TeLEx: Passive STL Learning Using Only Positive Examples

Susmit Jha[1], Ashish Tiwari[1], Sanjit A. Seshia[2],
Tuhin Sahai[3], and Natarajan Shankar[1]

[1] CSL, SRI International
jha,tiwari,shankar@csl.sri.com
[2] EECS, UC Berkeley
sseshia@eecs.berkeley.edu
[3] United Technologies Research Center
tuhin.sahai@utrc.utc.com

Abstract. We propose a novel passive learning approach, `TeLEx` , to infer signal temporal logic formulas that characterize the behavior of a dynamical system using only observed signal traces of the system. The approach requires two inputs: a set of observed traces and a template Signal Temporal Logic (STL) formula. The unknown parameters in the template can include time-bounds of the temporal operators, as well as the thresholds in the inequality predicates. `TeLEx` finds the value of the unknown parameters such that the synthesized STL property is satisfied by all the provided traces and it is *tight*. This requirement of *tightness* is essential to generating interesting properties when only positive examples are provided and there is no option to actively query the dynamical system to discover the boundaries of legal behavior. We propose a novel quantitative semantics for satisfaction of STL properties which enables `TeLEx` to learn tight STL properties without multidimensional optimization. The proposed new metric is also smooth. This is critical to enable use of gradient-based numerical optimization engines and it produces a 30X-100X speed-up with respect to the state-of-art gradient-free optimization. The approach is implemented in a publicly available tool.

1 Introduction

Signal Temporal Logic (STL) [26] is a discrete linear time temporal logic used to reason about the future evolution of a continuous time behaviour. Generally, this formalism is useful in describing the behaviours of trajectories of differential equations or hybrid models. Several approaches [30, 31, 20, 21, 14, 25] have been recently proposed to automatically design systems and controllers to satisfy given temporal logic specifications. But practical systems are still often created as an assembly of components - some of which are manually designed. Further, many practical systems also include the physical plant, and the overall property of such systems are not known a-priori. Consequently, specification mining has emerged as an effective approach to create abstractions of monitored behavior to better understand complex systems, particularly in autonomy and robotics.

Existing approaches to learning STL properties fall into two categories. The approaches in the first category are classifier-learning techniques which rely on the presence of both positive and negative examples to learn STL formula as a classifier. The approaches in the second category are active-learning approaches that require the capability to experiment with the system to actively try falsifying candidate STL properties in order to obtain counterexamples. In this paper, we address the problem of learning STL properties where negative examples are not provided and it is not possible to actively experiment with the system in a safe manner. For example, learning properties of a vehicle-deployed autonomous driving system must rely on only positive examples. We neither have easy access to negative example trajectories that the system will never execute nor do have an easy way to design safe experiments for falsifying properties.

We propose a novel technique, `TeLEx` that addresses this challenge of data-driven learning of STL formulae from just positive example trajectories. An initial learning bias is provided to `TeLEx` as a template formula. `TeLEx` is restricted to learning parameters of the provided template STL formula and not its structure. `TeLEx` does not have access to either negative examples or the model of the system for falsification. Thus, the boundaries of legal behaviour are not directly available. It has to be inferred just from positive examples. The challenge is to avoid over-generalization in absence of negative examples or counterexamples obtained from active falsification. `TeLEx` addresses this research gap of mining temporal specifications of systems where active experimentation is not possible and failing traces (negative examples) are not available.

`TeLEx` uses a novel quantitative metric that measures the tightness of satisfiability of STL formulas over the traces. This metric uses smooth functions to represent predicates and temporal operators. This keeps the metric differentiable, which would not be possible by just taking the absolute value of standard robustness-metric or directly using the qualitative metric. While sigmoid and exponential-like functions are often used in fields such as deep-learning which rely on numerical-optimization, `TeLEx` is the first to use these to *smoothly* represent tight-satisfiability of STL formulas. The smoothness of the proposed metric allows the effective use of gradient-based numerical optimization techniques. `TeLEx` can be used with a number of different numerical optimization back-ends to synthesize parameters that minimize the new metric over positive examples, and thus, learn a tight STL formula consistent with all the traces.

2 Preliminaries

We present some preliminary concepts and definitions used in our work.

Definition 1. *An interval I is a convex subset of $\mathbb{R}$. A singular interval $[a, a]$ contains exactly one point and $\emptyset$ denotes empty interval. Let $I = [a, b]$, $I_1 = [a_1, b_1]$, and $I_2 = [a_2, b_2]$ be three closed intervals. Then,*
1. $-I = [-b, -a]$ 2. $c+I = [c+a, c+b]$ 3. $I_1 \oplus I_2 = [a_1+a_2, b_1+b_2]$
4. $\min(I_1, I_2) = [\min(a_1, a_2), \min(b_1, b_2)]$
5. $I_1 \cap I_2 = [\max(a_1, a_2), \min(b_1, b_2)]$ *if* $\max(a_1, a_2) \leq \min(b_1, b_2)$ *and* $\emptyset$ *o.w.*

These definitions for various operations are naturally extended to closed, open-closed, and closed-open intervals.

Definition 2. *A time domain ST is a finite or infinite set of time instants such that* $ST \subseteq \mathbb{R}^{\geq 0}$ *with* $0 \in ST$*. A signal or signal-trace* τ *is a function from ST to a domain* $\mathcal{X} \subseteq \mathbb{R}$*. We assume the domain of all signals to be* $\mathbb{R}$ *to simplify notation. We also refer to signal-trace as simply trace or trajectory.*

Monitors used in cyberphysical systems, as well as simulation frameworks, typically provide signal values at discrete time instants due to discrete sampling, or due to limitations of numerical integration techniques. The actual signal can be reconstructed from discrete-time samples using some form of interpolation. In this paper, we assume constant interpolation to reconstruct the signal $\tau(t)$, that is, given a sequence of time-value pairs $(t_0, x_0), \ldots, (t_n, x_n)$, for all $t \in [t_0, t_n)$, we define $\tau(t) = x_i$ if $t \in [t_i, t_{i+1})$, and $\tau(t_n) = x_n$. The signal temporal logic (STL) formula are used to describe properties of signals. The syntax of STL is given as follows:

Definition 3. *A formula* $\phi \in \mathcal{F}$ *of bounded-time STL is defined as follows:*

$$\phi := \bot \mid \top \mid \mu \mid \neg\phi \mid \phi \vee \phi \mid \phi \wedge \phi \mid \phi \mathbf{U}_{[t_1,t_2]} \phi \mid \mathbf{F}_{[t_1,t_2]} \phi \mid \mathbf{G}_{[t_1,t_2]} \phi$$

where $0 \leq t_1 < t_2 < \infty$ *and the atomic predicates* $\mu : \mathbb{R}^n \to \{\top, \bot\}$ *are inequalities on a set X of n signals, that is,* $\mu(X)$ *is of the form* $g(X) \geq \alpha$*, where* $\alpha \in \mathbb{R}$ *and* $g : \mathbb{R}^n \to \mathbb{R}$ *is a continuous function.*

The eventually $\mathbf{F}$ and globally $\mathbf{G}$ operators are shorthands for $\top \mathbf{U}_{[t_1,t_2]} \phi$ and $\neg(\top \mathbf{U}_{[t_1,t_2]} \neg\phi)$ respectively. We keep them, nonetheless, to aid clarity when presenting the different ways of assigning semantics to these operators. We refer to [26, 10], and the survey in [27], for detailed discussion on STL. We briefly summarize its qualitative semantics in Definition 4. Let $\mathcal{T}$ denote the set of all signal-traces.

