Light-weight hybrid model checking facilitating online prediction of temporal properties*

Gerald Sauter[1], Henning Dierks[2], Martin Fränzle[1], and Michael R. Hansen[3]

[1] Carl von Ossietzky Universität, Oldenburg, Germany
{fraenzle|sauter}@informatik.uni-oldenburg.de
[2] OFFIS, Oldenburg, Germany
dierks@offis.de
[3] Technical University of Denmark, Kgs. Lyngby, Denmark
mrh@imm.dtu.dk

Abstract. We address the problem of online prediction of future temporal properties of open continuous systems based on time series obtained from online measurements. The objective is, given a partial time series stating (possibly inexact) measurements of observable variables of the system, to safely predict the truth value of a linear-time temporal-logic formula including future modalities, and to do so in real-time while the system is evolving. While this can in principle be done using techniques from hybrid-system model checking, these algorithms are computationally by far too expensive to support online prediction in real-time. As a computationally tractable alternative, we suggest an approach decomposing hybrid model-checking into an offline and an online phase, where the offline phase precomputes all computationally expensive entities, in particular those involving forms of branching, while the online part exploits these precomputed properties in a branching-free analysis based on computationally inexpensive interval constraint propagation.
The procedure is designed for system-in-the-loop testing or similar use cases requiring real-time monitoring of dynamic processes with respect to temporal properties at moderately high sampling frequencies.

"Prediction is very difficult, especially about the future."
[Niels Bohr]

1 Introduction

In this paper, we address the question of system-in-the-loop monitoring of dynamic systems with a moderately fast sampling rate. The objective is, given a partial time series of (possibly inexact) data obtained from online measurements of observable variables of the system, to safely predict the truth value of a Linear-time Temporal Logic (LTL) formula including future modalities, and to do so in real-time while thesystem is evolving.

We are, in particular, considering applications arising from the German IMoST (Integrated Modelling for Safe Transportation) project, which aims at model-based assessment of the total safety impact of new driver assistance systems in road-bound traffic. The general idea of this project is to engineer a model-based development platform for automotive assistant systems based on a seamless integration of cognitive models for human behavior, traffic models, model-based design of assistant systems, and models describing the dynamic behavior of cars. Validation and parameterization of such a development platform, in particular its cognitive-model component, requires a multitude of human-and-assistant-system-in-the-loop driving experiments, calling for fast online monitoring of such complex hybrid systems.

* This work has been partially supported by the Ministry for Science and Culture of Lower Saxony as part of the interdisciplinary project "Integrated Modelling for Safe Transportation" (IMoST), Velux Fonden, ARTIST2 (IST-004527), MoDES (Danish Research Council 2106-05-0022) and DaNES (the Danish National Advanced Technology Foundation)..

We therefore develop an online monitoring tool providing early warning based on safe extrapolation from system history, where the history is recorded as a time series of observations on the system and, in order to achieve sound predictions, the future is safely approximated using differential equations and other constraints pertinent to the system dynamics derived from the models.

Tool support for online monitoring of hybrid systems with respect to requirements expressed using temporal logics has previously been studied in [NM07], where a property based tool (named ATM) for monitoring analogue systems is described. The properties are expressed as formulae of STL/PSL, an extension of MITL [AFH96] and STL [MN04]. A serious limitation of AST, however, is that formulae are just interpreted over a history of past events, forcing online monitoring to fall behind real-time by a finite or even unbounded delay, depending on the sort of future modalities used in formulae. Our work can be seen as an extension of that in the sense that we interpret formulae over infinite extensions of the monitored traces, where the future events are safely approximated using laws describing the dynamics of the system, thus permitting safe prediction of truth values of future modalities. Two techniques are used in order to arrive at useful safe over-approximations of the possible infinite extensions of the sampled finite trace: interval constraint propagation [BG06] is used to efficiently estimate ranges of observations in the bounded future based on ordinary differential equations (ODE) governing parts of the system dynamics, thereby exploiting a mean-value form of the ODEs akin to that used in the hybrid-system model-checker HSolver [RS05], and an offline preprocessing step is used to permit efficient online computation of estimates covering the unbounded future.

Following the intuition that dynamical systems gradually turn their state to the good or bad (e.g., a collision between cars is necessarily preceded by a less and less controllable approach), we aim at providing more than just a qualitative yes/no answer, but instead let our monitor output a quantitative figure in the form of a numerical interval providing a safe estimate of how severe a property violation or how fragile a property satisfaction can at most be given the current situation. To this end, we resort to a *robust semantics* for LTL, where the meaning of a formula is the degree of accuracy of the constants within which it is satisfied. This notion of robustness originates from [Rat00], where it is applied to arithmetic first-order constraints, and has been lifted to the linear-time temporal logics Duration Calculus [FH05] and LTL [FLS08]. We adopt the latter and extend it to an interval-valued interpretation over interval-valued traces, thus being able to accommodate inexact measurements as well as not fully determined extrapolated values in the semantics.

