Title
A Metric for Linear Temporal Logic

Authors
Romeo, Íñigo Íncer
Lohstroh, Marten
Iannopollo, Antonio
et al.

Publication Date
2018-11-30

Peer reviewed

A Metric for Linear Temporal Logic

Íñigo Íncer Romeo, Marten Lohstroh,
Antonio Iannopollo, Edward A. Lee, and Alberto Sangiovanni-Vincentelli

Department of Electrical Engineering and Computer Sciences
University of California, Berkeley CA 94720, USA
{inigo, marten, antonio, eal, alberto}@berkeley.edu

Abstract. We propose a measure and a metric on the sets of infinite traces generated by a set of atomic propositions. To compute these quantities, we first map properties to subsets of the real numbers and then take the Lebesgue measure of the resulting sets. We analyze how this measure is computed for Linear Temporal Logic (LTL) formulas. An implementation for computing the measure of bounded LTL properties is provided and explained. This implementation leverages SAT model counting and effects independence checks on subexpressions to compute the measure and metric compositionally.

1 Introduction

Linear Temporal Logic (LTL), introduced by Amir Pnueli in [11], allows us to make logical statements over time. This reasoning device is often used to verify whether a system possesses a given temporal property, as well as to synthesize a system with such a property. Applications of temporal logic are vast, ranging from logic design to robotics. In this paper, we consider the following problem: given a temporal logic formula, how large is the set of traces that satisfies it? Given two temporal formulas, are they alike?

Others who have studied this problem [10] have shown that quantifying the difference between two specifications can be used, for instance, to evaluate the performance of temporal logic inference algorithms by comparing made inferences to a ground truth. Our own motivation to investigate this problem is its relevance to the design of Cyber-Physical Systems (CPS). "Late validation" (i.e., discovering errors or design flaws late in a product's development process or past its deployment) is particularly problematic for CPS because it comes at large human and financial cost; design faults may cause accidents, and solving them may require product recalls or cause manufacturing delays. In an effort to add more rigor to the design process for complex CPS, contract-based design [2, 12] provides tools for reasoning formally about the conditions for correctness of element integration and specification abstraction and refinement.

Using contracts, all components in a design clearly state their environment assumptions and the behaviors they guarantee (with behaviors expressed in formal languages), allowing for compositional reasoning, also in an automated fashion. In [6, 7], Iannopollo *et al.* describe techniques based on Counterexample-Guided

Inductive Synthesis (CEGIS) [14] to synthesize designs from libraries of contracts, satisfying a certain LTL specification. Although contracts and their properties help reduce the complexity of the overall synthesis problem, scalability remains elusive, as the CEGIS loop depends on LTL model checking, a PSPACE-complete problem [13]. To tackle complexity, synthesis must be incremental, and we believe an incremental procedure may be within reach soon. For example, Íncer Romeo *et al.* recently introduced means to compute the operation of quotient for assume-guarantee contracts in [8]. Given a specification $\mathcal{C}$ to be implemented, and the specification $\mathcal{C}'$ of a component to be used in the design, the quotient describes the properties that need to be satisfied, in addition to those required by $\mathcal{C}'$, in order to meet $\mathcal{C}$. Therefore, when synthesizing a specification given as an assume-guarantee contract, we can synthesize partial solutions of the design and keep the one whose quotient is the "smallest" since we believe the "size" of a quotient could be a good indication of complexity. But in order to measure the size of the specification, we need a measure of LTL properties. Introducing that measure is the purpose of this work.

In this paper, we propose a measure for sets of infinite traces spanned by a set of atomic propositions. This measure has the semantics of the amount of trace space that a given property represents. While the measure does not require that properties be specified in LTL, we provide an implementation to compute the measure of bounded LTL formulas and the distance between two such formulas. Our implementation checks whether the measure can be computed compositionally; if not, it computes it using SAT model counting [5].

In Section 2 we introduce the measure based on counting arguments, and we provide the algorithms for its computation for bounded LTL. We discuss our implementation and experiments in Section 3. Section 4 extends our measure to handle infinite traces. In Section 4, we also show how the counting arguments discussed in Section 2 are a special case of the definition of the measure given in Section 4. We revisit existing literature in Section 5 and conclude the paper in Section 6.

2 Measuring Properties by Counting Traces

In this section, we build a measure for bounded LTL formulas based on counting considerations. We argue that a measure that could be used to evaluate the "size" of a formula should tell the fraction of trace space represented by the property being measured. Algorithms are presented that compute this measure compositionally. The counting ideas of this section are generalized to infinite traces in Section 4.

Since an LTL formula denotes a property (i.e., a set of traces), our purpose is to introduce a measure for sets of traces. Let ν be our measure and ϕ an LTL formula. At a high level, we wish the measure to have the following characteristics:

– $\nu(\mathbf{True}) = 1$. The maximum value of the measure is 1 and is achieved when *almost* all possible traces satisfy the formula. The meaning of *almost* will be made clear later. It will coincide with the real-analytic notion of *all traces up to a set of measure zero.*

– $\nu(\mathbf{False}) = 0$. The minimum value of the measure is 0 and is achieved when almost no traces satisfy the formula.

– A *priori*, we state no preference for a trace over another, i.e., all traces that satisfy the formula should contribute equally to the value of the measure.
– The measure is computable in reasonable time.

These high-level requirements motivate us to define the measure as *the fraction of the trace space that satisfies the formula*. Suppose that β is a Boolean expression. When interpreted as an LTL formula, β constrains only the first time step of all traces (i.e., $T=0$). If, say, β is a formula over n atomic propositions, and it is true for m possible combinations of the propositions, the measure of the LTL formula β would be $\nu(\beta) = \frac{\textsc{Count}(\beta)}{2^n} = \frac{m}{2^n}$; indeed, the remaining $1 - \frac{m}{2^n}$ of traces begin with a combination of the atomic propositions which does not satisfy β. Note we assume the existence of a function Count which takes a Boolean expression and outputs the number of satisfying assignments for that expression; in other words, Count carries out SAT model-counting.