Definition 4. *The qualitative semantics of STL formulas is given by the function* $\psi : \mathcal{F} \times \mathcal{T} \times ST \to \texttt{Bool}$ *that maps an STL formula* ϕ*, a given signal-trace* $\tau \in \mathcal{T}$*, and a time* $t \in ST$ *to a Boolean value (True* $\top$*, or False* $\bot$*) such that*

- $\psi(\top, \tau, t) = \top$
- $\psi(\mu, \tau, t) = \mu(\tau(t))$
- $\psi(\neg\phi, \tau, t) = \neg\psi(\phi, \tau, t)$
- $\psi(\phi_1 \vee \phi_2, \tau, t) = \psi(\phi_1, \tau, t) \vee \psi(\phi_2, \tau, t))$
- $\psi(\phi_1 \wedge \phi_2, \tau, t) = \psi(\phi_1, \tau, t) \wedge \psi(\phi_2, \tau, t))$
- $\psi(\mathbf{F}_{[t_1,t_2]}\phi, \tau, t) = \exists t' \in [t + t_1, t + t_2]\ \psi(\phi, \tau, t')$
- $\psi(\mathbf{G}_{[t_1,t_2]}\phi, \tau, t) = \forall t' \in [t + t_1, t + t_2]\ \psi(\phi, \tau, t')$
- $\psi(\phi_1 \mathbf{U}_{[t_1,t_2]} \phi_2, \tau, t) = \exists t' \in [t+t_1, t+t_2]\ (\psi(\phi_2, \tau, t') \wedge \forall t'' \in [t, t')\ \psi(\phi_1, \tau, t''))$

Motivated by the need to define how robustly a trace satisfies a formula, formulas in STL were given a quantitative semantics, where formulas are interpreted over numbers such that positive numbers indicate that the formula is True, and negative numbers indicate falsehood. We summarize the quantitative semantics (robustness metric) from [13, 11] below.

Definition 5. *The robustness metric ρ maps an STL formula $\phi \in \mathcal{F}$, a signal trace $\tau \in \mathcal{T}$, and a time $t \in ST$ to a real value, that is, $\rho : \mathcal{F} \times \mathcal{T} \times ST \rightarrow \mathbb{R} \cup \{\infty, -\infty\}$ such that:*

- $\rho(\top, \tau, t) = +\infty$
- $\rho(\mu, \tau, t) = g(\tau(t)) - \alpha$ *where* $\mu(X)$ *is* $g(X) \geq \alpha$
- $\rho(\neg\phi, \tau, t) = -\rho(\phi, \tau, t)$
- $\rho(\phi_1 \vee \phi_2, \tau, t) = \max(\rho(\phi_1, \tau, t), \rho(\phi_2, \tau, t))$
- $\rho(\mathbf{F}_{[t_1,t_2]}\phi, \tau, t) = \sup_{t' \in [t+t_1, t+t_2]} \rho(\phi, \tau, t')$
- $\rho(\mathbf{G}_{[t_1,t_2]}\phi, \tau, t) = \inf_{t' \in [t+t_1, t+t_2]} \rho(\phi, \tau, t')$
- $\rho(\phi_1 \mathbf{U}_{[t_1,t_2]} \phi_2, \tau, t) = \sup_{t' \in [t+t_1, t+t_2]} (\min(\rho(\phi_2, \tau, t'), \inf_{t'' \in [t, t')} \rho(\phi_1, \tau, t'')))$

A STL formula ϕ is satisfied by a trace τ at time t, that is, $\psi(\phi, \tau, t) = \top$ if and only if $\rho(\phi, \tau, t) \geq 0$. Intuitively, ρ quantifies the *degree of satisfiability*. This has motivated its use in learning STL formulae for specification mining [11, 23, 7, 18], diagnosis [24], falsification [6, 1, 2], and system synthesis [9, 4, 31].

3 Related Work

In this section, we summarize related work on learning STL formulae and contrast them to the approach presented in this paper. We categorize related work into three groups: learning STL formula, quantitative metrics for temporal logic and learning concepts from positive examples.

Learning STL formula: Existing techniques for learning STL formulae can be broadly classified into active and passive methods. Active STL learning methods rely on availability of a simulation model on which candidate temporal properties can be falsified [6, 1, 33, 3]. This generates counterexamples. Since these models are often complex executable models, black-box optimization techniques such as simulated annealing are used in falsification of candidate temporal logic properties. If the falsification succeeds, the incorrect parameter values are eliminated and the obtained negative example is used in the next iteration of inferring new candidate parameters values of the temporal logic property. We address a different problem of learning signal temporal logic formula when the simulation model is not available. Further, instead of using gradient-free optimization methods such as simulated annealing, Monte Carlo and ant colony optimization to falsify models, we use more scalable gradient-based numerical optimization methods to infer tightest STL property consistent with a given set of traces. Gradient-based methods for falsification [2] have also been proposed recently to exploit the differentiable nature of simulation models but our approach does not have access to a simulation model. Instead, we define a *smooth* tightness metric for satisfiability of STL properties, and use gradient-based methods to search over the parameter space of STL formulae.

Passive data driven approaches for learning STL formula from positive and negative example traces have also been proposed in literature. Learning STL formula is reduced to a two class supervised classification problem [24, 7, 15] that is solved using a mixture of discrete and continuous optimization using decisions trees and simulated annealing. A model based approach that relies on statistical induction of models before learning STL formulae is presented in [7]. In contrast, TeLEx addresses the problem of passive learning of STL formulae in presence of only positive examples.

Metrics for STL Satisfiability: Signal temporal logic was introduced [26, 11] within the context of monitoring temporal properties of signals. It is possible to quantify the degree of satisfiability of an STL property on a signal trace, thus going beyond the Boolean interpretation. Robustness metric was proposed [13, 11] to provide such a quantitative metric, as described in Section 2. Intuitively, this metric captures the closest distance between the signal trace and the boundary of set of signals satisfying the STL property. This is the worst-case measure of degree of satisfiability. More recently, an *average robustness metric* has also been proposed [25] in the context of task and motion planning application where the `min` (`inf`) operator in the metric definition for globally properties is replaced by an averaging operator. This allows more efficient encoding to linear programs for certain planning problems. These metrics are monotonic, that is, the measure is higher for formulas that are more robustly satisfiable.