The paper is organized as follows: In the next section we introduce the main ideas of the approach in terms of an example inspired by the IMoST project, where two cars are approaching a highway on an interweaving lane. In Section 3 the notion interval trace is introduced and the robust semantics for linear-time temporal logic formulae is for such traces. Thereafter, in Section 4, it is shown how the models can be approximated and made suitable for computerized predictions. The last section contains a summary and directions to future work.

2 Robust interpretation with prediction – the scenario

The core of the IMoST approach is to integrate engineering and psychological perspectives in a comprehensive model-based design framework for safe transportation systems. The ultimate goal is to provide model-based analysis methods capable of predicting the behavior of a vehicle driven by a computer-assisted human operator. Such analysis is facilitated by means of a seamlessly integrated, heterogeneous dynamic model, replacing the human by a cognitive driver model, the car and its embedded assistance systems by a hybrid-state model, and the surrounding traffic by adequate stochastic behavioral models. The benchmark scenario is embedded assistance for filtering into the motorway, comprising assistance for selecting a slot between cars on the motorway to filter in, help for adequately accelerating the own car on an interweaving lane, and the eventual merge into the motorway. It is obvious that for this scenario, a crucial aspect of safety is adequately measured in terms of the minimum safety distance between cars maintained throughout the whole maneuver, i.e. between entering the interweaving lane and the completed filtering in. Furthermore, it is

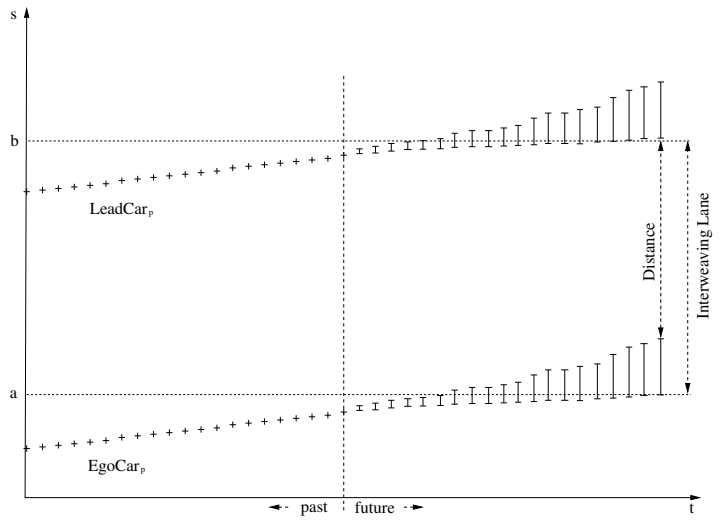

Fig. 1. A trajectory of two traffic agents on an interweaving lane

clear from practical experience with the situation that, over reasonable time frames of prediction, potentially dangerous situations can be predicted in advance on the basis of a model of the driver and the dynamics of the system involving, for example, the position, velocity, acceleration, and physical parameters of the cars. Thus, it is in principle possible to safely evaluate certain temporal logic formulae comprising future operators referring to the future of the current situation.The goal is a timely prediction, which allows for proper actions to avoid crashes.

The dynamic model of a system is expressed in terms of a set of *observables* $\mathcal{O}$ (with typical elements $x, y, z, \dots$) of time-dependent real-valued variables within the domains $D_x, D_y \dots \subseteq \mathbb{R}$. An example is given in Fig. 1, showing a model for two cars on an interweaving lane. There is an observable for each car, $\texttt{LeadCar}_\texttt{p}$ and $\texttt{EgoCar}_\texttt{p}$, respectively, showing the positions of the car at various time instants. The positions a and b are the beginning and end positions, respectively, of the interweaving lane. The time line is segmented into a past and a future. For each time instant in the past, each observable has a specific value, being the position of the particular car at that time.[4] However, for each time in the future the exact positions of the cars are unknown, but an interval of possible positions at a given instant can be estimated exploiting knowledge about suitable bounds on the velocity and acceleration using $v = \frac{\mathrm{d}s}{\mathrm{d}t}$ and $a = \frac{\mathrm{d}v}{\mathrm{d}t}$. Section 4 shows how this can be done effectively and efficiently.

