Title

Directed Specifications and Assumption Mining for Monotone Dynamical Systems

Permalink

https://escholarship.org/uc/item/58g4p1dj

ISBN

9781450339551

Authors

Kim, Eric S
Arcak, Murat
Seshia, Sanjit A

Publication Date

2016-04-11

DOI

10.1145/2883817.2883833

Directed Specifications and Assumption Mining for Monotone Dynamical Systems [*]

Eric S. Kim, Murat Arcak, Sanjit A. Seshia
{eskim, arcak, sseshia}@eecs.berkeley.edu
University of California at Berkeley, Berkeley, CA, USA
Department of Electrical Engineering and Computer Sciences

ABSTRACT

Given a dynamical system and a specification, assumption mining is the problem of identifying the set of admissible disturbance signals and initial states that generate trajectories satisfying the specification. We first introduce the notion of a directed specification, which describes either upper or lower sets in a partially ordered signal space, and show that this notion encompasses an expressive temporal logic fragment. We next show that the order preserving nature of monotone dynamical systems makes them amenable to a systematic form of assumption mining that checks numerical simulations of system trajectories against directed specifications. The assumption set is then located with a multidimensional bisection method that converges to the boundary from above and below. Typical objectives in vehicular traffic control, such as avoiding or clearing congestion, are directed specifications. In an application to a freeway flow model with monotone dynamics, we identify the set of vehicular demand profiles that satisfy a specification that congestion be intermittent.

Keywords

Monotone Control Systems, Temporal Logic, Partially Ordered Sets, Assumption Mining

1. INTRODUCTION

Component-based design and analysis is a common paradigm for managing complexity in large networked systems. Each component is characterized by an input-output relationship, such as a finite input-output gain in control theory or an assume-guarantee contract in the formal methods literature, enabling higher order reasoning about global behavior.

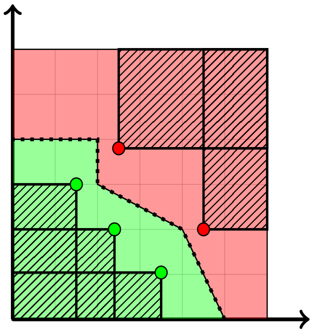


Figure 1: Geometry of a lower set (green), an upper set (red) and their boundary (dotted line). If a point (green dot) is in a lower set, then we can extrapolate that all points below it (patterned box) are also contained in that set.

To develop complex, yet robust, systems through interconnections of simpler components, identifying failure modes and determining the limits for safe system operation is of utmost importance. Towards this end, we formulate an assumption mining problem for dynamical systems where, given a deterministic system and a specification encoded as a set of acceptable state trajectories, we seek the largest set of initial states and exogenous disturbances for which the system satisfies the specification. Computing an exact representation of the assumption set for arbitrary specifications and dynamics is impossible. The next best option is to systematically and extensively test the system under a variety of environmental disturbances to construct an approximation of the assumption set.

In this paper, we define directed specifications and show that when paired with monotone dynamical systems, they are well suited to a systematic form of assumption mining. Directed specifications correspond to lower and upper sets in a signal space and thus favor signals with low or high values respectively. For instance, in a traffic network where the state represents number of vehicles, the specification "congestion will never be present" is a directed specification because it encourages state signals with low vehicle counts. As depicted in Fig. 1, lower and upper sets have a convenient geometry that makes it possible to use individual samples to extrapolate information about set membership. We provide a set of syntactic rules that provide a sufficient condition for a temporal logic specification to be directed. This condition is agnostic to timing semantics of the chosen temporal

[*] This work was supported in part by NSF grant CNS-1446145, the NSF Graduate Research Fellowship Program, NSF Expeditions grant CCF-1139138 and by STARnet, a Semiconductor Research Corporation program, sponsored by MARCO and DARPA.

logic specification language and encompasses linear temporal logic [17] and signal temporal logic [12] as long as predicates are over a partially ordered set.

Monotone dynamical systems exhibit order preserving dynamics and provide a clear functional relationship between the space of initial states and disturbance signals with the trajectories they generate. In particular, they preserve the aforementioned directed property, and the assumption set must be lower(upper) if the specification is lower(upper).

To construct a tight approximation of a directed set in Euclidean space, it suffices to converge to its boundary from below and above. We exploit the extrapolation property of directed sets highlighted in Fig. 1 and utilize a variant of multi-dimensional binary search for discrete time signals of finite length and spaces of finite dimension to converge to the boundary. Our solution uses simulation traces and harnesses a satisfiability modulo theories (SMT) solver [3] to systematically explore the space of initial state and disturbance signals until it provides a certificate that the assumption set is approximated to a desired precision.

To summarize, we make two primary contributions about mining assumptions for monotone control systems:

1. We define directed specifications and show that they encompass an expressive temporal logic fragment.
2. Given a directed specification and a monotone system, we characterize the set of admissible disturbance signals and show how a bisection algorithm exploits the ordering present in the problem.

Section 3 first describes directed specifications geometrically as a subset of the signal space and shows how to construct directed temporal logic specifications, Section 4 reviews monotone dynamical systems, and Section 5 explains how we can exploit both the specification and dynamics via a generalized bisection method.

Related Work

Assumption mining has previously been studied for the synthesis of discrete controllers that realize a temporal logic specification [1][4][16]. Our formulation of the assumption mining problem resembles the problem of computing weakest preconditions but we also compute admissible disturbance profiles [10].

Our work contains a number of parallels to prior work on requirement mining by Jin, Donze, Deshmukh, and Seshia [14] and robust controller synthesis by Topcu, Ozay, Liu, and Murray [19]. Both make use of partial orderings and allude to similar bisection search heuristics, but neither utilize properties of directed sets and their orderings are over different sets than those found in this paper. Jin et al. use monotonicity of a parametric signal temporal logic (pSTL) template to prune regions of the parameter space and find a tight overapproximation of the system's possible state trajectories. Fundamentally, monotonicity with respect to pSTL parameters is about comparing two different specifications and checking if one implies the other. Directedness, in contrast, is a property that is intrinsic to a single specification. In addition, in our work monotonicity is a dynamical system property and should not be confused with monotonicity with respect to specification parameters. Topcu et al. seek to synthesize a controller that is robust to the presence of an environmental adversary with varying levels of strength, where a stronger adversary has more available moves in the synthesis game. Our problem is formulated in a way that stronger environments are encoded directly in the partial ordering on disturbance signals, instead of as a larger set of available environment disturbances.

2. PRELIMINARIES

2.1 Notation and Terminology

For a given set $\mathcal{P}$ we let $\mathcal{P}^C$ and $\mathcal{P} \times \mathcal{Q}$ respectively denote its complement(with respect to some universal set) and its Cartesian product with $\mathcal{Q}$. The empty set is $\emptyset$. The symbol $\Rightarrow$ represents the Boolean implication operator while the symbol $\mapsto$ represents a map between a domain and codomain. The *image* $f(\mathcal{M})$ of a set $\mathcal{M} \subseteq \mathcal{P}$ under function $f : \mathcal{P} \mapsto \mathcal{Q}$ is the set of points $\{f(x) : x \in \mathcal{M}\}$ and the *preimage* $f^{-1}(\mathcal{N})$ of the set $\mathcal{N} \subseteq \mathcal{Q}$ is $\{x \in \mathcal{P} : f(x) \in \mathcal{N}\}$.