If we use robustness metric to learn STL properties from a set of positive example traces, then we would learn very weak properties. This is because a weaker STL property would have a higher robustness value for any given set of positive example signal traces. For example, even if $G(x > 0)$ holds for a given set of traces, the formula $G(x > -100)$ holds more robustly, and would be preferred if we optimized for the standard robustness metric. Hence, in this paper, we define a new metric that captures *tight satisfiability* of an STL property over positive example traces.

A possible approach for finding a tight formula would be to seek a formula that minimizes the absolute-value of the robustness-metric. However, this is not ideal because the absolute-value function is non-differentiable at the optimum and hence, optimizing such a metric would be very challenging. Our proposed novel metric uses smooth functions, such as sigmoid and exponentials, to model tight-satisfiability while still retaining differentiability to aid optimization.

Learning from Positive Examples: Learning from positive examples has been investigated extensively in machine learning. Gold et al [16] showed that even learning regular languages from a class with at least one infinite language is not possible with only positive examples in a deterministic setting. Horining [17] considered the case of stochastic context-free grammars and assumed that the positive examples were generated by sampling from the unknown grammar according to the probabilities assigned to the productions. He proved that such positive examples could be used to converge to the correct grammar in the limit with probability one. Angluin [5] generalized these results to identifying any unknown formal language in the limit with probability one as long as

positive examples are drawn according to an associated probability distribution. Apart from the literature on language learning, Muggleton [28] showed that logic programs are learnable with arbitrarily low expected error just from positive examples within a Bayesian framework. Valiant [32] showed monomials and k-CNF formulas are Probably Approximately Correct (PAC) learnable using only positive examples. While learning from positive examples and its limitations have been studied for other concept classes [22], our approach is the first to consider learning STL properties from positive examples.

4 Learning STL from Positive Examples

Before we present the proposed approach for learning STL properties from just positive examples, we present a simple motivating example.

Illustrative Example: Let us consider an autonomous vehicle system where the steering angle `ang` and speed `spd` are being observed. Each element of the observed trace is a tuple of the form (`timestamp`, `ang`, `spd`). We would like to learn an STL property with the template: $\phi = |\texttt{ang}| \geq 0.2 \Rightarrow F_{[0,6]}\texttt{spd} \leq \alpha$, which intuitively means that we would like to learn the minimum speed α reached within 6 seconds of initiating a turn. Let us consider a timestamped signal trace: $\tau = (0, 0.1, 15), (2, 0.2, 14), (4, 0.3, 12), (6, 0.35, 10), (8, 0.4, 8), \ldots$. For this trace, we notice that $(|\texttt{ang}| \geq 0.2 \Rightarrow F_{[0,6]}\texttt{spd} \leq 8)$ would tightly fit the data. But if we used the robustness metric for optimization, increasing the value of α would be preferred since it increases the robustness value. The robustness metric value for the instantiated template ϕ and the trajectory τ is $\rho(\phi, \tau, 0) = 0$ when $\alpha = 8$, $\rho(\phi, \tau, 0) = 2$ when $\alpha = 10$, $\rho(\phi, \tau, 0) = 992$ when $\alpha = 1000$, and so on. A weak property like $|\texttt{ang}| \geq 0.2 \Rightarrow F_{[0,6]}\texttt{spd} \leq 1000$ has higher robustness score than the tight property $|\texttt{ang}| \geq 0.2 \Rightarrow F_{[0,6]}\texttt{spd} \leq 8$ but clearly, the latter is a more fitting description of the observed behavior.

Problem Definition: We next present some definitions essential to formulating the problem of learning STL properties from positive examples.

Definition 6. *A template STL formula $\phi(p_1, p_2, \ldots, p_k)$ with k unknown parameters is a negation-free bounded-time signal temporal logic formula with the syntax in Definition 3 where some of the time bounds of temporal operators and thresholds of atomic predicates are not constants but instead, free parameters. The parameters are optionally associated with interval constraints providing lower and upper bounds; that is, $l_i \leq p_i \leq u_i$ for $1 \leq i \leq k$ where l_i, u_i are constant bounds.*

Note that we assume templates are negation free. If there are no **U** operator in a formula ϕ, then the negation in $\neg\phi$ can be pushed inside a formula until we are only left with negated atomic predictes. Negated predicates can themselves be rewritten in negation-free form.

We say that an STL formula $\phi(v_1, v_2, \ldots, v_k)$ completes the STL template if the values $v_i \in \mathbb{R}$ for parameters p_i satisfy all the bound constraints on p_i.

Definition 7. *Given a temporal logic property $\phi(v_1, v_2, \ldots, v_k)$ that completes a template $\phi(p_1, p_2, \ldots, p_k)$, we define the ϵ-neighborhood of $\phi(v_1, v_2, \ldots, v_k)$ as* $\mathcal{N}_\epsilon(\phi(v_1, v_2, \ldots, v_k)) = \{\phi(v'_1, v'_2, \ldots, v'_k)$ *s.t.* $|v_i - v'_i| \leq \epsilon$ *for* $1 \leq i \leq k\}$.

We now formally define the problem of learning signal temporal logic formula. The second condition in Definition 8 ensures ϵ-tightness while the first condition ensures that the STL formula is consistent with positive examples.

Definition 8. *Given a set of traces $\mathcal{T}$ and template STL $\phi(p_1, p_2, \ldots, p_k)$, the problem of learning ϵ-tight STL formula is to learn the values of the parameters, $p_i = v_i^*$, such that*

- *the STL formula $\phi(v_1^*, v_2^*, \ldots, v_k^*)$ holds over all traces in $\mathcal{T}$, that is, $\forall \tau \in \mathcal{T} : \tau \models \phi(v_1^*, v_2^*, \ldots, v_k^*)$ and*
- *there exists some $\phi(v_1, v_2, \ldots, v_k) \in \mathcal{N}_\epsilon(\phi(v_1^*, v_2^*, \ldots, v_k^*))$ that does not hold over at least one trace in $\mathcal{T}$; that is, $\exists \tau \in \mathcal{T} : \tau \not\models \phi(v_1, v_2, \ldots, v_k)$*

We have used the notation $\tau \models \phi$ here to denote $\psi(\phi, \tau, 0) = \top$, where ψ is the qualitative semantics presented in Definition 4. We can solve the problem of learning ϵ-tight STL formulas by formulating the following constrained multi-objective optimization problem where minimization is done with respect to free parameters $p_1, \ldots, p_k$.

$$\texttt{minimize } \{|\epsilon_1|, |\epsilon_2|, \ldots, |\epsilon_k|\} \text{ s.t.}$$
$$\epsilon_1 = p_1 - p'_1, \epsilon_2 = p_2 - p'_2, \ldots, \epsilon_k = p_k - p'_k$$
$$\forall \tau \in \mathcal{T}\ \tau \models \phi(p_1, p_2, \ldots, p_k),\ \exists \tau' \in \mathcal{T}\ \tau' \not\models \phi(p'_1, p'_2, \ldots, p'_k)$$

We can check if the solution of the above problem solves our ϵ-tight learning problem by checking if $\max\{|\epsilon_1|, \ldots, |\epsilon_k|\}$ is less than the desired ϵ (or, we could alternatively change the above optimization problem to a min-max problem). However, the above optimization problem is difficult to solve in practice for two reason - first, it requires multi-objective optimization where the number of objectives, k, grows with the number of parameters in the signal temporal logic formula. Further, the constraints require checking satisfiability of the bounded-time STL formula over finite traces which is itself an NP hard problem.