Linear-time temporal logic will be used to formalize safety requirements, and a sample requirement could be that the longitudinal distance between two cars should be at least $5m$ when they are in the region of the interweaving lane:

$$\Box \begin{pmatrix} (LeadCar_p > a \wedge LeadCar_p < b) \Rightarrow \\ (LeadCar_p - EgoCar_p > 5m) \end{pmatrix}$$

Since positions (and other observables) are interpreted as intervals, the meaning of such formulae will be interpreted as intervals over the reals as well, constituting a multi-valued interpretation. Suppose that at time t, we have that

$$LeadCar_p(t) = [L_l, L_u] \qquad \text{and} \qquad EgoCar_p(t) = [E_l, E_u]$$

and consider the atomic formula $LeadCar_p - EgoCar_p > 5m$ at time t. We shall give this formula a so-called *robust* or *quantitative* interpretation [FH05,FLS08] expressing the amount of persistency of the truth value of the formula under variation of its constants.

[4] Note that in general, also some values from the past may be missing due to lost or lacking samples. These information lacks may be bridged with exactly the same method we propose for extrapolation into the future.

If the exact positions of the cars are known, i.e. $L_l = L_u = pos_L$ and $E_l = E_u = pos_E$, then the formula is basically interpreted as a real-numbered value $x_t = (pos_L - pos_E) - 5$ (actually by a singleton interval $[(pos_L - pos_E) - 5, (pos_L - pos_E) - 5]$). Notice that x_t is the robustness margin for (i.e. amount of variation applicable to) the constant 5 before the formula changes its truth value, and confidence intervals on such robustness margins will be computed when monitoring a given property. In general, robust interpretation of a formula will yield an interval $[l, u]$ and the formula is called *robustly true* if $l > 0$, *robustly false* if $u < 0$, and *fragile* or —in monitoring applications— *inconclusive* if $0 \in [l, u]$. In the next section, this form of semantics is elaborated for formulae of linear-time temporal logic.

3 Semantics

Reflecting the fact that in online monitoring, traces are obtained by measurements which may be inexact or even missing (either due to insufficient sampling or due to their time of sampling being in the future), we associate intervals to the individual measurements on the observable variables of the system. Given a finite set of observables $\mathcal{O}$ (with typical elements $x, y, z, \dots$), we assume that such an observable $x \in \mathcal{O}$ ranges over an interval $[l_x, u_x] \subseteq \mathbb{R}$.

Definition 1. *We call a mapping $V : \mathcal{O} \longrightarrow \mathcal{P}(\mathbb{R})$, where for each observable $x \in \mathcal{O}$ it holds that*

- *$V(x)$ is a non-empty interval and*
- *$V(x) \subseteq [l_x, u_x]$*

an interval valuation *of $\mathcal{O}$. Let $\mathcal{V}$ denote the set of all interval valuations of $\mathcal{O}$. We call a mapping $T : \mathbb{N} \longrightarrow \mathcal{V}$ an* interval trace *of $\mathcal{O}$ and denote the set of all interval traces of $\mathcal{O}$ by $\mathcal{T}$.*

Hence, an interval trace describes the evolution of the observables over discrete time[5] where for each time instance and each observable an interval of values is given. Note that this is a conservative extension of the standard definition of discrete time valuations. An interval valuation can be conceived as a (convex) set of standard valuations.

Based on interval traces, we define a semantics of LTL that robustly interprets LTL formulae over such traces and consequently assigns a numeric interval instead of a single truth value, with the assigned interval encoding both the range of possible truth values over the possible concretizations of the interval trace as well as the robustness (wrt. permissible perturbation of the constants in the formula) of these truth values.

Definition 2. *Let T be an interval trace of $\mathcal{O}$. The* quantitative interval semantics *$T[\![\varphi]\!] : \mathbb{N} \longrightarrow 2^{\mathbb{R}}$ for a LTL formula φ is inductively defined as follows:*

$$
\begin{array}{ll}
T[\![x > k]\!]\, i = T(i)(x) - [k,k] & T[\![\neg\varphi]\!]\, i = -T[\![\varphi]\!]\, i \\
T[\![\varphi \wedge \psi]\!]\, i = \min(T[\![\varphi]\!]\, i, T[\![\psi]\!]\, i) & T[\![\bigcirc\varphi]\!]\, i = T[\![\varphi]\!]\, (i+1) \\
T[\![\Box\varphi]\!]\, i = \inf_{j \geq i} T[\![\varphi]\!]\, j & T[\![\varphi \mathsf{U} \psi]\!]\, i = \sup_{j \geq i} \min(T[\![\psi]\!]\, j, \min_{k=i}^{j-1} T[\![\varphi]\!]\, k)
\end{array}
$$