The sets $\mathbb{R}_{\geq 0}$ and $\mathbb{Z}_{\geq 0}$ are the sets of non-negative real numbers and integers with $\mathbb{R}^n_{\geq 0}$ representing the non-negative orthant. A discrete time interval $I = [a, b]$ is a contiguous subset of $\mathbb{Z}_{\geq 0}$ where $a, b \in \mathbb{Z}_{\geq 0} \cup \{\infty\}$ and $a \leq b$. The Boolean domain is denoted $\mathbb{B} = \{\perp, \top\}$ where $\top$ is true and $\perp$ is false.

The sets $\mathcal{X} \subset \mathbb{R}^n$, $\mathcal{D} \subset \mathbb{R}^m$ and $\mathcal{Y} \subset \mathbb{R}^p$ represent state, disturbance, and output spaces for appropriate positive integers n, m, p and let $x[k], d[k], y[k]$ respectively denote variables in these spaces at time k. When clear from context, the time index is dropped. For a set $\mathcal{P}$ and an interval I, the space of *signals*, $\mathcal{P}[\cdot]$, is given by a Cartesian product indexed by elements of I:

$$\mathcal{P}[\cdot] = \prod_{k \in I} \mathcal{P}. \tag{1}$$

For instance, a discrete-time real signal of length N can be thought of as a point in $\mathbb{R}^N$. In this paper, the terms signal, trace, and trajectory are synonyms. For notational brevity, the interval I is typically omitted and only specified when necessary. The sets $\mathcal{X}[\cdot], \mathcal{D}[\cdot], \mathcal{Y}[\cdot]$ are the set of state signals, disturbance signals, and output signals.

A specification ϕ can be viewed as the subset of the signal space for which it is true. We signify that a signal $x[\cdot]$ *satisfies* a specification ϕ by $x[\cdot] \models \phi$. One can switch between set theoretic and Boolean views of ϕ by the definition $\phi = \{x[\cdot] \in \mathcal{X}[\cdot] : x[\cdot] \models \phi\}$ and the identity $x[\cdot] \models \phi$ if and only if $x[\cdot] \in \phi$ (using the set theoretic definition of ϕ).

2.2 Assumption Mining

A deterministic dynamical system $\Sigma : \mathcal{X} \times \mathcal{D}[\cdot] \mapsto \mathcal{X}[\cdot]$ is a map from an initial state and disturbance pair to a state trajectory. We formulate assumption mining as the problem of determining which exogenous disturbances and initial conditions are permitted for a system to satisfy a specification.

Problem 1 (Assumption Mining). *Given a deterministic system $\Sigma : \mathcal{X} \times \mathcal{D}[\cdot] \mapsto \mathcal{X}[\cdot]$ and a specification $\phi \subseteq \mathcal{X}[\cdot]$,*

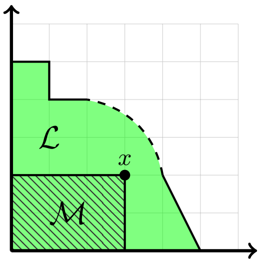


Figure 2: A lower set in $\mathcal{L} \subset \mathbb{R}^2_{\geq 0}$ with the standard ordering. Lower sets are not necessarily convex, open, or closed. The principal lower set $\mathcal{M}$ with associated point x is a subset of $\mathcal{L}$.

what is the subset of initial states and disturbance signals, $\Sigma^{-1}(\phi) \subseteq \mathcal{X} \times \mathcal{D}[\cdot]$, that ensures satisfaction of ϕ?

By viewing a dynamical system Σ as a mapping $\Sigma : \mathcal{X} \times \mathcal{D}[\cdot] \mapsto \mathcal{X}[\cdot]$, the solution to the assumption mining problem is found by computing the pre-image $\Sigma^{-1}(\phi)$ of specification ϕ. Computation of pre-images is typically intractable for arbitrary specifications ϕ and non-linear dynamics for Σ. We focus on monotone dynamical systems and directed specifications and show that this pair is particularly amenable to assumption mining.

2.3 Partially Ordered Sets and Signals

A partially ordered set $\mathcal{P}$ has an associated binary relation $\leq_\mathcal{P}$ if all $p_1, p_2, p_3 \in \mathcal{P}$ satisfy 1) $p_1 \leq_\mathcal{P} p_1$, 2) if $p_1 \leq_\mathcal{P} p_2$ and $p_2 \leq_\mathcal{P} p_1$ then $p_1 = p_2$ and, 3) if $p_1 \leq_\mathcal{P} p_2$ and $p_2 \leq_\mathcal{P} p_3$ then $p_1 \leq_\mathcal{P} p_3$. We define $\geq_\mathcal{P}$ so that $p_1 \geq_\mathcal{P} p_2$ holds if and only if $p_2 \leq_\mathcal{P} p_1$. If neither $p_1 \leq_\mathcal{P} p_2$ nor $p_1 \geq_\mathcal{P} p_2$ hold, we say that p_1 and p_2 are *incomparable*.

Given a collection of partially ordered sets $\mathcal{P}_i$ and relations $\leq_{\mathcal{P}_i}$ indexed by $\mathcal{A}$, let $\mathcal{P} = \prod_{i \in \mathcal{A}} \mathcal{P}_i$, and $\pi_i(p) : \mathcal{P} \mapsto \mathcal{P}_i$ map $p \in \mathcal{P}$ to its i-th component. For $p_1, p_2 \in \mathcal{P}$, the product ordering relation $p_1 \leq_\mathcal{P} p_2$ holds if and only if $\pi_i(p_1) \leq_{\mathcal{P}_i} \pi_i(p_2)$ for all $i \in \mathcal{A}$. Time will frequently play the role of the index set as it does in (1).

In this paper, all sets $\mathcal{X}$, $\mathcal{D}$, and $\mathcal{Y}$ are equipped with partial orders $\leq_\mathcal{X}$, $\leq_\mathcal{D}$, and $\leq_\mathcal{Y}$. We also introduce an induced *signal partial ordering* $\leq_{\mathcal{P}[\cdot]}$ over $\mathcal{P}$ and interval $I = [a, b]$ such that for signals $p_1[\cdot], p_2[\cdot]$ the ordering $p_1[\cdot] \leq_{\mathcal{P}[\cdot]} p_2[\cdot]$ signifies that $p_1[k] \leq_\mathcal{P} p_2[k]$ for all $k \in [a, b]$.

A function between partially ordered sets $f : \mathcal{P} \mapsto \mathcal{Q}$ is a *monotone function* if $p_1 \leq_\mathcal{P} p_2$ implies $f(p_1) \leq_\mathcal{Q} f(p_2)$ for all $p_1, p_2 \in \mathcal{P}$. The composition of monotone functions is also a monotone function [7].