The robustness metric for quantitative satisfiability of STL formula allows us to replace satisfiability checking with nonlinear constraints in the above optimization problem.

$$\texttt{minimize } \{|\epsilon_1|, |\epsilon_2|, \ldots, |\epsilon_k|\} \text{ s.t.}$$
$$\epsilon_1 = p_1 - p'_1, \epsilon_2 = p_2 - p'_2, \ldots, \epsilon_k = p_k - p'_k$$
$$\forall \tau \in \mathcal{T}\ \rho(\phi(p_1, p_2, \ldots, p_k), \tau, 0) \geq 0,\ \exists \tau' \in \mathcal{T}\ \rho(\phi(p'_1, p'_2, \ldots, p'_k), \tau', 0) < 0$$

Next, we notice that the robustness metric is continuous in the parameters p_i corresponding to inequality thresholds and time-bounds and hence, one could expect that we will obtain a reasonable solution for the above problem by solving the following simpler scalar optimization problem :

$$\texttt{minimize}_{p_1, p_2, \ldots, p_k} \ \min_{\tau \in \mathcal{T}} |\rho(\phi(p_1, p_2, \ldots, p_k), \tau, 0)|$$

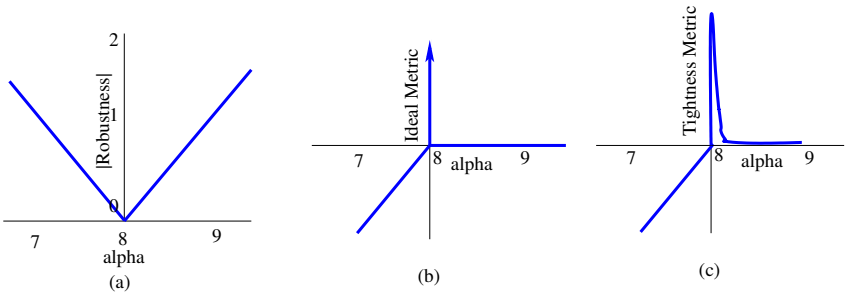


Fig. 1: (a) The absolute value of robustness metric reaches 0 at $\alpha = 8$. It is close to 0 even at 7.99 even though the temporal property corresponding to $\alpha = 7.99$ is violated by the trace. (b) The ideal metric should be negative when $\alpha < 8$ and jump to ∞ when $\alpha = 8$ and drop down to 0 when $\alpha > 8$. (c) A metric which is negative for $\alpha < 8$, reaches its maxima between 8 and $8 + \epsilon$ and then drops to 0.

There are two problems with this approach of solving the tight-STL learning problem using the above optimization problem. This optimization problem uses the absolute value of the robustness metric. This metric is generally not differentiable at $\rho(\phi(p_1, p_2, \ldots, p_k)) = 0$. Further, if we get an ϵ-approximate solution for the above optimization problem, it no longer guarantees that all traces will satisfy the instantiated template ϕ. This is because the absolute value can be a small positive number even when the actual value is a small negative number. In Figure 1, we use the example at the beginning of the section to illustrate the problem. Figure 1(b) illustrates an ideal metric, because it achieves its maximum at the the boundary of satisfiability and unsatisfiability. Maximizing this metric would yield tight STL property but optimizing such a discontinuous function is difficult. Figure 1(c) illustrates a more practical incarnation of the ideal metric, which is not discontinuous but still useful to learn ϵ tight STL property. Our main contribution is designing such a metric.

We begin by first defining a tightness metric for predicates. We would like the metric to achieve its maximum value at the boundary in order to discover tight STL properties. For a predicate $\mu(\mathbf{x}) := g(\mathbf{x}) \geq \alpha$, recall that the robustness metric is $\rho(\mu, \tau, t) = g(\tau(t)) - \alpha = r$. We would like to define a tightness metric $\theta(\mu, \tau, t)$ such that it is similar to Figure 1(c), and hence we define it to be

$$\frac{1}{r + e^{-\beta r}} - e^{-r}$$

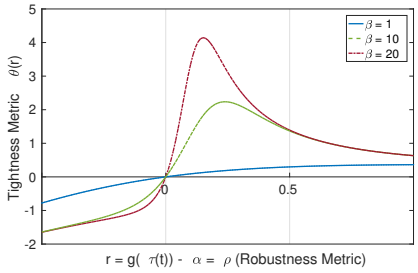
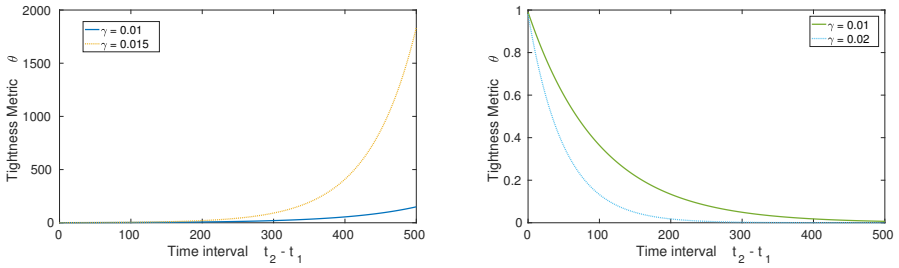


Fig. 2: Tightness metric θ for predicate

where $\beta \geq 1$ is an adjustable parameter. This function is plotted in Figure 2 and it approaches the ideal function in Figure 1(b) as β increases albeit at the cost of numerical stability during optimization. This function is smooth (its derivative is defined and also continuous), and hence, is amenable to gradient-based numerical optimization techniques. Finding an ϵ-tight value of α reduces to maximizing θ with appropriate choice of β - lower values of ϵ require higher values of β.



(a) Globally Operator: $\theta(G_{[t_2-t_1]}\top)$ (b) Eventually Operator: $\theta(F_{[t_2-t_1]}\top)$

Fig. 3: Tightness metric θ

Apart from the predicates, the other difficult cases for defining the tightness metric (θ) happen to be the temporal operators. The requirement here is that the metric θ should be defined such that it prefers longer time intervals for globally operator and shorter for eventually operator as illustrated in Figure 3.