Here, i denotes the temporal position in the trace at which the formula is evaluated and $-$, min*,* sup*, etc., denote the strongest interval liftings of the corresponding base operations.*

The idea of this definition is that the interval $T[\![\varphi]\!]\, i$ encodes two kinds of information. First, it encodes the possible truth values arising from the contained point-valued valuations in the standard interpretation of LTL: If the infimum of $T[\![\varphi]\!]\, i$ is positive, then T contains a (standard) valuation such that the LTL formula φ becomes true in the standard semantics of LTL. Analogously, a negative supremum encodes that T contains a (standard) valuation for which φ is false.

[5] For the sake of simplicity, we refer to equidistant discrete time throughout this paper. For the extension to time-stamped sequences of samples, cf. [Gez09].

Example 1. Assume that $T(0)(x) = [3,6]$. Then $T[\![x > 4]\!](0)$ evaluates to $[-1,2]$ and $T[\![x \leq 7]\!](0) = -[-4,-1] = [1,4]$ because $x \leq 7 \equiv \neg(x > 7)$. Hence, in the first case T contains both a valuation such that $x > 4$ holds and a valuation for which this is false. In the second case we can conclude that T only contains valuations for which $x \leq 7$ holds because the interval semantics yields an interval contained in $\mathbb{R}_{>0}$.

The second kind of information encoded is a measure of robustness. If $m > 0$ is the infimum of $T[\![\varphi]\!]\, i$ this means that there is a (standard) valuation T_s contained in T such that at time instant i satisfies ϕ and does also satisfy all formulae φ' which are structurally equal to φ, yet differ in the constants in the formulae, which may vary by $\pm m$ (including m and $-m$ if $T[\![\varphi]\!]\, i$ is right-closed and possibly excluding it otherwise). Below (Lemma 2), we will see that this does also imply robustness against slight variations in the trajectory T_s.

With these aims in mind we can motivate the choices made in Def. 2. First of all, we observe that the interval semantics corresponds closely to the standard semantics:

Lemma 1 (Interval semantics vs. classical semantics). *Let T be an interval trace of $\mathcal{O}$ and $T_s : \mathbb{N} \to \mathcal{O} \to \mathbb{R}$ a conventional trace consistent with T, i.e., satisfying $\forall i \in \mathbb{N} \forall x \in \mathcal{O} : T_s(i)(x) \in T(i)(x)$.*

1. *If* $\inf T[\![\phi]\!]\, i > 0$ *then* $T_s, i \models \phi$;
2. *if* $\sup\ T[\![\phi]\!]\, i < 0$ *then* $T_s, i \not\models \phi$.

Proof. By straightforward induction over the structure of ϕ. □

I.e., positivity of the interval-valued semantics is a sufficient, yet not necessary, condition for satisfaction by all traces consistent with the interval trace, while negativity is a sufficient, yet not necessary, condition for violation by all traces consistent with the interval trace.

Despite this close correspondence, the interval-valued interpretation has a number of interesting properties that distinguish it from the standard interpretation:

Lemma 2 (Lipschitz-continuity). *For any LTL formula ϕ, the semantic mapping $\_[\![\phi]\!] : \mathcal{T} \to 2^{\mathbb{R}}$ is Lipschitz continuous with constant l, where $l > 0$ is a lower bound of the individual global Lipschitz constants of the expressions occurring in ϕ, with respect to the metrics*

$$d(T_1, T_2) = \sup_{i \in \mathbb{N}} \sup_{x \in \mathcal{O}} ||T_1(i)(x) - T_2(i)(x)|| \ ,$$

where $||[a,b],[c,d]|| = \max\{|a-c|, |b-d|\}$. I.e., for any interval traces T_1 and T_2 and any $i \in \mathbb{N}$,

$$||T_1[\![\phi]\!]\, i - T_2[\![\phi]\!]\, i|| \leq l \cdot d(T_1, T_2)$$

Proof. By straightforward induction over the structure of ϕ. □

Lemma 3 (Monotonicity wrt. interval inclusion). *Let T_1 and T_2 be interval traces over $\mathcal{O}$, with T_2 a pointwise subset of T_1, i.e., satisfying $\forall i \in \mathbb{N} \forall x \in \mathcal{O} : T_1(i)(x) \supseteq T_2(i)(x)$. Let $i \in \mathbb{N}$.*

Then $T_1[\![\phi]\!]\, i \supseteq T_1[\![\phi]\!]\, i$, i.e., the interval-valued semantics is monotonic wrt. interval inclusion.