3. DIRECTED SPECIFICATIONS

3.1 Lower and Upper Sets

Definition 1. *(Lower Set) Given a partially ordered set $\mathcal{P}$ with relation $\leq_\mathcal{P}$, a subset $\mathcal{L} \subseteq \mathcal{P}$ is a lower set if for all $p, q \in \mathcal{P}$:*

$$p \in \mathcal{L} \text{ and } q \leq_\mathcal{P} p \Longrightarrow q \in \mathcal{L}. \qquad (2)$$

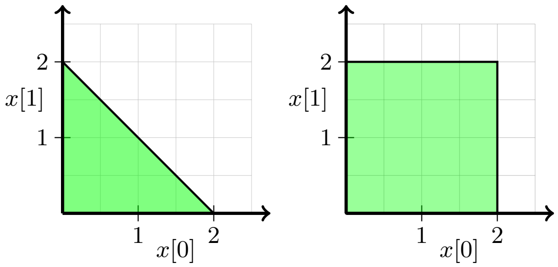


Figure 3: Sets with norm bounds $\{x[\cdot] : ||x[\cdot]||_i \leq 2\}$ for $i \in \{1, \infty\}$ are lower specifications on positive signals $x[\cdot] : [0, 1] \mapsto \mathbb{R}_{\geq 0}$.

An *upper set* satisfies Definition 1 with the relation $\geq_\mathcal{P}$ instead. Common alternative names for lower sets are down sets and downward closed sets [7].

A lower set $\mathcal{M} \subseteq \mathcal{P}$ is a *principal subset* if there exists $x \in \mathcal{P}$ such that $\mathcal{M} = \{y : y \leq x\}$. If $x \in \mathcal{L}$, then $\mathcal{M} \subseteq \mathcal{L}$. With a coordinate-wise ordering, principal sets are rectangles. See Fig. 2 for visualizations of $\mathcal{L}$ and $\mathcal{M}$ in $\mathbb{R}^n_{\geq 0}$.

Lower sets on $\mathcal{P}$ have the following useful properties (the dual properties for upper sets can be obtained by swapping "lower" and "upper") [7]:

Property 1. *If $\mathcal{L} \subseteq \mathcal{P}$ is a lower set, then $\mathcal{L}^C$ is an upper set.*

Property 2. *The collection of all lower sets of $\mathcal{P}$ is closed under arbitrary unions and intersections.*

Property 3. *Let the collection of lower sets $\mathcal{L}_i \subseteq \mathcal{P}_i$ be indexed by a set $\mathcal{A}$ and $\mathcal{P} = \prod_{i \in \mathcal{A}} \mathcal{P}_i$. Their Cartesian product $\prod_{i \in \mathcal{A}} \mathcal{L}_i$ is a lower set with the product ordering $\leq_\mathcal{P}$.*

Property 4. *Sets $\mathcal{P}$ and $\emptyset$ are both upper and lower sets.*

We now define a set of specifications ϕ that are satisfied on lower/upper sets in the signal space.

Definition 2 (Lower/Upper Specifications). *A lower specification ϕ on signals $\mathcal{X}[\cdot]$ satisfies*

$$\Big((x_1[\cdot] \leq_{\mathcal{X}[\cdot]} x_2[\cdot]) \wedge (x_2[\cdot] \models \phi)\Big) \Rightarrow (x_1[\cdot] \models \phi). \qquad (3)$$

Likewise, an upper specification ϕ on signals $\mathcal{X}[\cdot]$ satisfies

$$\Big((x_1[\cdot] \leq_{\mathcal{X}[\cdot]} x_2[\cdot]) \wedge (x_1[\cdot] \models \phi)\Big) \Rightarrow (x_2[\cdot] \models \phi). \qquad (4)$$

A directed specification is one that is either a lower or upper specification.

Common examples of lower specifications in the control theory literature are the set of non-negative signals with upper

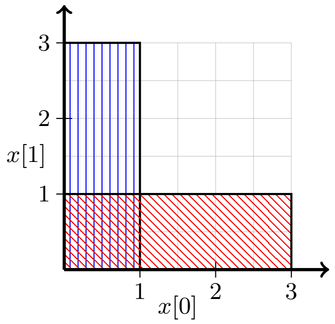


Figure 4: The set of signals $x[\cdot]$ over $\mathcal{X} = [0,3]$ with interval $I = [0,1]$ that satisfy lower specification $\phi = \Diamond_{[0,1]}(x[\cdot] \leq 1)$. Specification ϕ is the union of two clauses $(x[0] \leq 1)$ and $(x[1] \leq 1)$, respectively depicted with vertical and diagonal lines.

bounds on some norm, as shown in Fig. 3, and the set of non-negative signals that converge to zero. However, only considering specifications of interest to be over sublevel sets with respect to some norm is too restrictive and does not permit non-convex specifications such as "intermittent spikes in freeway occupancy are permitted as long as they do not last longer than 30 minutes". Lower sets do not need to be convex, open, or closed, and allow specifications like the freeway specification above. The following section covers a set of rules that provides a sufficient condition for a temporal logic specification to be directed.

3.2 Constructing Directed Specifications

Temporal logics are a logical formalism for expressing specifications as sets of satisfying signals [17]. We restrict our interest to signals over a partially ordered set $\mathcal{X}$. A *predicate* $\mu : \mathcal{X} \mapsto \mathbb{B}$ assigns a truth value to elements in $\mathcal{X}$. As an example, consider a discrete-time variant on signal temporal logic(STL) [12] where specifications can be constructed with the grammar

$$\phi := \top|\mu|\neg\phi|\phi_1 \wedge \phi_2|\phi_1 \mathbf{U}_I \phi_2 \qquad (5)$$

where I is an interval, $\neg$ is Boolean negation, and $\wedge$ is a Boolean AND. Specification $\phi_1 \mathbf{U}_I \phi_2$ is true if there exists a time $k \in I$ such that ϕ_2 is true at time k and ϕ_1 is true until k. From the above grammar, one can derive additional temporal operators $\Diamond_I \phi = \top \mathbf{U}_I \phi$ for "ϕ is eventually true in I" and $\Box_I \phi = \neg(\Diamond_I \neg\phi)$ for "ϕ is always true in I". Formally:

$(x[\cdot], k) \models \mu$	iff	$x[\cdot]$ satisfies μ at time k
$(x[\cdot], k) \models \neg\phi$	iff	$(x[\cdot], k) \not\models \phi$
$(x[\cdot], k) \models \phi_1 \wedge \phi_2$	iff	$(x[\cdot], k) \models \phi_1$ and $(x[\cdot], k) \models \phi_2$
$(x[\cdot], k) \models \phi_1 \mathbf{U}_{[a,b]} \phi_2$	iff	$\exists p \in [k+a, k+b]$ such that $(x[\cdot], p) \models \phi_2$ and $\forall q \in [k+a, p-1], (x[\cdot], q) \models \phi_1$

When temporal operators omit the interval I, it is assumed that $I = [0, \infty)$ and $x[\cdot] \models \phi$ is a shorthand for $(x[\cdot], 0) \models \phi$.

In this section, we adopt a geometric view of temporal logic formulas with predicates on $\mathbb{R}$ by viewing them as subsets of a Euclidean signal space, just as level sets of l_1, l_2 and l_∞ norms are visualized as high dimensional diamonds, balls and boxes. Fig. 4 depicts a specification consisting of an eventually operator and shows that it can be thought of as a union over different sets in the signal space. The mixed-integer constraints appearing in model predictive control with temporal logic constraints effectively encode unions of polyhedra in a signal space [20] [18].

3.3 Order Preserving Operations

We outline a fragment of the temporal logic over a partially ordered set by restricting the grammar (5) in such a way that all generated specifications are directed.