We next formally define the tight quantitative semantics over negation-free STL properties and show how it can be used to formulate the problem of learning consistent and tight STL property as a numerical optimization problem over a single (scalar) cost metric. If the original formula has negation, it is pushed inwards through Boolean combinations, F and G temporal operations, and the inequality in predicate is flipped. Negation can also be pushed inwards through discrete bounded time U operator via case-splitting. Further, since we deal with continuous signals, we consider only non-strict inequalities as predicates and relax strict inequalities if needed.

Definition 9. *The tightness metric $\theta : \mathcal{F} \times \mathcal{T} \times ST \mapsto \mathbb{R} \cup \{-\infty, \infty\}$ maps an STL formula $\phi \in \mathcal{F}$, a trace $\tau \in \mathcal{T}$, and a sampled time instance $t \in ST$ to a real value s.t.:*

- $\theta(\top, \tau, t) = \infty, \quad \theta(\bot, \tau, t) = -\infty$
- $\theta(\mu, \tau, t) = \mathtt{P}(g(\tau(t)) - \alpha)$ *where* $\mu(\mathbf{x}) := (g(\mathbf{x}) \geq \alpha)$
- $\theta(\phi_1 \wedge \phi_2, \tau, t) = \min(\theta(\phi_1, \tau, t), \theta(\phi_2, \tau, t))$
- $\theta(\phi_1 \vee \phi_2, \tau, t) = \max(\theta(\phi_1, \tau, t), \theta(\phi_2, \tau, t))$
- $\theta(\mathbf{F}_{[t_1,t_2]}\phi, \tau, t) = \mathtt{C}(\gamma, t_1, t_2) \sup_{t' \in [t+t_1, t+t_2]} \theta(\phi, \tau, t')$
- $\theta(\mathbf{G}_{[t_1,t_2]}\phi, \tau, t) = \mathtt{E}(\gamma, t_1, t_2) \inf_{t' \in [t+t_1, t+t_2)} \theta(\phi, \tau, t')$
- $\theta(\phi_1 \mathbf{U}_{[t_1,t_2]}\phi_2, \tau, t) = \mathtt{E}(\gamma, t_1, t_2) \sup_{t' \in [t+t_1, t+t_2]} (\min(\theta(\phi_2, \tau, t'), \inf_{t'' \in [t, t')} \theta(\phi_1, \tau, t'')))$

where the peak function $\mathtt{P}(r) = \frac{1}{r+e^{-\beta r}} - e^{-r}$,
the contraction function $\mathtt{C}(\gamma, t_1, t_2) = \frac{2}{1+e^{\gamma(t_2-t_1+1)}}$,
the expansion function $\mathtt{E}(\gamma, t_1, t_2) = \frac{2}{1+e^{-\gamma(t_2-t_1+1)}}$,
$\beta \geq 1$ *is a coefficient chosen to determine sharpness of peak and* $\gamma \geq 0$ *is a coefficient chosen to trade-off tightness in time vs tightness over predicates for a given time-scale and spread of continuous variables. We choose to use the expansion function* $\mathtt{E}$ *in the definition of tightness of* $\mathbf{U}$*-formulae. We could replace* $\mathtt{E}$ *by* $\mathtt{C}$ *if shorter time-intervals are preferred in the* $\mathbf{U}$*-operator.*

If both the time-interval and predicate threshold is unknown for a temporal operator, then there is a choice in either tightening time-intervals and discovering predicates that hold over these or to find tighter predicates over longer (in case of eventually) and shorter (in case of globally) operators. Increasing γ would result in tighter time-intervals. Increasing β would result in tighter predicates. In the following theorem, we summarize the relation between the tightness metric and satisfaction of STL formula.

Theorem 1. *The tightness metric for a given STL formula* ϕ*, namely* $\theta(\phi, \tau, t)$ *is nonnegative if and only if* τ *satisfies* ϕ *at time* t*.*

Proof. We first show that $\theta(\phi, \tau, t) \geq 0$ if and only if $\rho(\phi, \tau, t) \geq 0$ using structural induction. We have only two nontrivial cases:
- Atomic Predicates: We know that $\frac{1}{r+e^{-\beta r}} - e^{-r} \geq 0$ where $\beta \geq 1$ if and only if $r \geq 0$. Hence, $\theta(\mu, \tau, t) = \frac{1}{r+e^{-\beta r}} - e^{-r} \geq 0$ if and only if $r = g(\tau(t)) - \alpha = \rho(\mu, \tau, t) \geq 0$
- Temporal Operators: $\mathtt{C}(\gamma, t_1, t_2) = \frac{2}{1+e^{\gamma(t_2-t_1+1)}} \geq 0$ for all $t_2 > t_1$ and $\mathtt{E}(\gamma, t_1, t_2) = \frac{2}{1+e^{-\gamma(t_2-t_1+1)}} \geq 0$ for all $t_2 > t_1$. Hence, θ has the same sign as ρ, that is, $\theta(\phi, \tau, t) \geq 0$ if and only if $\rho(\phi, \tau, t) \geq 0$.
Thus, $\theta(\phi, \tau, t) \geq 0$ if and only if $\rho(\phi, \tau, t) \geq 0$ and we know that $\rho(\phi, \tau, t) \geq 0$ if and only if τ satisfies ϕ at time t. □

The theorem above shows that a STL formula ϕ that has positive tightness metric (over all the traces τ in some set $\mathcal{T}$) will also evaluate to `True` in all these traces. But we want a formula that is not only consistent with the traces, but also tight on the traces. The following lemma says that optimizing for the tightness metric results in tight formulas.

Lemma 1. *Given a trace* τ *and a template STL formula* $\phi(p_1, p_2, \ldots, p_k)$ *with* k *unknown parameters (Definition 6), let*

$$(v_1^*, v_2^*, \ldots, v_k^*) = \arg \max_{p_1, p_2, \ldots, p_k} \theta(\phi(p_1, p_2, \ldots, p_k), \tau, 0)$$

be a solution $\boldsymbol{v}^* = (v_1^*, \ldots, v_k^*)$ *such that* $\theta(\phi(\boldsymbol{v}^*), \tau, 0)$ *is a finite nonnegative value. Then* $\boldsymbol{v}^*$ *is a solution for the* ϵ*-tight STL learning problem on the singleton set* $\{\tau\}$ *of traces for any value of* ϵ *such that* $\epsilon > \eta$*, where* η *is no more than the robustness* $\rho(\phi(\boldsymbol{v}^*), \tau, 0)$ *of the discovered instantated formula. The value* η *can be made arbitrarily small with appropriate choice of* β, γ*.*

Proof. (Sketch) We again argue by structural induction over the template ϕ. Since ϕ is negation-free, we have three cases. (Case 1) If the top symbol of ϕ is a temporal operator with a time bound $[t_1, t_2]$ such that either t_1 or t_2 is a parameter, then our definition of θ guarantees that the interval $[t_1^*, t_2^*]$ (in the instantiated solution) is maximally elongated or contracted, and hence $\phi(\boldsymbol{v}^*)$ can be falsified by an ϵ perturbatation to the interval, for any $\epsilon > 0$. (Case 2) If ϕ is an atomic predicate, then the robustness measure ρ clearly defines the minimum perturbation required to falsify it. (Case 3) If the top symbol of ϕ is $\vee$ or $\wedge$, we can reason inductively one or both of the subformulas.