Proof. By straightforward induction over the structure of ϕ. □

4 Interval traces for prediction

Aiming at reliable prediction of truth values of temporal formulae based on finite time series of measurements and on known constraints on the system dynamics, we do need the following ingredients:

1. A finite representation of infinite interval time series, facilitating effective representation and manipulation,

2. a means of effectively evaluating the infinite-time interval semantics on such finite representations of infinite interval traces,
3. a mechanism for refining (a finite representation of) an interval time series based on known facts about the system dynamics, and
4. a mechanism for refining (a finite representation of) an interval time series upon appearance of new measurements.

In this section, we will provide these techniques.

4.1 Finite representation of infinite interval traces

Given a finite trace prefix $tr : \{0, \ldots, n\} \to \mathcal{V}$ obtained from measurements, and thus having all $tr(i)(o)$ being represented by computational arithmetic data types, the infinite interval trace $T : \mathbb{N} \to \mathcal{V}$

$$T(i)(x) = \begin{cases} tr(i)(x) & \text{iff } i \leq n \\ [l_x, u_x] & \text{iff } i > n \end{cases}$$

is the (least refined or weakest) infinite extension of tr. Such infinite extensions permit an obvious finite representation by adding a single "collecting interval" to the trace prefix tr, where the single collecting interval represents its infinite repetition in the constant tail of the infinite interval trace.

Extending this idea, given $l \in \mathbb{N}$, any infinite interval trace $T : \mathbb{N} \to \mathcal{V}$ permits a conservative finitely representable approximation T_l with

$$T_l(j)(x) = \begin{cases} r(T(j)(x)) & \text{iff } j < l, \\ r(\bigcup_{k=l}^{\infty} T(k)(x)) & \text{iff } j \geq l, \end{cases}$$

where $r(I)$ denotes outward rounding of the interval I to the next-wider computer-representable interval. In the sequel, we will manipulate such eventually constant approximations T_l of interval traces and will represent them by the finite interval trace $tr_l : \{0, \ldots, l\} \to \mathcal{V}$ with $tr_l(j) = T_l(j)$ for each $j \leq l$.

Formally speaking, we call an infinite interval trace $T : \mathbb{N} \to \mathcal{V}$ *eventually constant* iff there exists $n \in \mathbb{N}$ such that $T(n) = T(m)$ for each $m > n$. The set $\mathcal{ECT}$ of eventually constant infinite interval traces can be identified with the set $\mathcal{FT} = \{tr : \{0, \ldots, n\} \to \mathcal{V} \mid n \in \mathbb{N}\}$ of finite interval traces via the homomorphism $\Phi : \mathcal{FT} \to \mathcal{ECT}$ defined as

$$\Phi(t)(n) = \begin{cases} t(n) & \text{if } n \in \mathrm{dom}(t)\,, \\ t(\max \mathrm{dom}(t)) & \text{otherwise.} \end{cases}$$

4.2 Evaluating the interval semantics over finite trace representations

It follows immediately from the definitions of the interval semantics that all LTL operators, including the temporally unbounded modalities $\Box$ and U, can be effectively evaluated on finite trace representations of eventually constant interval trace, as the following Lemma holds:

Lemma 4. *Let $t \in \mathcal{FT}$ and $i \in \mathbb{N}$. Then*

$$\Phi(t)[\![x > k]\!]\, i = \begin{cases} t(i)(x) - [k,k] & \textit{if } i \in \mathrm{dom}(t) \\ t(\max \mathrm{dom}(t))(x) - [k,k] & \textit{otherwise} \end{cases}$$

$$\Phi(t)[\![\neg\varphi]\!]\, i = -\Phi(t)[\![\varphi]\!]\, i$$

$$\Phi(t)[\![\varphi \wedge \psi]\!]\, i = \min(\Phi(t)[\![\varphi]\!]\, i, \Phi(t)[\![\psi]\!]\, i)$$

$$\Phi(t)[\![\bigcirc\varphi]\!]\, i = \Phi(t)[\![\varphi]\!]\, i + 1$$

$$\Phi(t)[\![\Box\varphi]\!]\, i = \min_{j \in \{i, \ldots, \max \mathrm{dom}(t)\}} \Phi(t)[\![\varphi]\!]\, j$$

$$\Phi(t)[\![\varphi \mathsf{U} \psi]\!]\, i = \max_{j \in \{i, \ldots, \max \mathrm{dom}(t)\}} \min(\Phi(t)[\![\psi]\!]\, j, \min_{k=i}^{j-1} \Phi(t)[\![\varphi]\!]\, k)$$

Thus, even unbounded future temporal modalities can be evaluated by visiting only a finite sample of time instants.