Theorem 1. *Let μ^l and μ^u be restricted to predicates that are true on lower and upper sets of $\mathcal{X}$ respectively. Let ϕ^d be constructed with the grammar*

$$\begin{aligned}
\phi^d &:= \phi^l|\phi^u \\
\phi^l &:= \top|\mu^l|\neg\phi^u|\phi_1^l \wedge \phi_2^l|\phi_1^l \mathbf{U}_I \phi_2^l \\
\phi^u &:= \top|\mu^u|\neg\phi^l|\phi_1^u \wedge \phi_2^u|\phi_1^u \mathbf{U}_I \phi_2^u
\end{aligned}$$

Any specification ϕ^d respecting the grammar above is a directed specification of $\mathcal{X}[\cdot]$ satisfying (3) or (4).

Proof. We adopt a syntax directed approach to proving that a specification is directed and only prove the following statements about lower specifications ϕ^l. The dual statements for upper specifications are easily derived.

- Formulas $\phi = \top$ and $\phi = \bot$ are both lower and upper specifications.
  Proof: Follows from Property 4.
- If predicate μ^l is true on a lower set in $\mathcal{X}$, then $\phi = \mu^l$ is a lower specification in $\mathcal{X}[\cdot]$.
  Proof Sketch: Follows from the definition of lower set, Property 3, and the identity:
  $$\{x[\cdot] : x[0] \in \mu^l\} \equiv \{x[\cdot] : x[\cdot] \in \left(\mu^l \times \prod_{i \in [1,\infty]} \mathcal{X}\right)\}$$
- If ϕ^l is a lower specification, then $\neg\phi^l$ is an upper specification, as shown using DeMorgan's law:
  $$\begin{aligned}
  &(x_1[\cdot] \leq_{\mathcal{X}[\cdot]} x_2[\cdot] \wedge x_2[\cdot] \models \phi^l) \Rightarrow (x_1[\cdot] \models \phi^l) \\
  &\equiv \neg(x_1[\cdot] \leq_{\mathcal{X}[\cdot]} x_2[\cdot] \wedge x_2[\cdot] \models \phi^l) \vee x_1[\cdot] \models \phi^l \\
  &\equiv x_1[\cdot] \not\leq_{\mathcal{X}[\cdot]} x_2[\cdot] \vee x_2[\cdot] \models \neg\phi^l \vee x_1[\cdot] \models \phi^l \\
  &\equiv \neg(x_1[\cdot] \leq_{\mathcal{X}[\cdot]} x_2[\cdot] \wedge x_1[\cdot] \models \neg\phi^l) \vee x_2[\cdot] \models \neg\phi^l \\
  &\equiv (x_1[\cdot] \leq_{\mathcal{X}[\cdot]} x_2[\cdot] \wedge x_1[\cdot] \models \neg\phi^l) \Rightarrow x_2[\cdot] \models \neg\phi^l.
  \end{aligned}$$
- If ϕ_1^l and ϕ_2^l are both lower specifications, then $\phi_1^l \wedge \phi_2^l$ also is a lower specification.
  Proof Sketch: Consider two signals $x_1[\cdot], x_2[\cdot]$ where $x_1[\cdot] \leq_{\mathcal{X}[\cdot]} x_2[\cdot]$. If $x_2[\cdot] \models \phi_1^l \wedge \phi_2^l$, then the inequality $x_1[\cdot] \leq_{\mathcal{X}[\cdot]} x_2[\cdot]$ and the definition of lower specification guarantee that $x_1[\cdot] \models \phi_1^l$ and $x_1[\cdot] \models \phi_2^l$.
- If ϕ_1^l and ϕ_2^l are lower specifications, then $\phi_1^l \mathbf{U}_{[a,b]} \phi_2^l$ also is a lower specification.
  Proof Sketch: Consider two signals $x_1[\cdot], x_2[\cdot]$ where $x_1[\cdot] \leq_{\mathcal{X}[\cdot]} x_2[\cdot]$ and $x_2[\cdot] \models \phi_1^l \mathbf{U}_{[a,b]} \phi_2^l$. At some time $p \in [a,b]$, signal $(x_2[\cdot], p) \models \phi_2^l$ and the definition

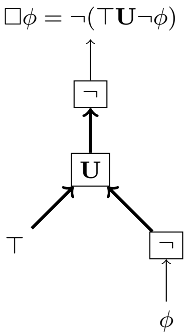


Figure 5: Parse tree that uses order preserving operations to determine that $\Box\phi$ is a lower specification if ϕ is also a lower specification. Thick and thin lines respectively denote upper and lower specifications.

of lower specification guarantees that $(x_1[\cdot], p) \models \phi_2^l$ ($x_1[\cdot]$ may satisfy ϕ_2^l earlier than time p). A similar argument can be made about $x_1[\cdot] \models \phi_1^l$ for all time in $[a, p-1]$.

Although this proof is on discrete-time specifications, the above properties also apply to specifications with continuous-time semantics. □

It is straightforward to prove the above properties about temporal logic operators in a set theoretic context using Properties 1-4 of lower/upper sets. The above are sufficient conditions to determine satisfaction of properties (3) and (4), and allow derivation of similar statements for $\Box\phi$, $\Diamond\phi$, $\phi_1 \vee \phi_2$, and $\phi_1 \Rightarrow \phi_2$.

- $\Box\phi$ and $\Diamond\phi$ are lower specifications if ϕ is a lower specification. Fig. 5 demonstrates how order preserving operators are used to derive $\Box\phi$'s directed property.
- $\phi_1 \vee \phi_2$ is a lower specification if ϕ_1 and ϕ_2 are lower specifications.
- $\phi_1 \Rightarrow \phi_2$ is a lower specification if ϕ_1 is an upper specification and ϕ_2 is a lower specification.

Note that the proof for Theorem 1 only assumes that sets have a partial ordering and no makes no restrictions that predicates be over discrete or continuous sets such as $\mathbb{B}^n$ or $\mathbb{R}^n$, respectively.

Curiously, it is possible to generate lower specifications that combine elements of temporal logics and norms that are not expressible in either alone. Consider

$$\Box_{[0,100]}\Diamond_{[0,7]}(\sum_{i=0}^{4} x[i] \leq 3)$$

which encodes that a running average is periodically below a constant, is a specification that cannot be written in signal temporal logic, yet is still a lower specification.

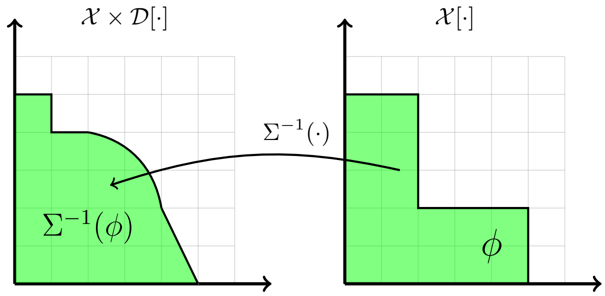


Figure 6: A monotone function's preimage of a lower set is itself a lower set. Therefore, the assumption set $\Sigma^{-1}(\phi) \subseteq \mathcal{X} \times \mathcal{D}[\cdot]$ of the lower specification $\phi \subseteq \mathcal{X}[\cdot]$ is a lower set. Although $\Sigma^{-1}(\phi)$ is unknown, the lower set property is useful for constructing approximations.

4. MONOTONE SYSTEM DYNAMICS

Let the discrete time system Σ have an associated update equation $F_\Sigma : \mathcal{X} \times \mathcal{D} \mapsto \mathcal{X}$ such that

$$x[k+1] = F_\Sigma(x[k], d[k]) \tag{6}$$

for all $k \geq 0$.