For the second part, note that we can decrease η by choosing a large β and $\gamma > 0$. (Case 1) The value of r at which the function $\frac{1}{r+e^{-\beta r}} - e^{-r}$ peaks monotonically decreases with β and hence, more tight predicates (smaller r) can be learnt by increasing β. Hence, η decreases by increasing β. (Case 2) From the definition of $\mathtt{C}$, we observe that the function $\frac{2}{1+e^{\gamma(\Delta t+1)}}$ decreases monotonically with γ and the function $\frac{2}{1+e^{-\gamma(\Delta t+1)}}$ increases monotonically with γ. Thus, if $\gamma > 0$, these functions cause us to learn the largest or smallest possible time interval, and hence changing the learnt intervals even slightly falsifies the formula. Hence, if $\gamma > 0$, then $\eta = 0$ for formulas that have a parametric temporal operator at the top. □

We can lift Lemma 1 to a set of traces, but we lose the ability to arbitrarily decrease η.

Theorem 2. *Given a set of traces $\mathcal{T}$ and a template STL formula $\phi(p_1, p_2, \ldots, p_k)$, let*

$$(v_1^*, v_2^*, \ldots, v_k^*) = \arg \max_{p_1, p_2, \ldots, p_k} [\min_{\tau \in \mathcal{T}} \theta(\phi(p_1, p_2, \ldots, p_k), \tau, 0)]$$

define the solution $\boldsymbol{v}^* = (v_1^*, \ldots, v_k^*)$ such that $\min_{\tau \in \mathcal{T}} \theta(\phi(\boldsymbol{v}^*), \tau, 0)$ is nonnegative. Then the learnt formula $\phi(\boldsymbol{v}^*)$ solves the ϵ-tight STL learning problem for a value of ϵ such that $\epsilon > \eta$, where $\eta = \min_{\tau \in \mathcal{T}} \rho(\phi(v_1^*, \ldots, v_k^*), \tau, 0)$ is the standard robustness measure of the discovered instantated formula. The value η gets no larger by increasing β and γ.

We use an off-the-shelf solver - quasi-Newton algorithm [12, 34] to solve the above optimization problem. It uses gradient during optimization where the search direction in each iteration i is computed as $d_i = -H_i g_i$. H_i is the inverse of the Hessian matrix and g_i is the current derivative. The Hessian is a matrix of second-order partial derivatives of the cost function and describes its local curvature. Due to the smoothness of the defined tightness metric θ, gradient-based optimization techniques are very effective in solving the STL learning problem since both the gradient and the Hessian can be conveniently computed. We also used the gradient-free optimization to experimentally validate the advantage of smoothness of tightness metric. The optimization engine behind gradient-free optimization is differential evolution [29].

5 Experimental Evaluation

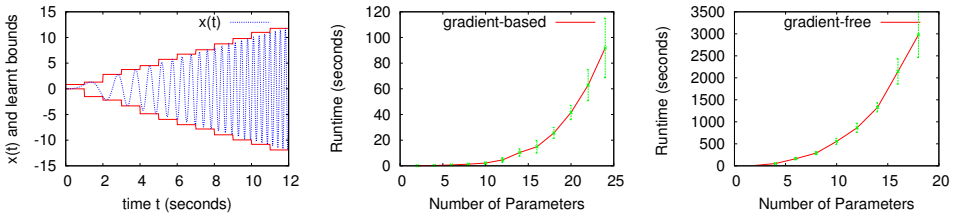
The presented approach is implemented in a publicly available tool: TeLEx[4]. We evaluated the effectiveness of `TeLEx` on a number of synthetic and real case-studies. All experiments were conducted on a quad core Intel Core i5-2450M CPU @ 2.50GHz with 3MB cache per core and 4 GB RAM.

1. Temporal Bounds on Signal $x(t) = t\sin(t^2)$

This case-study was designed to evaluate the scalability of `TeLEx` as well as the tightness of learnt STL formulae using a synthetic trajectory for which we already know the correct answer. We also compare gradient-based `TeLEx` with gradient-free optimization to demonstrate the utility of smoothness of proposed tightness metric. We consider the signal $x(t) = t\sin(t^2)$. We consider 12 STL templates of the form:

$$template(k) \equiv \bigwedge_{i=0}^{k} (G_{[i,i+1]}(x \leq p_{2i} \wedge x \geq p_{2i+1}))$$

where $k = 0, 1, \ldots, 11$. Thus, the number of parameters in these templates grow from 2 to 24. We repeated learning experiments 10 times in each case since numerical optimization routines are not deterministic.



(a) $x(t)$ and learned bounds (b) Gradient-based runtime (c) Gradient-free runtime

Fig. 4: Tightness and Scalability of TeLEx Using Gradient Based Optimization

Figure 4(a) shows the signal trace from time $t = 0$ to $t = 12$ along with the bounds discovered by `TeLEx` while synthesizing the STL property using template *template*(12) (the largest template) and gradient-based optimization. The tightness of bounds demonstrates that the learnt STL properties are tight (and have very low variance) even with 24 parameters. The robustness values for learnt STL properties were always very small (between 0.02 and 0.12). We observed that gradient-free differential evolution also discovered tight properties in all cases (robustness value between (0.06 and 0.35) in which it terminated. Figure 4(b) and (c) show the runtime of gradient-based and gradient-free opti-

[4] https://github.com/susmitjha/TeLEX

mization techniques respectively. Gradient-free methods did not terminate in an hour for more than 18 parameters. We plot the mean runtime (along with standard deviation) from 10 runs with respect to the number of parameters being learnt for each of the 12 templates. The variability in runtime (standard deviation plotted as error bars) increases with the number of parameters. We observe a speed-up of 30X-100X using gradient-based approach due to the smoothness of tightness metric (scales of y-axis in Figure 4(b) and (c) are different).

2. Two Agent Surveillance

We consider a two agent surveillance system in which both agents monitor a 10x10 grid as illustrated in Figure 5. Intruders can pop up at any of the 8 locations marked by circles. But at any point, there are at most two intruders. The two agents are initially at $0, 0$ and $10, 10$ respectively. The agents follow a simple protocol. At each time-instant, the agents calculate the distance from their current location to the intruders (if any), then they select the intruder closest to them as their target for inspection and move towards it. The target of an agent might change while moving (when second intruder pops up and it is closer to the agent moving towards first). After an intruder location is inspected, it is considered neutralized and the agent stays there until new target emerges. The simulator for this simple surveillance protocol is available at the tool website[5]. We simulated this for 1000 time-steps and then used `TeLEx` to learn STL corresponding to the following two properties.