	$T=0$	$T=1$	$T=2$	$T=3$	$\cdots$
x_0	$x_0[0]$	$x_0[1]$	$x_0[2]$	$x_0[3]$	$\cdots$
x_1	$x_1[0]$	$x_1[1]$	$x_1[2]$	$x_1[3]$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...
x_{n-1}	$x_{n-1}[0]$	$x_{n-1}[1]$	$x_{n-1}[2]$	$x_{n-1}[3]$	$\cdots$

Table 1: If defined over n Boolean atomic propositions, traces are uniquely determined by an assignment of values to all variables of the form $x_i[t]$ for all $i \in \{0,...,n-1\}$ and $t \in \omega$.

Assume that our LTL formulas are defined over a finite set Σ of atomic propositions and that we consider finite traces up to $T=N$ ($N \in \mathbb{N}$). A trace is defined by an assignment of values to all atomic propositions for each time step. As a result, we can interpret an LTL formula as a Boolean expression involving n variables for each time step, as shown in Table 1. As an example, suppose our formulas are defined over the propositions a and b. Then the formula $\phi = a \wedge (b \vee Xa)$ can be expressed as the Boolean formula $a[0] \wedge (b[0] \vee a[1])$. We assume we have access to a routine AssignedVars which accepts an LTL formula and returns a set of all atomic propositions that appear in the formula, together with the time step for which they appear in the formula. Applying this procedure to our example yields $\textsc{AssignedVars}(\phi) = \{(a,0),(a,1),(b,0)\}$. Note that, using this notation, we interpret (a,i) as a different variable from (b,j) whenever $a \neq b$ or $i \neq j$. Let ϕ and ϕ' be LTL formulas, we also define the function $\textsc{VarUnion}(\phi,\phi') = \textsc{AssignedVars}(\phi) \cup \textsc{AssignedVars}(\phi')$, which returns all time-indexed variables referenced by both formulas, and $\textsc{VarInt}(\phi,\phi')$ for the intersection. For any pair of LTL formulas, ϕ and ϕ', $\textsc{AssignedVars}(\phi \wedge \phi') = \textsc{AssignedVars}(\phi \vee \phi') = \textsc{VarUnion}(\phi,\phi')$.

We define the function VarTimes, which accepts an LTL formula and an atomic proposition, and returns the set of time indices for which that atomic proposition appears in the formula. Applying this function to our previous example gives $\textsc{VarTimes}(\phi,a) = \{0,1\}$ and $\textsc{VarTimes}(\phi,b) = \{0\}$.

Finally, we assume there is a bounded version $\textsc{AssignedVars}(\phi,N)$ that contain the same information as the unbounded version of the function, but with all time indices larger than N removed.

2.1 Boolean operators

We explore how the measure behaves with the Boolean operators. Let ϕ and ϕ' be LTL formulas.

1. NOT. The traces that meet $\neg\phi$ are those that do not meet ϕ. Thus,

$$\nu(\neg\phi)=1-\nu(\phi). \tag{1}$$

2. AND. If a trace σ satisfies $\sigma\models\phi\wedge\phi'$, then

$$\sigma\models\phi \quad \text{and} \tag{2}$$
$$\sigma\models\phi'. \tag{3}$$

If ϕ and ϕ' do not share variables, i.e., they do not make assertions over the same atomic proposition at the same time steps, we observe that the requirement that a trace satisfies (2) is independent of the requirement that it satisfies (3). Therefore, the fraction of traces satisfying $\phi\wedge\phi'$ is the product of the fraction of traces satisfying ϕ and the fraction of traces satisfying ϕ'. On the contrary, if ϕ and ϕ' share variables, we invoke $\textsc{Count}$. Thus, the measure of conjunction is given by

$$\nu(\phi\wedge\phi')=\begin{cases}\nu(\phi)\nu(\phi'), & \text{if } \textsc{VarInt}(\phi,\phi')=\emptyset\\ \frac{\textsc{Count}(\phi\wedge\phi')}{2^{|\textsc{VarUnion}(\phi,\phi')|}}, & \text{otherwise}\end{cases}. \tag{4}$$

Calling $\textsc{Count}$ may be prohibitively expensive if ϕ or ϕ' comprise a large number of variables. It may thus be useful to compute bounds for the measure of the composition. We can state the following bounds for the measure of an AND of any two LTL formulas:

$$\nu(\phi)\nu(\phi')\leq\nu(\phi\wedge\phi')\leq\min\{\nu(\phi),\nu(\phi')\}. \tag{5}$$

3. OR. By de Morgan's laws, we can derive the measure of the disjunction from the previous two operations. We have $\nu(\phi\vee\phi')=1-\nu(\neg\phi\wedge\neg\phi')$. Thus, the measure of the disjunction of two formulas is

$$\nu(\phi\vee\phi')=\begin{cases}1-(1-\nu(\phi))(1-\nu(\phi')), & \text{if } \textsc{VarInt}(\phi,\phi')=\emptyset\\ \frac{\textsc{Count}(\phi\vee\phi')}{2^{|\textsc{VarUnion}(\phi,\phi')|}}, & \text{otherwise}\end{cases}. \tag{6}$$

The following bounds apply to an OR composition for all LTL formulas:

$$\begin{aligned}&\max\{\nu(\phi),\nu(\phi')\}\leq\nu(\phi\vee\phi') \quad \text{and}\\ &\nu(\phi\vee\phi')\leq1-(1-\nu(\phi))(1-\nu(\phi')).\end{aligned} \tag{7}$$

Let NotMeasure, AndMeasure and OrMeasure implement, respectively, (1), (4), and (6). We observe that only when ϕ and ϕ' do not involve the same proposition at the same time steps, the measure of the compositions $\phi \wedge \phi'$ and $\phi \vee \phi'$ can be computed exactly, just using the measures of ϕ and of ϕ'. Thus, when computing the measure of an LTL formula, it is advantageous to rewrite the formula to achieve maximum independence.