Definition 3 (Monotone Systems). *A system (6) is monotone with respect to ordering $\leq_\mathcal{X}$ and $\leq_\mathcal{D}$ if*

$$x_1 \leq_\mathcal{X} x_2 \text{ and } d_1 \leq_\mathcal{D} d_2 \Longrightarrow F_\Sigma(x_1, d_1) \leq_\mathcal{X} F_\Sigma(x_2, d_2). \tag{7}$$

The system has a monotone output if the output function $h(\cdot) : \mathcal{X}[\cdot] \mapsto \mathcal{Y}[\cdot]$ is monotone.

To extend the definition of monotone system from a single-step update equation F_Σ to a definition about the signals generated by Σ, consider $d_1[\cdot] \leq_{\mathcal{D}[\cdot]} d_2[\cdot]$ on the interval $[a, b]$ and $x_1[a] \leq_\mathcal{X} x_2[a]$. If the system is monotone, it follows from iterating Definition 3 via (6) that:

$$x_1[\cdot] \leq_{\mathcal{X}[\cdot]} x_2[\cdot] \tag{8}$$

for state signals $x_1[\cdot], x_2[\cdot]$ on the interval $[a, b+1]$.

Thus monotonicity of $F_\Sigma(\cdot,\cdot)$ in (7) implies the monotonicity of $\Sigma : \mathcal{X} \times \mathcal{D}[\cdot] \mapsto \mathcal{X}[\cdot]$.

Example 1. *Let $a \geq 0, x \in \mathbb{R}_{\geq 0}, d \in \mathbb{R}$. The system*

$$x[k+1] = \max(0, ax[k] + d[k])$$

is monotone.

Consider the assumption mining problem as formalized in Problem 1, but with the additional information that the system Σ is monotone and the specification ϕ is a lower set. The following property of lower sets and monotone functions lets us deduce that the assumption set is also a lower set as depicted in Fig. 6.

Property 5. *If $f : \mathcal{P} \mapsto \mathcal{Q}$ is a monotone function, the preimage $f^{-1}(\mathcal{M})$ of a lower(upper) set, $\mathcal{M} \subseteq \mathcal{Q}$, is itself a lower(upper) set.*

Proof. Consider the case when $\mathcal{M}$ is a lower set. The preimage $f^{-1}(\mathcal{M}) = \{x \in \mathcal{P} | f(x) \in \mathcal{M}\}$ may be the empty set, in which case it satisfies Definition 1. If $f^{-1}(\mathcal{M})$ is nonempty then let there be p_1, p_2 such that $p_2 \in f^{-1}(\mathcal{M})$ and $p_1 \leq_{\mathcal{P}} p_2$. By monotonicity of f, it follows that $f(p_1) \leq_{\mathcal{Q}} f(p_2)$ and $f(p_1) \in \mathcal{M}$ because $\mathcal{M}$ is a lower set. Thus, p_1 is an element of the preimage $f^{-1}(\mathcal{M})$, which satisfies the definition of a lower set. A similar argument can be used when $\mathcal{M}$ is an upper set. □

Because Σ is monotone and ϕ is true on a lower set, the assumption set $\Sigma^{-1}(\phi)$ is a lower set and the assumption violation set $\Sigma^{-1}(\neg\phi)$ is an upper set. Because a composition of monotone functions is monotone, if $\phi \subseteq \mathcal{Y}[\cdot]$ is a directed specification on the system's output signals, then the assumption set will also be directed with the same polarity.

5. ASSUMPTION MINING FOR MONOTONE SYSTEMS

5.1 Approximating the Assumption Set

Determining the set of all admissible initial state and disturbance signals that satisfy (or falsify) arbitrary specifications for nonlinear or hybrid systems is intractable. However, because Σ is monotone and ϕ is directed, we can take advantage of $\Sigma^{-1}(\phi)$'s geometric properties.

We assume that a single simulation of Σ with different initial states and disturbance signals induces a trajectory that either satisfies ϕ or $\neg\phi$. To evaluate a trace's satisfaction of a specification, we would use the Breach or S-TaLiRo toolboxes [11][2]. An initial state and disturbance pair $(x_0, d[\cdot])$ is used to underapproximate $\Sigma^{-1}(\phi)$ if $\Sigma(x_0, d[\cdot]) \models \phi$ and is used to underapproximate $\Sigma^{-1}(\neg\phi)$ if $\Sigma(x_0, d[\cdot]) \models \neg\phi$ (see lines 8-13 of Algorithm 1 and green/red points in Fig. 7a).

Observe that the boundary between a lower and upper set can be under-approximated arbitrarily well by a union of principal sets, which under a coordinate-wise ordering are rectangular. If the assumption space is of finite dimension and is bounded, then the set can be approximated arbitrarily well (in a way that will be made precise) with a finite number of simulations. Solely for the purpose of assumption mining, we impose a practical restriction that disturbance and state signals be of finite length because simulations are of finite length. This implicitly limits the set of specifications that can be mined to those whose satisfaction can be decided for signals of that length. For instance, given a signal $x[\cdot]$ over $I = [0, N]$, the specifications $\Box_{[0,2N]}(x[\cdot] \leq 1)$ and $\Diamond_{[0,2N]}(x[\cdot] \leq 1)$ are disallowed.

A natural question arises of how to intelligently generate disturbance inputs to simulate. This problem becomes more difficult as the signal length increases because the lower set resides in progressively larger dimension spaces. Ideally, one would like to represent the lower set as accurately as possible with a small number of sample points in the assumption space. We provide a modified algorithm that uses the Z3 satisfiability modulo theories (SMT) solver as an oracle to determine the location of the next query [8]. Legriel, Le Guernic, Cotton, and Maler take a similar approach when tackling the problem of estimating a Pareto front of a multi-criteria optimization problem by making queries to a satisfiability solver [15].

If we query a signal in the white region of Fig. 7a, then we are guaranteed to refine our approximation of the ordered set. A rectangle is encoded as an intersection of half planes, and its complement is a union over the opposite half planes. The white region is equivalent to the intersection of the complements of each rectangle. Thus, finding a point in that region can be posed as a conjunctive normal form-like satisfiability query, which we pose to an SMT solver.

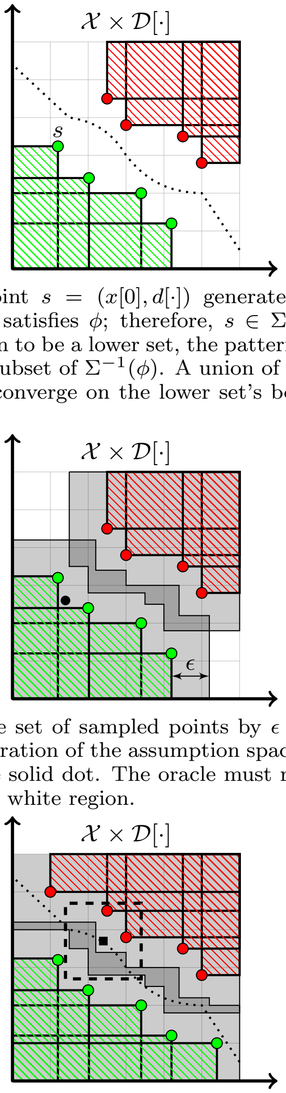
The query may return a point that does not contribute to improving the boundary approximation such as the black dot in Fig. 7b. To explore the signal space effectively, we first "bloat" each rectangle along each side by a constant ϵ immediately before sending the query to the SMT solver. The bloating factor is represented by the grey region in Fig. 7b, and this technique serves a similar purpose to tabu search in [9]. A satisfying query will return a point in the white region, whereas unsatisfiability signifies that the white region does not exist and the entire space $\mathcal{X} \times \mathcal{D}[\cdot]$ is covered.