4.3 Refining a finitely represented interval trace

We will use Interval Constraint Propagation (ICP, cf. [BG06]) as well as measurements to refine interval traces and the value of the interval semantics (by the monotonicity Lemma 3) in an online fashion.

Interval constraint propagation is a proven technique in arithmetic constraint programming (for more information see `http://slash.math.unipd.it/cp/`). For the domain of the real numbers, given a constraint ϕ over real-valued variables $\{x_1, \ldots, x_n\}$ and a floating-point box $B \subseteq \mathbb{R}^n$, so-called *contractors* compute another floating-point box $C(\phi, B)$ such that $C(\phi, B) \subseteq B$ and such that $C(\phi, B)$ contains all solutions of ϕ in B, i.e. does chop off non-solutions only (cf. the notion of *narrowing operator* [BMVH94,Ben96]).

There are several methods for implementing such contractors. The most basic method [Dav87,Cle87] decomposes all atomic constraints (i.e., constraints of the form $t \geq 0$ or $t = 0$, where t is a complex-structured arithmetic term) into conjunctions of so-called primitive constraints resembling three-address code (i.e., constraints such as $x + y = z$, $xy = z$, $z \in [\underline{a}, \overline{a}]$, or $z \geq 0$) by introducing additional auxiliary variables (e.g., decomposing $x + \sin y \geq 0$ to $\sin y = v_1 \wedge x + v_1 = v_2 \wedge v_2 \geq 0$). Then it applies a contractor for these primitive constraints until coming close to a fixpoint, which shows by ceasing effectiveness of the narrowing process.

We illustrate contractors for primitive constraints using the example of a primitive constraint $x + y = z$ with the intervals $[1, 4]$, $[2, 3]$, and $[0, 5]$ for x, y, and z, respectively: We can solve the primitive constraint for each of the free variables, arriving at $x = z - y$, $y = z - x$, and $z = x + y$. Each of these forms allows us to contract the interval associated with the variable on the left-hand side of the equation: Using the first solved form we subtract the interval $[2, 3]$ for y from the interval $[0, 5]$ for z, concluding that x can only be in $[-3, 3]$. Intersecting this interval with the original interval $[1, 4]$, we know that x can only be in $[1, 3]$. Proceeding in a similar way for the solved form $y = z - x$ does not change any interval, and finally, using the solved form $z = x + y$, we can conclude that z can only be in $[3, 5]$. Contractors for other primitive constraints can be based on interval arithmetic in a similar way. There is extensive literature [Neu90,HJvE01] providing precise formulae for interval arithmetic for addition, subtraction, multiplication, division, and the most common transcendental functions. The floating point results are always rounded outwards, such that the result remains correct also under rounding errors.

There are several variants, alternatives and improvements of the approach described above (cf. [BG06] for a survey of the literature). These do in particular deal with stronger contractors based on non-decomposed constraints. All of them could easily be included into our approach, as the only property we need of the contractors is that they are safe in the sense of never dropping a solution through over-aggressive narrowing.

Using ICP for approximating ODE solutions. As has been observed by Hickey [HW04] as well as Ratschan and She [RS05,RS06], interval narrowing operators can be obtained directly from ordinary differential equations by means of

1. deriving necessary conditions for existence of a solution to the ODE from analytic properties of solutions like the mean-value theorem and
2. exploiting these necessary conditions, if they happen to be in or can be brought to existential first-order form, for interval narrowing by means of ICP.

In the sequel, we build on the mean-value condition of Ratschan and She, but modify it slightly for enhancing its reasoning power, especially in a sampled-time setting:[6] According to Ratschan and She, [RS05,RS06], a necessary first-order condition for connectedness of two points in the $\mathbb{R}^n$ via a continuous trajectory satisfying an ODE can be obtained as follows.

[6] A similar enhancement based on applying the mean-value theorem recursively for the continuous-time setting can be found in more recent versions of HSolver than those described in [RS05,RS06].

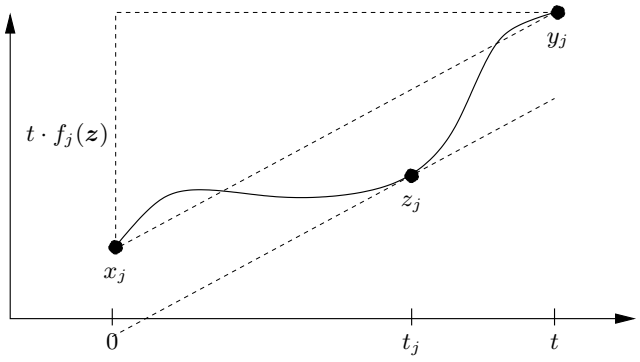

Fig. 2. The mean-value theorem.