Example. Let $\Sigma = \{a, b, c\}$. Let $\phi = a \wedge \mathbf{X}b$, $\phi' = b \vee \mathbf{X}a$, and $\phi'' = a \wedge \neg c$. Consider the formula $\psi = \phi \wedge \phi' \wedge \phi''$. Due to considerations of independence, $\nu(\psi) = \nu(\phi')\nu(\phi \wedge \phi'')$ since ϕ' shares no variables with ϕ or with ϕ''. In this case, the routine AndMeasure would call Count on $\phi \wedge \phi''$, which makes statements over 3 variables. Moreover, $\nu(\psi) \neq \nu(\phi)\nu(\phi' \wedge \phi'')$ since ϕ shares variables with $\phi' \wedge \phi''$. Thus, in this case, AndMeasure would call Count on the entire ψ, which contains 5 variables.

2.2 Temporal operators

In principle, the grammar of LTL is the Boolean grammar plus the symbols $\mathbf{X}$ and $\mathbf{U}$. Other temporal operators can be expressed in terms of this grammar. For instance, $\mathbf{F}\phi = \mathbf{True}\mathbf{U}\phi$ and $\mathbf{G} = \neg\mathbf{F}\neg\phi$. If a bounded semantics is used, we can express $\mathbf{U}$ in terms of $\mathbf{X}$: $\phi\mathbf{U}\psi = \bigvee_{j=0}^{N}\left(\bigwedge_{i=0}^{j-1}\mathbf{X}^i\phi\right) \wedge \mathbf{X}^j\psi$ for $N > 0$ and $\phi\mathbf{U}\psi = \psi$ for $N = 0$. Note that the effect of $\mathbf{X}$ on a formula consists in adding 1 to all temporal indices of the atomic propositions that appear in the formulas; this is the case because $\mathbf{X}$ commutes with all Boolean (and temporal) operators. As a result, for any LTL formula, we have $\textsc{AssignedVars}(\mathbf{X}\phi) = \{(x, T+1) | (x, T) \in \textsc{AssignedVars}(\phi)\}$.

We can apply the expressions we obtained for the Boolean formulas to define the relation of the measure and the temporal operators. We will assume the temporal operators are bounded to $T = N$.

1. $\mathbf{X}$. If no temporal bound is enforced, applying a next step operator to a formula results in the same measure since the effect of this operator is to rename all variables without causing collisions. We thus have $\nu(\mathbf{X}\phi) = \nu(\phi)$. On the other hand, if a bound *is* enforced, we must provide an interpretation for the case of atomic propositions leaving the bound. In coherence with [3], for an atomic proposition a and a time bound N, we provide the following semantics for the bounded $\mathbf{X}$ operator:

$$\mathbf{X}^i a = \begin{cases} \mathbf{X}^i_{\mathrm{LTL}} a, & i \leq N \\ \mathbf{False}, & \text{otherwise} \end{cases},$$

where $\mathbf{X}^i_{\mathrm{LTL}}$ is the LTL next operator. For instance, consider the formula $\phi = \mathbf{X}a$ and suppose we impose a time bound $T = 0$. Then no trace can satisfy this formula in our bounded semantics, i.e., the formula is transformed into the empty property. On the other hand, the formula $\phi = \mathbf{X}\neg a$ is transformed into the entire trace space (i.e., all traces satisfy this property).

2. **U**. The expansion of the until operator produces several terms with shared dependencies. However, it can be shown that $\phi \mathbf{U} \psi = \theta_N$, where θ_N is defined recursively as follows:

$$\theta_0 = \psi \quad \text{and} \quad \theta_i = \psi \vee \phi \wedge \mathbf{X}\theta_{i-1}. \tag{8}$$

This expansion factors out the applications of the **X** operator. In order to compute $\nu(\phi \mathbf{U} \psi)$ compositionally, we need two conditions: ϕ and ψ must be independent, and their atomic propositions need to be qualified at most at one time step (e.g., $\psi' = b \vee a \wedge \mathbf{X}a$ does not pass this test because a is qualified by two different time steps). Under the condition of independence, $\nu(\phi \mathbf{U} \psi) = \gamma_N$, where

$$\begin{aligned} &\gamma_0 = \nu(\mathbf{X}^N \psi) \quad \text{and} \\ &\gamma_i = 1 - \big(1 - \nu(\mathbf{X}^{N-i}\psi)\big)\big(1 - \gamma_{i-1} \cdot \nu(\mathbf{X}^{N-i}\phi)\big). \end{aligned} \tag{9}$$

A routine for computing the measure of formulas with the **U** operator is given below. Analogous routines for the remaining temporal operators are special cases of this routine.

```
function UntilMeasure(φ, ψ, N)                                   ▷ U
    if N = 0 then
        return ν(ψ)
    else if VarInt(φ,ψ) = ∅ and ∀a ∈ Σ.
      |VarTimes(φ,a,N)| ≤ 1 and
      |VarTimes(ψ,a,N)| ≤ 1 then
        γ ← ν(X^N ψ)
        for i ← 1,N do
            α ← (1 − ν(X^{N−i} ψ))
            γ ← 1 − α · (1 − γ · ν(X^{N−i} φ))
        end for
        return γ
    else
        φ' ← ⋁_{j=0}^{N} ( ⋀_{i=0}^{j−1} X^i φ ) ∧ X^j ψ
        NumAssertions ← Count(φ')
        NumVars ← |AssignedVars(φ';N)|
        return NumAssertions/2^NumVars
    end if
end function
```