Unsatisfiability of the queried formula indicates that the rectangle bloating was too aggressive and covered the entire space; it serves as a convenient certificate for the quality of our approximation of the ordered set. Unsatisfiability with an ϵ bloating indicates that every point on the boundary must lie within ϵ coordinate-wise of a point in either the set of upper or lower points as in Fig. 7c. If a query is unsatisfiable, then we decrease the size of ϵ by multiplying by a learning rate $\alpha \in (0, 1)$ until our algorithm reaches a given desired precision ϵ_{final} (see lines 2,4-7 of Algorithm 1) or a timeout.

5.2 Remarks on Tractability

If the assumption space is of finite dimension and is bounded, then a finite number of queries are needed for our algorithm to approximate the set to an arbitrary degree of precision $\epsilon_{\text{final}} > 0$. As a worst case scenario, our algorithm has an upper bound of $(\frac{b}{\epsilon_{\text{final}}})^d$ queries, where d is the dimension of the assumption set and $b = \sup(||x - y||_\infty, x, y \in \mathcal{X} \times \mathcal{D}[\cdot])$ is the diameter of the assumption space. If the assumption space and assumption set are not bounded, it may still be possible for the latter to have a finite ϵ-approximation for all $\epsilon > 0$, but establishing this requires prior knowledge about the assumption set, the very object we are trying to identify.

As highlighted in our freeway example in Section 6 below, the aforementioned upper bound is rather conservative and the number of queries is typically orders of magnitude less. The variable bloating ϵ effectively amounts to a way to vary the grid granularity. Picking a learning rate α requires a delicate tradeoff between exploring the space earlier. A learning rate that is too conservative will result in slower convergence but better coverage of the space, whereas too aggressive a learning rate will rapidly increase the number of points in the grid.


$\mathcal{X} \times \mathcal{D}[\cdot]$
s
$\mathcal{X} \times \mathcal{D}[\cdot]$
ϵ
$\mathcal{X} \times \mathcal{D}[\cdot]$

(a) A green point $s = (x[0], d[\cdot])$ generates a state signal $\Sigma(s) \models \phi$ that satisfies ϕ; therefore, $s \in \Sigma^{-1}(\phi)$. Because $\Sigma^{-1}(\phi)$ is known to be a lower set, the patterned principal set $\{t : t \leq s\}$ is a subset of $\Sigma^{-1}(\phi)$. A union of upper and lower rectangles can converge on the lower set's boundary (dotted line).

(b) Bloating the set of sampled points by ϵ (grey region) to encourage exploration of the assumption space and avoid new samples like the solid dot. The oracle must return a point in the now smaller white region.

(c) After two more samples are taken, all points on the boundary, e.g. the solid square, lie within an ϵ neighborhood (dashed box) of a sampled point.

Figure 7: The assumption mining algorithm consists of generalizing information from simulations (Fig. 7a) and systematic exploration of the assumption space (Fig. 7b). After the algorithm terminates, we have a certificate for the approximation's quality (Fig. 7c).

Algorithm 1 ϵ-approximation of assumption $\Sigma^{-1}(\phi)$

```
Input: $\Sigma, \phi, \alpha \in (0,1), \epsilon, \epsilon_{\text{final}}$
Output: $lowerPts, upperPts$
$lowerPts = [], upperPts = []$
while $\epsilon \geq \epsilon_{\text{final}}$ do
    $(x_0, d[\cdot]) = orderedQuery(lowerPts, upperPts, \epsilon)$
    if ($x_0$ is NaN) then
        $\epsilon = \alpha * \epsilon$
        continue
    end if
    $x[\cdot] = \Sigma(x_0, d[\cdot])$
    if ($x[\cdot] \models \phi$) then
        $lowerPts.append(x_0, d[\cdot])$
    else
        $upperPts.append(x_0, d[\cdot])$
    end if
end while
return $(lowerPts, upperPts)$
```

Algorithm 2 Ordered Query

```
Input: $lowerPts, upperPts, \epsilon$
Output: $x, d[\cdot]$
smtConstraints = []
for all ($(x_i, d_i[\cdot] \in lowerPts$) do
    rect = getLowerRect($x_i, d_i[\cdot]$)
    bloatedRect = bloatRect(rect,$\epsilon$)
    smtConstraints.append(bloatedRect)
end for
for all ($(x_i, d_i[\cdot] \in upperPts$) do
    rect = getUpperRect($x_i, d_i[\cdot]$)
    bloatedRect = bloatRect(rect,$\epsilon$)
    smtConstraints.append(bloatedRect)
end for
satisfied = smt.solve(smtConstraints)
if satisfied then
    return $(x, d[\cdot])$ = smt.model()
else
    return $(x, d[\cdot])$ = (NaN, NaN)
end if
```

6. EXAMPLES

6.1 Monotone Integrator Example

Our first example illustrates the use of our mining algorithm for a simple system and short disturbance signals. Let Σ be a discrete time system with $x \in \mathbb{R}_{\geq 0}, d \in \mathbb{R}$ and update equations

$$x[k+1] = \max(0, ax[k] + d[k]) \tag{9}$$

where $a = .4$. We let the initial state $x[0] = 0$. This system is monotone with respect to the standard ordering on $\mathbb{R}$. The specification $\phi = \Box_{[0,1]}(\Diamond_{[0,2]}(x(t) \leq 4.0))$ is a lower specification according to the rules outlined in Section 3.3 and it is either satisfied or violated with a state signal on the interval $[0,3]$ and an input signal on the interval $[0,2]$. Fig. 8 shows the approximate assumption set for ϕ with an absolute precision $\epsilon = .25$ with a total of 430 points.

6.2 Freeway Example

Our second example is of a freeway traffic network, where we seek to determine the limits of the assumption mining

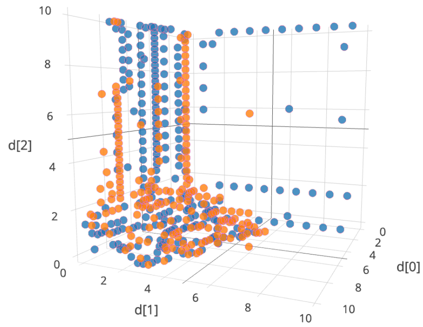


Figure 8: Converging on the boundary of system (9)'s assumption set. Dark blue balls represent points contained within the lower set $\Sigma^{-1}(\phi)$ and lighter red balls are in the upper set $\Sigma^{-1}(\neg\phi)$. We fix the initial state to be zero and only plot the $\mathcal{D}[\cdot]$ component of the assumption set.