Lemma 5. *[after Ratschan and She, 2005] Let $t > 0$, $I \subseteq \mathbb{R}^n$, $\boldsymbol{x} \in I$, and $\frac{\mathrm{d}\boldsymbol{z}}{\mathrm{d}t} = \boldsymbol{f}(\boldsymbol{z})$ be an ODE in $\boldsymbol{z}$. The existence of a solution $\boldsymbol{g} : [0,t] \to I$ of the ODE with $\boldsymbol{g}(0) = \boldsymbol{x}$ and $\boldsymbol{y} = \boldsymbol{g}(t)$ implies validity of*

$$\mathit{flow}(\boldsymbol{x},\boldsymbol{y},t) \equiv \bigwedge_{j=1}^{n} \exists \boldsymbol{z} \in I : y_j - x_j = t \cdot f_j(\boldsymbol{z})$$

Proof. This follows directly from the mean-value theorem of real-valued analysis, as illustrated in Fig. 2 (cf. [RS05,RS06] for details). □

Due to its role as a witness point in the mean-value theorem, each point $\boldsymbol{z}^j$ itself has to be visited by the trajectory. Therefore, above constraint can obviously be tightened to

Lemma 6. *Let $t > 0$, $I \subseteq \mathbb{R}^n$, $\boldsymbol{x} \in I$, and $\frac{\mathrm{d}\boldsymbol{z}}{\mathrm{d}t} = \boldsymbol{f}(\boldsymbol{z})$ be an ODE in $\boldsymbol{z}$. The existence of a solution $\boldsymbol{g} : [0,t] \to I$ of the ODE with $\boldsymbol{g}(0) = \boldsymbol{x}$ and $\boldsymbol{y} = \boldsymbol{g}(t)$ implies for each $i \geq 1$ that*

$$\mathit{Flow}^i(\boldsymbol{x},\boldsymbol{y},t) = \begin{cases} \bigwedge_{j=1}^{n} \exists \boldsymbol{z} \in I : & y_j - x_j = t \cdot f_j(\boldsymbol{z}) & \text{if } i = 1, \\ \bigwedge_{j=1}^{n} \exists t_j \in \mathbb{R}, \boldsymbol{z} \in I : & \begin{pmatrix} 0 < t_j < t \\ \wedge\ y - x = t \cdot f_j(\boldsymbol{z}) \\ \wedge\ \mathit{Flow}^{i-1}(\boldsymbol{x},\boldsymbol{z},t_j) \\ \wedge\ \mathit{Flow}^{i-1}(\boldsymbol{z},\boldsymbol{y},t-t_j) \end{pmatrix} & \text{if } i > 1. \end{cases}$$

holds.

Note that $\mathit{Flow}^1(\boldsymbol{x},\boldsymbol{y},t)$ is equivalent to $\mathit{flow}(\boldsymbol{x},\boldsymbol{y},t)$ while Flow^i for $i \geq 2$ is stronger, in particular if t is a constant as in a sampled-time regime.

In the sequel, we will exploit $\mathit{Flow}^2(\boldsymbol{x},\boldsymbol{y},\Delta t)$, where Δt is the sampling time, for interval narrowing by ICP. To illustrate its effect, and to show the substantial difference to the weaker $\mathit{flow}(\boldsymbol{x},\boldsymbol{y},\Delta t)$, we demonstrate it on the ODE

$$x_1 = -x_3 \ , \tag{1}$$

$$\frac{\mathrm{d}x_2}{\mathrm{d}t} = x_1 \ , \tag{2}$$

$$\frac{\mathrm{d}x_3}{\mathrm{d}t} = x_2 \tag{3}$$

of a harmonic oscillator. Given $I = [-100,100]^3$, $\Delta t = 0.1$, and interval bounds $x_2 \in [0,0]$, $x_3 \in [49,51]$ on $\boldsymbol{x}$, we obtain the following constraint systems and corresponding narrowing sequences: Using $\mathit{flow}(\boldsymbol{x},\boldsymbol{y},\Delta t)$ as a constraint, we obtain (after eliminating the existential quantifier and

dropping unused variables) the constraint system.