We observe that the given recursion (9) has a fixed point, which allows us to provide the measure for the unbounded case when ϕ and ψ do not share variables:

$$\nu(\phi \mathbf{U} \psi) = \frac{\nu(\psi)}{1 - \nu(\phi)(1 - \nu(\psi))}. \tag{10}$$

3 Practical Evaluation

We have developed a Python implementation[1] of the algorithms provided in Section 2. Our tool parses LTL formulas and measures them; if a pair of properties is given, it computes their respective distance. After turning the input expression into an abstract syntax tree, we traverse the tree (depth-first, post-order) and annotate visited nodes with dependency information. A subsequent pre-order traversal recursively breaks down the computation of the measure into smaller parts, until a bifurcating node (i.e., $\wedge$, $\vee$, or $\mathbf{U}$) of which its subtrees are considered interdependent (see Section 2) and thus cannot be measured separately. In that case, we invoke Count on an expanded expression obtained from the node with interdependent subtrees, which is carried out by `sharpSAT`, a DPLL-style #SAT solver developed by Marc Thurley [16]. The expansion takes care of transforming all temporal operators into Boolean expressions with fresh variables for each time index up to the given time bound.

3.1 Examples

Specifications are usually composed out of idioms or templates that denote well-known classes of properties such as safety, liveness, correlation, precedence, and response. Each of these classes are thought to be semantically distinct, and thus should occupy a different fraction of the trace space; is this somehow reflected in their measure? Our results, plotted in Fig. 1, confirm this intuition. The measures of these idiomatic LTL expressions are spread remarkably evenly.

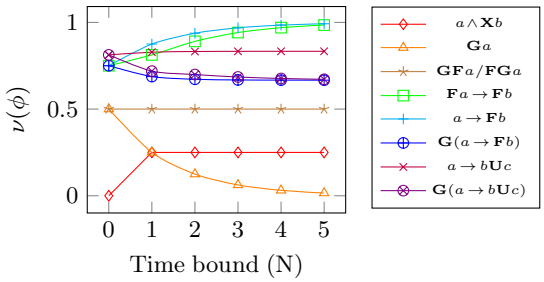

Fig. 1: Measures up to $N = 5$ for ($\Diamond$) initialization, ($\triangle$) safety, ($\star$) liveness/stability, ($\square$) correlation, ($+$, $\oplus$) response, and ($\times$, $\otimes$) precedence.

3.2 Tractability

Because model counting is a #P-complete task [5], unless $P = NP$ there exists no polynomial-time algorithm to implement Count. To make computing our measure more tractable, we employ a divide-and-conquer approach, breaking down the

[1] `https://github.com/icyphy/spec-space`

problem into smaller subproblems, and invoke COUNT only for subexpressions that cannot be broken down further due to interdependencies between variables. To il-

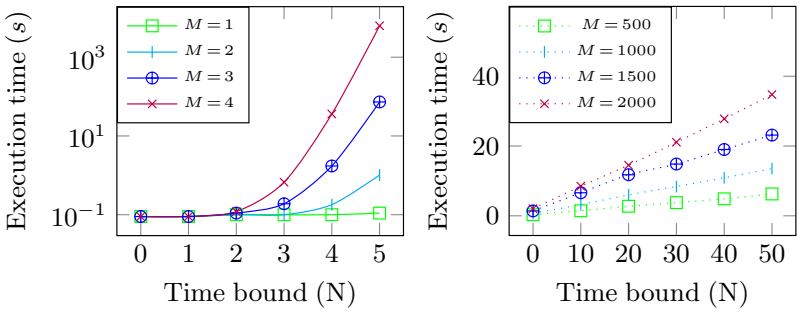

Fig. 2: Computing the measure of $r \to \mathbf{F}(\bigwedge_{i=1}^{M}(x_i \leftrightarrow x_{i+1}))$. (a) Without exploiting independence, the calculation becomes intractable even for small N. (b) Computing this measure compositionally scales linearly with M and N.

lustrate the efficacy of this method, let us consider the formula $r \to \mathbf{F}(\psi)$, where $\psi = \bigwedge_{i=1}^{M}(x_i \leftrightarrow x_{i+1})$. The execution time of computing $\nu(r \to \mathbf{F}(\psi))$ is shown in Figure 2 for various M and N. Because ψ is chain of clauses, each of which shares a variable with the following, all clauses are interdependent. To make matters worse, the expansion of $\mathbf{F}$ yields a disjunction of N time-shifted versions of ψ. Applying COUNT to the resulting Boolean expression is clearly intractable for large N, as the number of clauses grows exponentially with N. Our algorithm, on the other hand, only has to invoke COUNT on expressions with $2M$ clauses that jointly comprise a total of $M+1$ distinct variables, and do so $N+1$ times. Because the expansion of $\mathbf{F}$ yields a disjunction of logically equivalent subexpressions, to calculate the measure of $r \to \mathbf{F}(\psi)$, we invoke COUNT only once and reuse the result.

4 Analytical Viewpoint

In Section 2, we discussed the measure based on counting considerations. In order to count the satisfying assignments for LTL formulas, we imposed a bound $T = N$ on them. In this section we extend the measure to handle *any* set of infinite traces and is, in fact, not limited to those expressible in LTL. First, we consider a map from the set of all traces in a finite number of atomic propositions to the unit interval $I = [0,1]$. With this map in place, we can interpret any set of traces as a subset of I. Thus, we are able to regard properties as subsets of $\mathbb{R}$. We then introduce the property measure as an integral and provide a calculus for computing it.

