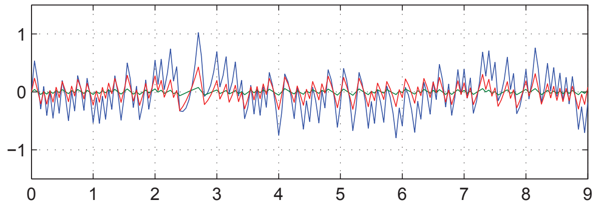

Fig. 3. The state vector of Example 4.1.

V. Numerical Experiments

All the numerical experiments were run on an Intel(R) Core(TM)2 Quad 3.00GHz with 8.00 GB of memory. The operating system was Windows 7 64-bit and the Matlab was version R2009b. In order to assess the performance of EACO/MTL, we compared it with Monte-Carlo (MC) testing on two benchmark examples. Due to space limitations, we cannot present the benchmark examples in detail, but they can be found in [10]. In brief, the Aircraft Benchmark (AB) example has a 2-dimensional input and a 3-dimensional state space while the Navigation Benchmark (NB) is a 4-dimensional hybrid system with 16 locations. All methods divide time into 10 intervals. MC uses piecewise constant signals while EACO/MTL uses piecewise cubic Hermite interpolating polynomials (for the Aircraft Benchmark). Therefore, the falsification problem reduces to an optimization problem with 23 continuous variables for the Aircraft Benchmark (3 for the initial state and 2×10 for the input space) and with 4 continuous variables for the Navigation Benchmark. All the parameters for the EACO/MTL were the same as in Example 4.1. EACO/MTL used 20 ants.

The results are presented in Table I. Even though the results in this paper are preliminary, it is evident that the performance of EACO/MTL is similar to the performance of the Monte-Carlo falsification algorithm. Currently, we have not optimized the parameters of EACO for the MTL falsification problem and we are using the same parameters as in EACO/IDAS [17]. We expect the performance to increase after we fine-tune the parameters of EACO and after we utilize additional heuristics such as upper and lower bounds on the pheromone, multiple ant colonies, different pheromone deposition policies etc.

TABLE I

EXPERIMENTAL COMPARISON OF EACO/MTL (AN) WITH MONTE-CARLO (MC) MTL FALSIFICATION. LEGEND: **#FALS.**: NUMBER OF INSTANCES FALSIFIED, **#TESTS**: MINIMUM, AVERAGE AND MAXIMUM TESTS REQUIRED FOR FALSIFICATION.

Bench.	ψ	#Run	#Iter.	#Fals.		Robustness $\langle min, avg, max\rangle$		#Tests
				AN	MC	AN	MC	AN
AB	$\neg(\Box_{[.5,1.5]} a \wedge \Diamond_{[3,4]} b)$	100	500	100	88	$\langle 0, 0, 0\rangle$	$\langle 0, 1.2, 18.6\rangle$	$\langle 20, 45.34, 420\rangle$
AB	$\neg(\Box_{[0,4]} c \wedge \Diamond_{[3.5,4]} d)$	100	1000	94	100	$\langle 0, .047, 1.55\rangle$	$\langle 0, 0, 0\rangle$	$\langle 20, 266.8, 1000\rangle$
AB	$\Diamond_{[1,3]} e$	100	2000	82	81	$\langle 0, 1.19, 20.47\rangle$	$\langle 0, .9, 40\rangle$	$\langle 20, 601.19, 2000\rangle$
AB	$\Diamond_{[.5,1]} f \wedge \Box_{[3,4]} g$	100	2500	0	0	$\langle 9.71, 10.62, 12.79\rangle$	$\langle 9.5, 9.7, 10.1\rangle$	$\langle 2500, 2500, 2500\rangle$
AB	$\neg\Box_{[0,.5]} h$	100	2500	100	100	$\langle 0, 0, 0\rangle$	$\langle 0, 0, 0\rangle$	$\langle 20, 26.53, 480\rangle$
AB	$\neg\Box_{[2,2.5]} i$	100	2500	100	99	$\langle 0, 0, 0\rangle$	$\langle 0, 0, 1.0\rangle$	$\langle 20, 155.01, 1620\rangle$
NB	$(\neg\ b)\ \mathcal{U}_{[0,25.0]}\ c$	35	1000	12	11	$\langle 0, .02, .04\rangle$	$\langle 0, .02, .04\rangle$	$\langle 60, 817.14, 1000\rangle$

VI. CONCLUSIONS AND RELATED WORK

In this paper, we have presented an application of the Extended Ant Colony Optimization (EACO) algorithm to the MTL falsification problem. We expect the results of this paper to be valuable to test engineers of complex control and mixed-signal systems. Even though the results are preliminary the performance of EACO/MTL looks very promising. Especially so, since no additional heuristics have been utilized. Among future work, we will focus on improving the performance of EACO/MTL.

This work spans many research areas and due to the space limitations an exhaustive literature review and comparison is not possible. Beyond the references in the introduction, [10] contains a fairly inclusive literature review on the falsification problem of hybrid systems. Our previous work [17] contains references to some discrete and continuous versions of ACO.