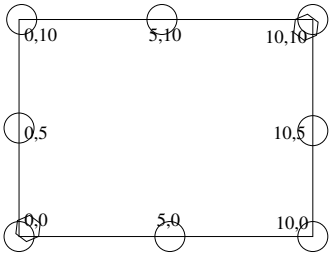


Fig. 5: Two Agent Surveillance

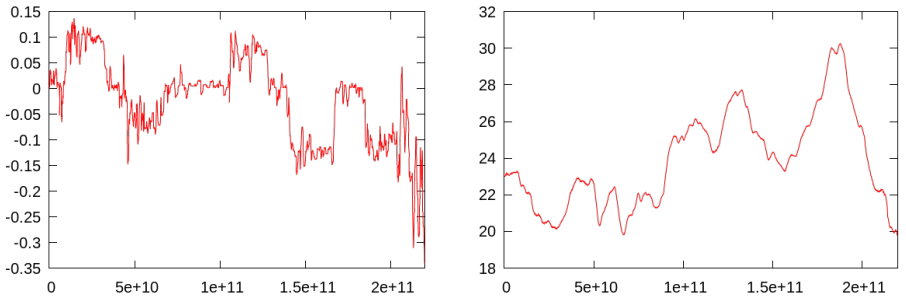
- The maximum time between intruder popping up and being neutralized is 39.001 time-steps.
- The distance between the two agents is at least 4.998. This non-collision between agents is an emergent property due to "move-to-closest" policy of agents and the fact that there are at most two intruders at any given time.

2. Udacity Autonomous-Car Driving Public Data-set

In this case-study, we use the data made available publicly by Udacity as a part of its second challenge for autonomous driving[6]. The data corresponds to an instrumented car (2016 Lincoln MKZ) driving along El Camino Real (a major road in San Francisco Bay Area) starting from the Udacity office in Mountain View and moving north towards San Francisco. We use HMB_1 data-set which is a 221 seconds snippet with a total of over 13205 samples. It has a mixture of turns and straight driving. The data-set includes steering angle, applied torque,

[5] https://github.com/susmitjha/TeLEX/blob/master/tests/twoagent.py
[6] https://github.com/udacity/self-driving-car/tree/master/challenges/challenge-2

speed, throttle, brake, GPS and image. For our purpose, we focus on non-image data. The goal of this data-set is to provide real-world training sample for autonomous driving. Figure 6 shows how the angle and speed vary in the Udacity data-set.



(a) Angle (b) Speed

Fig. 6: Angle and Speed for a subset of Udacity data

We use the tight STL learning approach presented in this paper to learn temporal properties relating angle, torque and speed. Such learned temporal properties could have several utilities. It could be used to examine whether a driving pattern (autonomous or manual) is too conservative or too risky. It could be used to extract sensible logical relations that must hold between different control inputs (say, speed and angle) from good manual driving data, and then enforce these temporal properties on autonomous driving systems. It could also be used to compare different autonomous driving solutions. We are interested in the following set of properties and we present the result of extracting these using `TeLEx` . We would like the robustness metric to be as close to 0 as possible and in all experiments below, we found it to be below 0.005.

1. The speed of the car must be below some upper bound $a \in [15, 25]$ if the angle is larger than 0.2 or below -0.2. Intuitively, this property captures required slowing down of the car when making a significant turn.
   Template STL: $G[0, 2.2e11](((angle \geq 0.2)|(angle \leq -0.2)) \Rightarrow (speed \leq a?15; 25))$
   Synthesized STL: $G[0.0, 2.2e11](((angle \geq 0.2)|(angle \leq -0.2)) \Rightarrow (speed \leq 22.01))$
   Performance: Tightness Metric = 0.067, Robustness Metric = 0.004
   Runtime: 8.64 seconds
2. Similar to the property above, the speed of the car must be low while applying a large torque (say, more than 1.6). Usually, torque is applied to turn along with brake when driving safely to avoid slipping.
   Template STL: $G[0, 2.2e11](((torque \geq 1.6)|(torque \leq -1.6)) \Rightarrow (speed \leq a?15; 25))$
   Synthesized STL: $G[0.0, 2.2e11](((torque \geq 1.6)|(torque \leq -1.6)) \Rightarrow (speed \leq 23.64))$
   Performance: Tightness Metric = 0.221, Robustness Metric = 0.005
   Runtime: 10.12 seconds

3. Another property of interest is to ensure that when the turn angle is high (say, above 0.06), the magnitude of negative torque applied is below a threshold. This avoids unsafe driving behavior of making late sharp compensation torques to avoid wide turns.

Template STL:	$G[0, 2.2e11]((angle \geq 0.06) \Rightarrow (torque \geq b? - 2; -0.5))$
Synthesized STL:	$G[0.0, 2.2e11]((angle \geq 0.06) \Rightarrow (torque \geq -1.06))$
Performance:	Tightness Metric = 0.113, Robustness Metric = 0.003
Runtime:	7.30 seconds

4. Similarly, when the turn angle is low (say, below -0.06), the magnitude of positive torque applied is below a threshold to avoid late sharp compensating torques.

Template STL:	$G[0, 2.2e11]((angle \leq -0.06) \Rightarrow (torque \leq b?0.5; 2))$
Synthesized STL:	$G[0.0, 2.2e11]((angle \leq -0.06) \Rightarrow (torque \leq 1.25))$
Performance:	Tightness Metric = 0.472, Robustness Metric = 0.002
Runtime:	5.00 seconds

5. The torque also must not be so low that the turns are very slow and so, we require that application of negative torque should decrease the angle below a threshold within some fixed time.

Template STL:	$G[0, 2.2e11]((torque \leq 0.0) \Rightarrow F[0.0, 1.2e8](angle \leq a? - 1; 1))$
Synthesized STL:	$G[0.0, 2.2e11]((torque \leq 0.0) \Rightarrow F[0.0, 1.2e8](angle \leq 0.01))$
Performance:	Tightness Metric = 0.727, Robustness Metric = 0.002
Runtime:	46.59 seconds

6 Conclusion

In this paper, we presented a novel approach to learn tight STL formula using only positive examples. Our approach is based on a new tightness metric that uses smooth functions. The problem of learning tight STL properties admits a number of pareto-optimal solutions. We would like to add the capability of specifying preference in which parameters are tightened. Further, computation of the metrics on traces over optimization can be easily parallelized. Another dimension is to study other metrics proposed in literature to quantify conformance and extend tightness over these metrics [8, 19]. In conclusion, `TeLEx` automates the learning of high-level STL properties from observed time-traces given user-guidance in form of templates. It relies on a novel tightness metric defined in this paper which is smooth and amenable to gradient-based numerical optimization techniques.

Acknowledgement
This work is supported in part by DARPA under contract FA8750-16-C-0043 and NSF grant CNS-1423298.