4.1 Traces as reals

Suppose a is an atomic proposition. A trace σ over a will be given by $\sigma = a[0],a[1],a[2],...$ Consider the map $\|\cdot\|$ on traces given as follows:

$$\|\sigma\| = \sum_{i=0}^{\infty} a[i]2^{-(i+1)}. \tag{11}$$

Syntactically, this corresponds to the binary number "0.a[0]a[1]a[2]...," i.e., $\|\cdot\|$ prepends the string "0." to the trace σ, and removes all commas from it. The inverse operation removes the "0." and adds commas between the binary values of the number. As an example, consider the trace $\sigma = 1,0,1,1,1,0,1,0,0,1,...$ The map gives us $\|\sigma\| = 0.1011101001...$.

Let Σ be the set of atomic propositions. We assume Σ is finite. Then the set of all traces is given by $\mathcal{T} = \left(2^{\Sigma}\right)^{\omega}$, and thus the set of all properties is $\mathcal{P} = 2^{\mathcal{T}}$. We observe that equation (11) is a map $\|\cdot\| : \mathcal{T} \to I$. This map is clearly surjective, but it is not injective. To see this, consider the following cases: suppose we only have one atomic proposition a and suppose we have the trace $\sigma = a,\neg a,\neg a,\neg a,...$ (i.e., the only trace that satisfies the property $\phi = a \wedge \mathbf{XG}\neg a$). Then $\|\sigma\| = 0.1 = 2^{-1}$. Now suppose that $\sigma' = \neg a,a,a,a,...$ (i.e., the only trace that satisfies the property $\phi' = \neg a \wedge \mathbf{XG}a$. Then $\|\sigma'\| = \sum_{i=2}^{\infty} 2^{-i} = 2^{-1}$. So the real numbers 0.0111... and 0.1000... are the same. But we think of traces σ and σ' as being quite different. Indeed, we can build state machines to recognize one and not the other. This shows that $\|\cdot\|$ is not injective.

When Σ contains n atomic propositions instead of 1, traces are given in the form shown in Table 1. We observe that we can *linearize* the trace to

$$x_0[0],x_1[0],...,x_{n-1}[0],x_0[1],...,x_{n-1}[1],x_0[2]...$$

This interpretation allows us to extend $\|\cdot\|$ to a map from sets of infinite traces over any finite set of atomic propositions to the unit interval.

4.2 Properties as subsets of $\mathbb{R}$

Let ϕ be a property over Σ. We define a map $f : \mathcal{P} \to 2^I$ from properties to subsets of the reals given by $f\phi = \{\|\sigma\| : \sigma \in \phi\}$. We observe that f allows us to interpret ϕ as subset of $\mathbb{R}$. We call $f\phi$ the *property intervals* of ϕ.

4.3 A measure for properties

Now that we can interpret properties as subsets of the reals, we proceed to define the property measure and show that it meets the requirements of a measure. We also show that it absorbs as a special case the measure described in Section 2.

Definition 1. *The property measure $\nu : \mathcal{P} \to I$ is given by*

$$\nu = \mu \circ f, \tag{12}$$

where μ is the Lebesgue measure.

Theorem 1. *ν is a well-defined measure.*

Proof. Since μ is nonnegative, ν is nonnegative. Moreover, we have $\nu(\emptyset) = \mu \circ f(\emptyset) = \mu(\emptyset) = 0$. Finally, we have to verify countable additivity. Let $\phi, \phi' \in \mathcal{P}$ such that $\phi \cap \phi' = \emptyset$. We observe that irrational numbers have a unique binary representation. Therefore, if $\|\cdot\|$ maps more than one trace to the same real number, this number must be rational. We compute the measure: $\nu(\phi \cup \phi') = \mu \circ f(\phi \cup \phi') = \mu\big((f(\phi) \cup f(\phi')) \cap \big(I_q \cup I_q^c\big)\big)$, where I_q and I_q^c are respectively the rational and irrational numbers in I. The measure of $\phi \cup \phi'$ becomes

$$\begin{aligned}&\mu\big(f(\phi) \cap I_q^c \cup f(\phi') \cap I_q^c \cup (f(\phi) \cup f(\phi')) \cap I_q\big) = \\ &\mu\big(f(\phi) \cap I_q^c\big) + \mu\big(f(\phi') \cap I_q^c\big) \\ &+ \mu((f(\phi) \cup f(\phi')) \cap I_q).\end{aligned}$$

Since the Lebesgue measure of a countable set is zero, we have

$$\begin{aligned}\nu(\phi \cup \phi') &= \mu\big(f(\phi) \cap I_q^c\big) + \mu\big(f(\phi') \cap I_q^c\big) \\ &= \mu\big(f(\phi) \cap I_q^c\big) + \mu\big(f(\phi') \cap I_q^c\big) \\ &+ \mu(f(\phi) \cap I_q) + \mu(f(\phi') \cap I_q) \\ &= \mu(f(\phi)) + \mu(f(\phi')) = \nu(\phi) + \nu(\phi').\end{aligned}$$

Countable additivity follows by induction.

Equation (12) can be rewritten as $\nu(\phi) = \int_{f\phi} d\mu$. This expression allows us to easily generate new measures that weight traces unequally. Suppose g is a measurable function from I to the nonnegative reals with $\int_{x \in I} g(x)dx = 1$. Then g can be used to weight traces in order to attribute different importance to different traces. The weighted measure is given by

$$\nu(\phi) = \int_{f\phi} g d\mu. \tag{13}$$

We leave further research of weighted metrics for future work. Now we proceed to construct a metric using the measure, but first we show that this notion of the measure generalizes the discussion of Section 2.