algorithm by introducing a large state space and long disturbance signals. Consider the network depicted in Fig. 9, which has a main stretch of three links x_0, x_2, x_4 and two on-ramps x_1, x_3. The dynamics are taken from the cell transmission model (CTM) [6][13], a macroscopic fluid-like model of freeway dynamics. Individual vehicles are not a component of this model. Each discrete time instant represents a five minute interval. Our state space $\mathcal{X} = \prod_{i=1}^{5}[0,100] \subset \mathbb{R}^5_{\geq 0}$ represents the average occupancy over the five minute period in each of the five links. We overload notation and refer to links and their occupancy values using the same variable. The state update equations arise from conservation of mass:

$$\begin{aligned}
x_0[k+1] &= \min(100, x_0[k] - f_0^{\text{out}}[k] + 30)\\
x_1[k+1] &= \min(100, x_1[k] - f_1^{\text{out}}[k] + d_1[k])\\
x_2[k+1] &= \min(100, x_2[k] - f_2^{\text{out}}[k] + \sum_{i=\{0,1\}} f_i^{\text{out}}[k])\\
x_3[k+1] &= \min(100, x_3[k] - f_3^{\text{out}}[k] + d_3[k])\\
x_4[k+1] &= \min(100, x_4[k] - f_4^{\text{out}}[k] + \sum_{i=\{2,3\}} f_i^{\text{out}}[k])
\end{aligned}$$

where $f_i^{\text{out}}[k]$ represents the flow exiting link x_i at time k. The disturbances $d_1[k], d_3[k]$ represent the number of vehicles that would like to enter the network via on-ramps x_1, x_3 and lie within a range $[0, 30]$. We assume link x_0 experiences a constant disturbance of 30 vehicles from the exogenous upstream link. The disturbance space is $\mathcal{D} = \prod_{i=\{1,3\}}[0,30] \subset \mathbb{R}^2_{\geq 0}$. The minimization terms above prevent the occupancy from exceeding the maximum capacity of the freeway segments.

The flows into and out of a link are determined by *supply* and *demand*. A link's demand is the rate at which it would like to send vehicles to downstream links. The demand $\Phi_i(x_i[k])$ that link x_i exhibits is a non-decreasing function

$$\Phi_i(x_i[k]) = \min(c_i, \alpha_i x_i[k]) \tag{10}$$

where c_i is a saturation rate and $\alpha_i \in [0,1]$ is the fraction of current vehicles that will leave x_i. The primary links have saturation rates $c_0 = c_2 = c_4 = 40$ and on-ramps have saturation rates $c_1 = c_3 = 30$. Link x_i also exhibits a supply

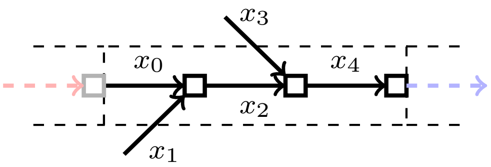


Figure 9: An example network with two on-ramps x_1, x_3. Dashed arrows are exogenous network links

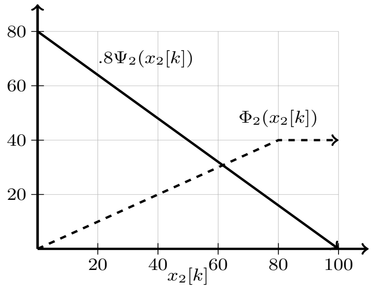


Figure 10: Supply (solid) that link x_2 provides to link x_0 and Demand (dashed) that link x_2 creates for link x_4

$$\Psi(x_i[k]) = 100 - x_i[k], \tag{11}$$

which is the rate of incoming vehicles that it can accept from upstream. A link's supply is partitioned among upstream links, with links x_2, x_4 allocating 80% of their supply to an upstream highway link and 20% to on-ramps. The flow out of a link x_i is the minimum between supply available to it and x_i's demand:

$$\begin{aligned}
f_0^{\text{out}}[k] &= \min\left(.8(100 - x_2[k]), 40, .5x_0[k]\right)\\
f_1^{\text{out}}[k] &= \min\left(.2(100 - x_2[k]), 30, x_1[k]\right)\\
f_2^{\text{out}}[k] &= \min\left(.8(100 - x_4[k]), 40, .5x_2[k]\right)\\
f_3^{\text{out}}[k] &= \min\left(.2(100 - x_4[k]), 30, x_3[k]\right)\\
f_4^{\text{out}}[k] &= \min\left(40, x_4[k]\right)
\end{aligned}$$

The exogenous link that is immediately downstream from x_4 can always accept up to 40 vehicles.

Congestion occurs when demand exceeds supply and the left term in the minimization is active; one can verify that congestion occurs on an upper set of the state space because the left term remains active after an increase in any $x_i[k]$. Let the predicate $c(x)$ be true if congestion is present. Our specification, $\phi = \Box_{[0,T]}\Diamond_{[0,1]}(\neg c(x[\cdot]))$, requires that congestion be intermittent. Note that ϕ's satisfaction value can be determined for inputs over $I = [0, T]$ because the generated state trajectories are over $I = [0, T+1]$. Because a maximum rate of 40 vehicles can exit the network but a net rate of 90 vehicles can enter via links x_0, x_1, x_3 at any time step, ϕ can easily be violated with a determined adversary. The network in Fig. 9 was shown to exhibit monotone dynamics in [5].

We consider disturbance signals $I = [0, T-1]$ with signal

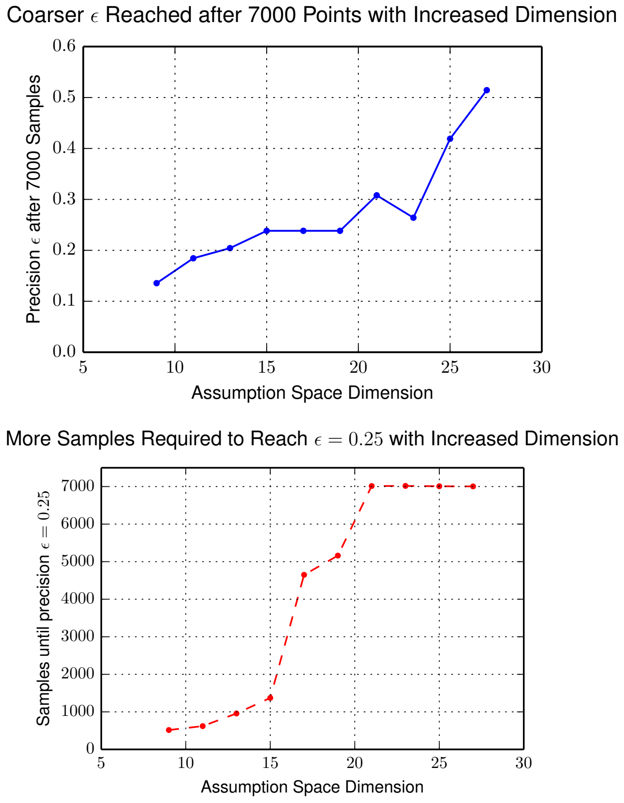


Figure 11: For assumption spaces of varying dimension, precision and number of samples are used as two stopping criteria, with the other criterion plotted after termination. Lower values are better for both precision and samples. The samples plateau in the lower graph appears because of a time out.

lengths $T \in \{2, 3, \ldots, 11\}$. The input is two dimensional and the state space has five dimensions, so the dimension of the assumption space is $2T+5$ and the boundary of the assumption set can be as complex as a $2T+4$ dimensional object. Fig. 11 shows how increasing the dimension of the problem presents a tradeoff between granularity and sample points. The precision ϵ is normalized along each axis; for example, $\epsilon = .1$ represents that demand signals $d_1[k], d_3[k] \in [0, 30]$ have a granularity of a rate of three vehicles at each time k and that $x_i[0] \in [0, 100]$ has granularity of ten vehicles. We opted to test our method for higher dimensions and less granularity (i.e. higher ϵ_{final}) because obtaining a fine approximation of the assumption set doesn't make sense when traffic model parameters are imprecise and the model does not incorporate higher-order dynamics. The learning rate was $\alpha = .95$ and the initial bloating factor was $\epsilon = .6$. Our algorithm required anywhere from 3 to 12 orders of magnitude less samples than the worst case scenario number of samples $(\frac{1}{\epsilon})^{2T+5}$.