Here, we briefly mention some related work in the ACO literature. In [22], the authors keep track of a number of previous candidate solutions to dynamically generate Probabilistic Density Functions (PDF) based on Gaussian functions. These PDFs represent the pheromone distribution over the solution space. In [23], the authors present Fuzzy ACO for optimal control. In Fuzzy ACO, the continuous optimization variables are also split into a number of regions. However, in [23], the authors use fuzzy functions in order to parameterize the search space while we use splines.

REFERENCES

[1] C. Tomlin, J. Lygeros, and S. Sastry, “A game theoretic approach to controller design for hybrid systems,” *Proceedings of the IEEE*, vol. 88, pp. 949–969, 2000.

[2] M. Senesky, G. Eirea, and T.-J. Koo, “Hybrid modelling and control of power electronics,” in *Hybrid Systems: Computation and Control*, ser. LNCS, vol. 2623, 2003, pp. 450–465.

[3] A. A. Julius, A. Halasz, V. Kumar, and G. J. Pappas, “Controlling biological systems: The lactose regulation system of escherichia coli,” in *American Control Conference*, New York, July 11-13 2007.

[4] P. Tabuada, *Verification and Control of Hybrid Systems: A Symbolic Approach*. Springer, 2009.

[5] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine, “The algorithmic analysis of hybrid systems,” *Theoretical Computer Science*, vol. 138, no. 1, pp. 3–34, 1995.

[6] A. Bhatia and E. Frazzoli, “Incremental search methods for reachability analysis of continuous and hybrid systems,” in *Hybrid Systems: Computation and Control*, ser. LNCS, vol. 2993. Springer, 2004, pp. 142–156.

[7] M. Branicky, M. Curtiss, J. Levine, and S. Morgan, “Sampling-based planning, control and verification of hybrid systems,” *IEE Proc.-Control Theory Appl.*, vol. 153, no. 5, pp. 575–590, 2006.

[8] T. Nahhal and T. Dang, “Test coverage for continuous and hybrid systems,” in *CAV*, ser. LNCS, vol. 4590. Springer, 2007, pp. 449–462.

[9] T. Dang, A. Donze, O. Maler, and N. Shalev, “Sensitive state-space exploration,” in *Proc. of the 47th IEEE Conference on Decision and Control*, Dec. 2008, pp. 4049–4054.

[10] T. Nghiem, S. Sankaranarayanan, G. E. Fainekos, F. Ivancic, A. Gupta, and G. J. Pappas, “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*, 2010, pp. 211–220.

[11] P. Zuliani, A. Platzer, and E. M. Clarke, “Bayesian statistical model checking with application to simulink/stateflow verification,” in *Proceedings of the 13th ACM International Conference on Hybrid Systems: Computation and Control*, 2010, pp. 243–252.

[12] E. Plaku, L. E. Kavraki, and M. Y. Vardi, “Falsification of ltl safety properties in hybrid systems,” in *Proc. of the Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, ser. LNCS, vol. 5505, 2009, pp. 368 – 382.

[13] S. Little, D. Walter, K. Jones, and C. J. Myers, “Analog/mixed-signal circuit verification using models generated from simulation traces,” in *Proceedings of the 5th International Symposium on Automated Technology for Verification and Analysis (ATVA)*, ser. LNCS, vol. 4762. Springer, 2007, pp. 114–128.

[14] A. Rizk, G. Batt, F. Fages, and S. Soliman, “On a continuous degree of satisfaction of temporal logic formulae with applications to systems biology,” in *International Conference on Computational Methods in Systems Biology*, ser. LNCS, no. 5307. Springer, 2008, pp. 251–268.

[15] R. Koymans, “Specifying real-time properties with metric temporal logic.” *Real-Time Systems*, vol. 2, no. 4, pp. 255–299, 1990.

[16] G. E. Fainekos and G. J. Pappas, “Robustness of temporal logic specifications for continuous-time signals,” *Theoretical Computer Science*, vol. 410, no. 42, pp. 4262–4291, 2009.

[17] G. E. Fainekos and K. C. Giannakoglou, “Inverse design of airfoils based on a novel formulation of the ant colony optimization method,” *Inverse Problems in Engineering*, vol. 11, no. 1, pp. 21–38, 2003.

[18] M. Dorigo and T. Sttzle, *Ant Colony Optimization*. MIT Press, 2004.

[19] M. Egerstedt and C. Martin, *Control Theoretic Splines: Optimal Control, Statistics, and Path Planning*. Princeton University Press, 2009.

[20] R. G. Sanfelice and A. R. Teel, “Dynamical properties of hybrid systems simulators,” *Automatica*, vol. 46, no. 2, pp. 239–248, 2010.

[21] T. Dang, A. Donzé, and O. Maler, “Verification of analog and mixed-signal circuits using hybrid system techniques,” in *5th International Conference on Formal Methods in Computer-Aided Design*, ser. LNCS, vol. 3312. Springer, 2004, pp. 21–36.

[22] K. Socha and M. Dorigo, “Ant colony optimization for continuous domains,” *European Journal of Operational Research*, vol. 185, pp. 1155–1173, 2008.

[23] J. Van Ast, R. Babuška, and B. De Schutter, “Fuzzy ant colony optimization for optimal control,” in *Proceedings of the 2009 conference on American Control Conference*. IEEE Press, 2009, pp. 1003–1008.