Bounded case. Suppose ϕ is an LTL formula defined for a bound $T = N$ over a finite set Σ of atomic propositions. While introducing the measure ν through counting considerations in Section 2, we first interpreted the given formula using the bounded semantics described in Section 2.2, thus generating a new formula ϕ' which makes statements only over the first $N+1$ time steps of the traces. Suppose ϕ' has m satisfying assignments $\sigma_1, \ldots, \sigma_m$, where the σ_i are traces of length $N+1$. Thus, we can write $\phi' = (\sigma_i)_{i=1}^m$. Let ϕ'' be a formula syntactically equal to ϕ', but interpreted over infinite traces. Since ϕ' only makes statements over the first $N+1$ time steps, any infinite extension of σ_i satisfies ϕ''. Let $\sigma_i' = \{\gamma | \gamma \in \mathcal{T} \text{ and } \gamma[0 \ldots N] = \sigma_i\}$; that is, σ_i' contains all infinite extensions of the bounded trace σ_i. Then, $\gamma \models \phi''$ for all $\gamma \in \sigma_i'$ for $1 \leq i \leq m$. Moreover, in this case we have $\phi'' = \cup_{i=1}^m \sigma_i'$. The σ_i' are disjoint, so $\nu(\phi'') = \sum_{i=1}^m \nu(\sigma_i') = m2^{-(N+1)|\Sigma|}$, which matches the counting interpretation of the measure discussed in Section 2.

4.4 A metric for properties

For any two sets X and Y, we let their symmetric difference be given by $X \triangle Y = (X-Y)\cup(Y-X)$. We now define a function that we use to build a metric.

Definition 2. *Let $\phi, \phi' \in \mathcal{P}$. The* property distance *$d : \mathcal{P}^2 \to I$ is given by $d(\phi,\phi') = \nu(\phi \triangle \phi')$.*

Proposition 1. *d satisfies the triangle inequality.*

Proof. Let $\phi,\phi' \in \mathcal{P}$. We provide some intermediate results:

a. Here we show that $\phi \subseteq \phi' \Rightarrow \nu(\phi) \leq \nu(\phi')$; Suppose $\phi \subseteq \phi'$; then $\nu(\phi') = \nu((\phi'-\phi)\cup\phi) = \nu(\phi'-\phi)+\nu(\phi) \geq \nu(\phi)$ because ν is nonnegative.
b. Now we wish to show that $\nu(\phi\cup\phi') \leq \nu(\phi)+\nu(\phi')$. We observe that

$$\begin{aligned}\nu(\phi\cup\phi') &= \nu((\phi-\phi')\cup(\phi'-\phi)\cup\phi\cap\phi') \\ &= \nu(\phi-\phi')+\nu(\phi'-\phi)+\nu(\phi\cap\phi') \\ &\leq \nu(\phi-\phi')+\nu(\phi'-\phi)+2\nu(\phi\cap\phi') \\ &= \nu(\phi)+\nu(\phi').\end{aligned}$$

Let $\psi \in \mathcal{P}$. We have $\phi \triangle \phi' \subseteq (\phi \triangle \psi)\cup(\phi' \triangle \psi)$. Therefore, applying (a) followed by (b), we obtain $d(\phi,\phi') \leq d(\phi,\psi)+d(\phi',\psi)$.

Proposition 2. *d does not satisfy the strong triangle inequality.*

Proof. Let $\phi,\phi',\phi'' \in \mathcal{P}$. The strong triangle inequality, or ultrametric inequality, requires $d(\phi,\phi') \leq \max\{d(\phi,\phi''),d(\phi',\phi'')\}$. Suppose that ϕ'' is empty and ϕ and ϕ' are nonempty, disjoint, and have positive measure. Then $d(\phi,\phi') = \nu(\phi)+\nu(\phi') > \max\{\nu(\phi),\nu(\phi')\} = \max\{d(\phi,\phi''),d(\phi',\phi'')\}$. This counterexample proves the proposition.

Let $\bar{\mathcal{P}} = \mathcal{P}/R$, where $R = \{(\phi,\phi') \in \mathcal{P}^2 \,|\, d(\phi,\phi') = 0\}$; that is, the equivalence classes of $\bar{\mathcal{P}}$ consist of properties whose pairwise distance is zero.

Definition 3. *Let $\bar{\phi},\bar{\phi}' \in \bar{\mathcal{P}}$. We define the property metric $\bar{d} : \bar{\mathcal{P}}^2 \to \mathbb{R}$ as $d(\bar{\phi},\bar{\phi}') = d(\phi,\phi')$, where $\phi \in \bar{\phi}$ and $\phi' \in \bar{\phi}'$.*

Proposition 3. *$\bar{d}$ is a metric.*

Proof. First we have to show that $\bar{d}$ is well-defined. Let $\bar{\phi}$ and $\bar{\phi}'$ be equivalence classes in $\bar{\mathcal{P}}$ and let $\phi,\psi \in \bar{\phi}$ and $\phi' \in \bar{\phi}'$. We have

$$\begin{aligned}d(\phi,\phi') &\leq d(\phi,\psi)+d(\phi',\psi) = d(\phi',\psi) \\ &\leq d(\phi',\phi)+d(\phi,\psi) = d(\phi',\phi) \Rightarrow d(\phi,\phi') = d(\phi',\psi).\end{aligned}$$

Thus, $\bar{d}$ is independent of the members of the equivalence classes used in its computations. It follows that $\bar{d}$ is well-defined. Moreover, $\bar{d}$ is nonnegative because ν

is nonnegative; it satisfies the triangle inequality because d satisfies it (see Proposition 1); it is symmetric due to the symmetry of $\triangle$. It remains to be shown that $\bar{d}(\bar{\phi},\bar{\phi}')=0 \Leftrightarrow \bar{\phi}=\bar{\phi}'$.