References

1. Abbas, H., Hoxha, B., Fainekos, G., Ueda, K.: Robustness-guided temporal logic testing and verification for stochastic cyber-physical systems. In: Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), 2014 IEEE 4th Annual International Conference on. pp. 1–6. IEEE (2014)
2. Abbas, H., Winn, A., Fainekos, G., Julius, A.A.: Functional gradient descent method for metric temporal logic specifications. In: American Control Conference (ACC), 2014. pp. 2312–2317. IEEE (2014)
3. Akazaki, T.: Falsification of conditional safety properties for cyber-physical systems with gaussian process regression. In: International Conference on Runtime Verification. pp. 439–446. Springer (2016)
4. Aksaray, D., Jones, A., Kong, Z., Schwager, M., Belta, C.: Q-learning for robust satisfaction of signal temporal logic specifications. In: Decision and Control (CDC), 2016 IEEE 55th Conference on. pp. 6565–6570. IEEE (2016)
5. Angluin, D.: Identifying languages from stochastic examples. Tech. rep., YALEU/DCS/RR-614, Yale University. Department of Computer Science (1988)
6. Annpureddy, Y., Liu, C., Fainekos, G., Sankaranarayanan, S.: S-taliro: A tool for temporal logic falsification for hybrid systems. In: International Conference on Tools and Algorithms for the Construction and Analysis of Systems. pp. 254–257. Springer (2011)
7. Bartocci, E., Bortolussi, L., Sanguinetti, G.: Data-driven statistical learning of temporal logic properties. In: International Conference on Formal Modeling and Analysis of Timed Systems. pp. 23–37. Springer (2014)
8. Deshmukh, J.V., Majumdar, R., Prabhu, V.S.: Quantifying conformance using the skorokhod metric. In: International Conference on Computer Aided Verification. pp. 234–250. Springer (2015)
9. Donzé, A.: Breach, a toolbox for verification and parameter synthesis of hybrid systems. In: International Conference on Computer Aided Verification. pp. 167–170. Springer (2010)
10. Donzé, A.: On signal temporal logic. In: International Conference on Runtime Verification. pp. 382–383. Springer (2013)
11. Donzé, A., Maler, O.: Robust satisfaction of temporal logic over real-valued signals. In: International Conference on Formal Modeling and Analysis of Timed Systems. pp. 92–106. Springer (2010)
12. Facchinei, F., Lucidi, S., Palagi, L.: A truncated newton algorithm for large scale box constrained optimization. SIAM Journal on Optimization 12(4), 1100–1125 (2002)
13. Fainekos, G.E., Pappas, G.J.: Robustness of temporal logic specifications. In: Formal Approaches to Software Testing and Runtime Verification, pp. 178–192. Springer (2006)
14. Fu, J., Topcu, U.: Synthesis of joint control and active sensing strategies under temporal logic constraints. IEEE Trans. Automat. Contr. 61(11), 3464–3476 (2016), http://dx.doi.org/10.1109/TAC.2016.2518639
15. Giuseppe, B., Cristian Ioan, V., Francisco, P.A., Hirotoshi, Y., Calin, B.: A Decision Tree Approach to Data Classification using Signal Temporal Logic. In: Hybrid Systems: Computation and Control (HSCC). pp. 1–10. Vienna, Austria (April 2016)
16. Gold, E.M.: Language identification in the limit. Information and control 10(5), 447–474 (1967)

17. Horning, J.J.: A study of grammatical inference. Tech. rep., DTIC Document (1969)
18. Hoxha, B., Dokhanchi, A., Fainekos, G.: Mining parametric temporal logic properties in model based design for cyber-physical systems. arXiv preprint arXiv:1512.07956 (2015)
19. Jakšić, S., Bartocci, E., Grosu, R., Ničković, D.: Quantitative monitoring of stl with edit distance. In: International Conference on Runtime Verification. pp. 201–218. Springer (2016)
20. Jha, S., Raman, V.: Automated synthesis of safe autonomous vehicle control under perception uncertainty. In: Rayadurgam, S., Tkachuk, O. (eds.) NASA Formal Methods: 8th International Symposium, NFM. pp. 117–132 (2016)
21. Jha, S., Raman, V.: On optimal control of stochastic linear hybrid systems. In: Fränzle, M., Markey, N. (eds.) Formal Modeling and Analysis of Timed Systems: 14th International Conference, FORMATS. pp. 69–84. Springer International Publishing (2016), http://dx.doi.org/10.1007/978-3-319-44878-7_5
22. Jha, S., Seshia, S.A.: A theory of formal synthesis via inductive learning. Acta Informatica (Feb 2017), https://doi.org/10.1007/s00236-017-0294-5
23. Jin, X., Donzé, A., Deshmukh, J.V., Seshia, S.A.: Mining requirements from closed-loop control models. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 34(11), 1704–1717 (2015)
24. Kong, Z., Jones, A., Medina Ayala, A., Aydin Gol, E., Belta, C.: Temporal logic inference for classification and prediction from data. In: Proceedings of the 17th international conference on Hybrid systems: computation and control. pp. 273–282. ACM (2014)
25. Lindemann, L., Dimarogonas, D.V.: Robust control for signal temporal logic specifications using average space robustness. arXiv preprint arXiv:1607.07019 (2016)
26. Maler, O., Nickovic, D.: Monitoring temporal properties of continuous signals. In: Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems, pp. 152–166. Springer (2004)
27. Maler, O., Nickovic, D., Pnueli, A.: Checking temporal properties of discrete, timed and continuous behaviors. In: Pillars of computer science, pp. 475–505. Springer (2008)
28. Muggleton, S.: Learning from positive data, pp. 358–376. Springer Berlin Heidelberg, Berlin, Heidelberg (1997)
29. Price, K., Storn, R.M., Lampinen, J.A.: Differential evolution: a practical approach to global optimization. Springer Science & Business Media (2006)
30. Raman, V., Donzé, A., Maasoumy, M., Murray, R.M., Sangiovanni-Vincentelli, A.L., Seshia, S.A.: Model predictive control with signal temporal logic specifications. In: CDC. pp. 81–87 (Dec 2014)
31. Sadraddini, S., Belta, C.: Robust temporal logic model predictive control. In: Communication, Control, and Computing (Allerton), 2015 53rd Annual Allerton Conference on. pp. 772–779. IEEE (2015)
32. Valiant, L.G.: A theory of the learnable. Communications of the ACM 27(11), 1134–1142 (1984)
33. Yang, H., Hoxha, B., Fainekos, G.: Querying parametric temporal logic properties on embedded systems. In: IFIP International Conference on Testing Software and Systems. pp. 136–151. Springer (2012)
34. Zhu, C., Byrd, R.H., Lu, P., Nocedal, J.: Algorithm 778:fortran subroutines for large-scale bound-constrained optimization. ACM Transactions on Mathematical Software (TOMS) 23(4), 550–560 (1997)