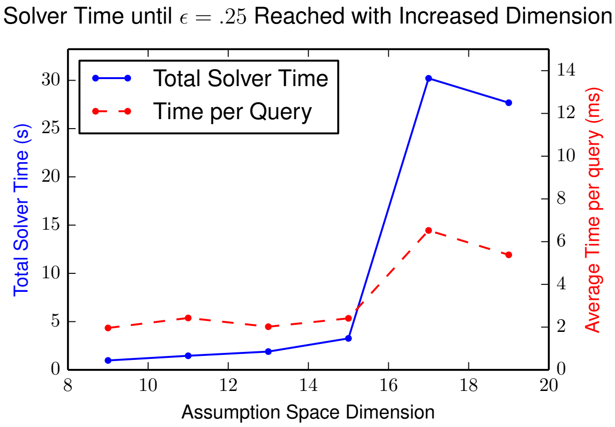


Figure 12: For a fixed $\epsilon_{\text{final}} = .25$, the total solver runtime and average runtime for each SMT query. The increase in total solver time is primarily due to the fact that more points were required to cover the space with an ϵ bloating. The total solver time does not include overhead for declaring constraints and was a variable proportion of total mining time.

Runtimes for $T = \{2, \ldots, 7\}$ are plotted in Fig 12. Larger dimensions did not achieve the desired precision by a mining algorithm timeout of 10 minutes. Experiments were run on a standard on a laptop with 8 GB memory and 2.4 GHz Intel Core i7 processor. As expected, the main factors that influenced the miner's total run time was dimension and the target ϵ_{final}. However, individual SMT query times depended much more on the presence of "easy solutions" than on problem dimension or the number of constraints. In lower dimensions, M samples provide more comprehensive coverage of the ambient space than M samples in a higher dimension space. Thus, for lower dimension spaces obvious solutions to the SMT queries become sparse at sooner than for higher dimensions.

7. CONCLUSION

We have introduced directed specifications and shown how their structure can be exploited in the assumption mining problem for monotone dynamical systems. Future work will focus on seeking out other connections between classes of dynamical systems and temporal logic fragments. Viewing temporal logic specifications as geometric objects in a signal space could enable a common language to explore connections between objectives written as temporal logic specifications and traditional control theoretic properties on signals and dynamical systems.

8. REFERENCES

[1] R. Alur, S. Moarref, and U. Topcu. Counter-strategy guided refinement of GR(1) temporal logic specifications. In *Formal Methods in Computer-Aided Design, FMCAD 2013, Portland, OR, USA, October 20-23, 2013*, pages 26–33, 2013.

[2] Y. Annapureddy, C. Liu, G. Fainekos, and S. Sankaranarayanan. S-TaLiRo: A Tool for Temporal Logic Falsification for Hybrid Systems. In *Proceedings of Tools and Algorithms for the Construction and Analysis of Systems*, 2011.

[3] C. Barrett, R. Sebastiani, S. Seshia, and C. Tinelli. Satisfiability Modulo Theories. In *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, chapter 26, pages 825–885. IOS Press, Feb. 2009.

[4] K. Chatterjee, T. Henzinger, and B. Jobstmann.

Environment Assumptions for Synthesis. In *CONCUR 2008 - Concurrency Theory*, volume 5201 of *Lecture Notes in Computer Science*, pages 147–161. Springer Berlin Heidelberg, 2008.

[5] S. Coogan and M. Arcak. Scalable finite abstraction of mixed monotone systems. *Proceedings of the 18th ACM International Conference on Hybrid Systems: Computation and Control*, 2015.

[6] C. F. Daganzo. The cell transmission model: A Dynamic Representation of Highway Traffic Consistent with the Hydrodynamic Theory. *Transportation Research*, 28:269–287, 1994.

[7] B. Davey and H. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 2nd edition.

[8] L. De Moura and N. Bjørner. Z3: An Efficient SMT Solver. In *Proceedings of the Theory and Practice of Software, 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, TACAS'08/ETAPS'08, pages 337–340, Berlin, Heidelberg, 2008. Springer-Verlag.

[9] J. Deshmukh, X. Jin, J. Kapinski, and O. Maler. Stochastic Local Search for Falsification of Hybrid Systems. In *13th International Symposium on Automated Technology for Verification and Analysis*, 2015.

[10] E. W. Dijkstra. Guarded Commands, Nondeterminacy and Formal Derivation of Programs. *Commun. ACM*, 18(8):453–457, Aug. 1975.

[11] A. Donzé. Breach, a Toolbox for Verification and Parameter Synthesis of Hybrid Systems. In *Proceedings of the 22nd International Conference on Computer Aided Verification*, CAV'10, 2010.

[12] A. Donzé and O. Maler. Robust satisfaction of temporal logic over real-valued signals. In *Proceedings of the 8th International Conference on Formal Modeling and Analysis of Timed Systems*, FORMATS'10, pages 92–106, Berlin, Heidelberg, 2010. Springer-Verlag.

[13] G. Gomes and R. Horowitz. Optimal freeway ramp metering using the asymmetric cell transmission model. *Transportation Research Part C: Emerging Technologies*, 14(4):244 – 262, 2006.

[14] X. Jin, A. Donze, S. A. Seshia, and J. V. Deshmukh. Mining Requirements from Closed-Loop Control Models. In *Hybrid Systems: Computation and Control*, 2013.

[15] J. Legriel, C. Le Guernic, S. Cotton, and O. Maler. Approximating the Pareto Front of Multi-criteria Optimization Problems. In *Tools and Algorithms for the Construction and Analysis of Systems*, volume 6015 of *Lecture Notes in Computer Science*, pages 69–83. Springer Berlin Heidelberg, 2010.

[16] W. Li, L. Dworkin, and S. A. Seshia. Mining assumptions for synthesis. In *In Proc. 9th MEMOCODE*, 2011.

[17] A. Pnueli. The Temporal Logic of Programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pages 46–57, 1977.

[18] V. Raman, A. Donzé, D. Sadigh, R. M. Murray, and S. A. Seshia. Reactive Synthesis from Signal Temporal Logic Specifications. In *18th International Conference on Hybrid Systems: Computation and Control*, HSCC '15, pages 239–248. ACM, 2015.

[19] U. Topcu, N. Ozay, J. Liu, and R. M. Murray. On Synthesizing Robust Discrete Controllers Under Modeling Uncertainty. In *15th ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '12, pages 85–94, New York, NY, USA, 2012. ACM.

[20] E. Wolff, U. Topcu, and R. Murray. Optimization-based trajectory generation with linear temporal logic specifications. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 5319–5325, May 2014.