Suppose $d(\bar{\phi},\bar{\phi}')=0$ and $\phi \in \bar{\phi}$ and $\phi' \in \bar{\phi}'$. Then $d(\phi,\phi')=0$. But this means that ϕ and ϕ' belong to the same equivalence class in $\bar{\mathcal{P}}$, so $\bar{\phi}=\bar{\phi}'$. Conversely, suppose that $\bar{\phi}=\bar{\phi}'$ and let $\phi \in \bar{\phi}$ and $\phi' \in \bar{\phi}'=\phi$. Then ϕ and ϕ' belong to the same equivalence class in $\bar{\mathcal{P}}$, which means that $d(\phi,\phi')=0$; therefore, $\bar{d}(\bar{\phi},\bar{\phi}')=0$.

In consequence, we can use d to measure the distance between two given LTL properties. Moreover, the distance between two formulas will be zero iff they differ by a property of measure zero.

4.5 Calculating with the property measure

Suppose ϕ is defined over a single atomic proposition a and $\phi=a$. Then ϕ contains all traces that start with a 1. Clearly, $f\phi=[0.5,1]$. Therefore, application of (12) results in $\nu(\phi)=0.5$.

Suppose $\phi=a\wedge\mathbf{X}a\vee\neg\mathbf{X}^2a$. We observe that $\nu(\mathbf{X}a)$ contains all numbers whose second digit is 1. Thus, $\nu(\mathbf{X}a)=[1/4,1/2]\cup[3/4,1]$. Figure 3 shows the intervals to which each of the terms in ϕ map. We observe that a has one segment of size 2^{-1}; $\mathbf{X}a$ has two segments of size 2^{-2}; and X^2a has 4 segments of size 2^{-3}. The property $a\wedge\mathbf{X}a$ intersects the segment of a with one of the segments of $\mathbf{X}a$, as shown in Figure 3b. Thus, we have $\nu(a\wedge\mathbf{X}a)=2^{-2}$. Finally, we observe that we have to add three of the 4 segments of $\mathbf{X}^2a$ to get the final answer, and we obtain $\nu(\phi)=2^{-2}+3\cdot2^{-3}=5/8$.

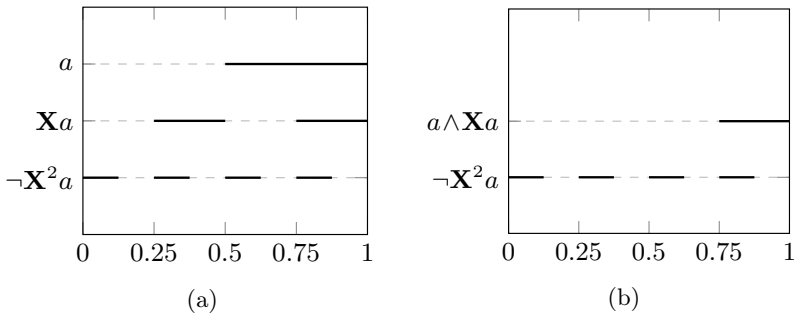

Fig. 3: Computing the measure of the property $a\wedge\mathbf{X}a\vee\neg\mathbf{X}^2a$.

5 Related Work

A *robust* semantics of LTL has been presented by Tabuada *et al.* in [15] which formalizes the idea that a trace that violates a property only finitely many times is

preferable over one that always violates it; the trace that is in lesser violation must therefore be "closer" to satisfying the specification. For LTL Assume/Guarantee properties ($\varphi \Rightarrow \psi$), a *da Costa* algebra is used to compute how much a trace is allowed to diverge from ψ given a violation of φ. Other papers, [9], [1], by defining metrics, also investigate the relation between traces and temporal properties. The main difference between these papers and our approach is that we focus solely on specifications, irrespective of particular implementations and the degree to which they satisfy those specifications.

In [4], Finkbeiner *et al.* explore the theoretical complexity of the problem of LTL model counting for safety formulas, extended to full LTL by Torfah *et al.* [17]. They distinguish between two classes of models, i.e., word- and tree-models of bounded size. In this paper, we focus on sets of traces, which closely relate to the problem of word-model counting. It is shown in [4] that this counting problem is #PSPACE-complete when a binary-encoded bound is used, and they provide a counting algorithm that is linear in the temporal bound but double-exponential in the size of the formula. Although the complexity of the problem in the worst case lies within this theoretical framework, our compositional approach is more tractable in practice. Indeed, by invoking a SAT model counter on smaller problem instances, we can compute the measure of longer properties, which are hard to analyze using the techniques introduced by Finkbeiner *et al.*

The work that we find most closely related to our own is a recent pre-print by Madsen *et al.* [10], which proposes two metrics for Signal Temporal Logic (STL) properties which are assumed rectangular and belonging to a compact vector space. One is based on the *Hausdorff* distance, and the other is based on the Symmetric Difference (SD) between two properties. Albeit developed independently, our LTL metric shares significant similarities with their SD metric. There are, however, some fundamental differences. Madsen *et al.* focus on STL, and formulates some restrictive assumptions to ensure the metrics and the proposed algorithms are correct. We focus on LTL, and do not make any such assumptions. Moreover, the Lebesgue measure used in the SD metric is applied directly to properties, whereas we first transform the property into a subset of I and take the Lebesgue measure afterwards. Lastly, unlike the SD metric, our metric can be computed compositionally provided the necessary independence conditions.

6 Conclusions

This paper provides a measure and a metric for sets of infinite traces satisfying a given LTL property (referred to as measure of the LTL formula) defined over a finite set of atomic propositions. We have shown how the measure behaves for bounded LTL properties, and provided an implementation that computes the measure of bounded LTL properties based on considerations of independence and on model counting. We plan to continue developing measure composition rules to handle common properties. Now that we have a measure and a metric available, we plan to use it to implement greedy approaches for LTL synthesis.

Future theoretical work includes developing the graphical calculus outlined in Section 4.5. This may enable more efficient means for computing the property measure and for carrying out LTL model counting. Moreover, the property intervals we use to compute the measure may have implications for model checking and synthesis. Indeed, we showed that these intervals simultaneously encode the measure and the logical structure of the properties. Moreover, the map $f : \mathcal{P} \to 2^I$ mapping properties to subsets of the unit interval possibly is also capable of mapping morphisms of properties to morphisms of subsets of I. This would turn f into a functor. Researching this could unveil additional algebraic structure that may be possible to import into the world of properties via tools from category theory. Finally, we have shown how the property measure can be extended to weighted property measures, but we leave the explorations of specific costs for future work.

Acknowledgments

We gratefully acknowledge valuable conversations with Tiziano Villa and Daniel Fremont during the preparation of this manuscript.

The work in this paper was supported in part by the National Science Foundation (NSF), awards #CNS-1739816 (Quantitative Contract-Based Design Synthesis and Verification for CPS Security) and #CNS-1836601 (Reconciling Safety with the Internet), and the iCyPhy Research Center (Industrial Cyber-Physical Systems), supported by Avast, Camozzi Group, Denso, Ford, and Siemens.

References

1. de Alfaro, L., Faella, M., Stoelinga, M.: Linear and branching metrics for quantitative transition systems. In: Díaz, J., Karhumäki, J., Lepistö, A., Sannella, D. (eds.) Automata, Languages and Programming. pp. 97–109. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)
2. Benveniste, A., Caillaud, B., Nickovic, D., Passerone, R., Raclet, J.B., Reinkemeier, P., Sangiovanni-Vincentelli, A., Damm, W., Henzinger, T.A., Larsen, K.G.: Contracts for system design. Foundations and Trends®in Electronic Design Automation **12**(2-3), 124–400 (2018)
3. De Giacomo, G., Vardi, M.Y.: Linear temporal logic and linear dynamic logic on finite traces. In: IJCAI. vol. 13, pp. 854–860 (2013)
4. Finkbeiner, B., Torfah, H.: Counting models of linear-time temporal logic. In: Dediu, A.H., Martín-Vide, C., Sierra-Rodríguez, J.L., Truthe, B. (eds.) Language and Automata Theory and Applications. pp. 360–371. Springer International Publishing, Cham (2014)
5. Gomes, C.P., Sabharwal, A., Selman, B.: Model counting. In: Biere, A., Heule, M., van Maaren, H. (eds.) Handbook of satisfiability, chap. 10, pp. 633–651. IOS Press (2008)
6. Iannopollo, A., Tripakis, S., Sangiovanni-Vincentelli, A.: Specification decomposition for synthesis from libraries of LTL Assume/Guarantee contracts. In: 2018 Design, Automation Test in Europe Conference Exhibition (DATE). pp. 1574–1579 (March 2018)

7. Iannopollo, A., Tripakis, S., Sangiovanni-Vincentelli, A.: Constrained synthesis from component libraries. In: Kouchnarenko, O., Khosravi, R. (eds.) Formal Aspects of Component Software. pp. 92–110. Springer International Publishing, Cham (2017)
8. Íncer Romeo, Í., Sangiovanni-Vincentelli, A., Lin, C.W., Kang, E.: Quotient for assume-guarantee contracts. In: MEMOCODE (2018)
9. Jakšić, S., Bartocci, E., Grosu, R., Ničković, D.: Quantitative monitoring of STL with edit distance. In: Falcone, Y., Sánchez, C. (eds.) Runtime Verification. pp. 201–218. Springer International Publishing, Cham (2016)
10. Madsen, C., Vaidyanathan, P., Sadraddini, S., Vasile, C.I., DeLateur, N.A., Weiss, R., Densmore, D., Belta, C.: Metrics for signal temporal logic formulae. ArXiv e-prints (Aug 2018), https://arxiv.org/abs/1808.03315
11. Pnueli, A.: The temporal logic of programs. In: 18th Annual Symposium on Foundations of Computer Science (SFCS 1977). pp. 46–57 (Oct 1977)
12. Sangiovanni-Vincentelli, A., Damm, W., Passerone, R.: Taming Dr. Frankenstein: Contract-based design for cyber-physical systems. European journal of control **18**(3), 217–238 (2012)
13. Sistla, A.P., Clarke, E.M.: The complexity of propositional linear temporal logics. Journal of the ACM **32**(3), 733–749 (Jul 1985)
14. Solar-Lezama, A., Tancau, L., Bodik, R., Seshia, S., Saraswat, V.: Combinatorial sketching for finite programs. In: Proceedings of the 12th International Conference on Architectural Support for Programming Languages and Operating Systems. pp. 404–415. ASPLOS XII, ACM, New York, NY, USA (2006)
15. Tabuada, P., Neider, D.: Robust linear temporal logic. ArXiv e-prints (Oct 2015), https://arxiv.org/abs/1510.08970
16. Thurley, M.: SharpSAT – Counting models with advanced component caching and implicit BCP. In: Biere, A., Gomes, C.P. (eds.) Theory and Applications of Satisfiability Testing - SAT 2006. pp. 424–429. Springer Berlin Heidelberg, Berlin, Heidelberg (2006)
17. Torfah, H., Zimmermann, M.: The complexity of counting models of linear-time temporal logic. Acta Informatica **55**(3), 191–212 (May 2018)