$$x_1 = -x_3 \quad \text{from (1)} \tag{4}$$
$$y_1 = -y_3 \quad \text{from (1)} \tag{5}$$
$$y_2 - x_2 = 0.1z_1 \quad \text{from (2)} \tag{6}$$
$$y_3 - x_3 = 0.1z_2 \quad \text{from (3)} \tag{7}$$
$$z_1 \in [-100, 100]^3 \quad \text{from the box bounds} \tag{8}$$
$$z_2 \in [-100, 100]^3 \quad \text{from the box bounds} \tag{9}$$

This yields the maximal reduction sequence

Step	0	1	2	3
Reason		(4, 8, 9)	(6, 7)	(5)
x_1	$[-100, 100]$	$[-51, -49]$		
x_2	$[0, 0]$			
x_3	$[49, 51]$			
z_1^2		$[-100, 100]$		
z_2^3		$[-100, 100]$		
y_1	$[-100, 100]$			$[-61, -39]$
y_2	$[-100, 100]$		$[-10, 10]$	
y_3	$[-100, 100]$		$[39, 61]$	

If instead using $Flow^2(\boldsymbol{x}, \boldsymbol{y}, \Delta t)$ as a constraint, we get the constraint system

The constraint system using Lemma 6

From (1):

$$x_1 = -x_3 \tag{10}$$
$$y_1 = -y_3 \tag{11}$$
$$z_{1,1} = -z_{3,1} \tag{12}$$

$L : Flow^2(\boldsymbol{x}, \boldsymbol{z}, t)$

$$y_2 - x_2 = 0.1z_{1,1} \quad \text{from (2)} \tag{13}$$
$$y_3 - x_3 = 0.1z_{2,1} \quad \text{from (3)} \tag{14}$$

From the box bounds of I:

$$z_{1,1} \in [-100, 100]^3 \tag{15}$$
$$z_{2,1} \in [-100, 100]^3 \tag{16}$$
$$z_{3,1} \in [-100, 100]^3 \tag{17}$$

$L : Flow^1(\boldsymbol{x}, \boldsymbol{z}, t_j)$	$R : Flow^1(\boldsymbol{z}, \boldsymbol{y}, t - t_j)$
From (1):	From (1):
(18) $z_{1,2\cdot L} = -z_{3,2\cdot L}$	(19) $z_{1,2\cdot R} = -z_{3,2\cdot R}$

$L : Flow^1(\boldsymbol{x}, \boldsymbol{z}, t_j)$		$R : Flow^1(\boldsymbol{z}, \boldsymbol{y}, t - t_j)$	
From (2):		From (2):	
(20)	$z_{2,1} - x_2 = t_{2\cdot L} z_{1,2\cdot L}$	(26)	$y_2 - z_{2,1} = t_{2\cdot R} z_{1,2\cdot R}$
From (3):		From (3):	
(21)	$z_{3,1} - x_3 = t_{2\cdot L} z_{2,2\cdot L}$	(27)	$y_3 - z_{3,1} = t_{2\cdot R} z_{2,2\cdot R}$
From the box bounds of I:		From the box bounds of I:	
(22)	$z_{1,2\cdot L} \in [-100, 100]^3$	(28)	$z_{1,2\cdot R} \in [-100, 100]^3$
(23)	$z_{2,2\cdot L} \in [-100, 100]^3$	(29)	$z_{2,2\cdot R} \in [-100, 100]^3$
(24)	$z_{3,2\cdot L} \in [-100, 100]^3$	(30)	$z_{3,2\cdot R} \in [-100, 100]^3$
(25)	$t_{2\cdot L} := t_j$	(31)	$t_{2\cdot R} := t - t_j$

instead (again after quantifier elimination and appropriate renaming). It is easy to see that this constraint set, when starting from the same interval valuation $x_1 = [-100, 100]$, $x_2 = [0, 0]$, $x_3 = [49, 51]$, and $y_1 = y_2 = y_3 = [-100, 100]$, narrows this down to the considerably tighter approximation

Step	0	1	2	3	4	5
Reason	init	$(10, 20, 21)$	(12)	$(13, 14)$	(11)	R *analog* : $(11, 27)$
x_1	$[-100, 100]$	$[-51, -49]$				
x_2	$[0, 0]$					
x_3	$[49, 51]$					
$z_{1,1}$	$[-100, 100]$		$]-61, -39[$			
$z_{2,1}$	$[-100, 100]$	$]-10, 10[$				
$z_{3,1}$	$[-100, 100]$	$]39, 61[$				
$z_{1,2\cdot L}$	$[-100, 100]$					
$z_{2,2\cdot L}$	$[-100, 100]$					
$z_{3,2\cdot L}$	$[-100, 100]$					
y_1	$[-100, 100]$				$]-52, -48[$	$]-51.61, -48[$
y_2	$[-100, 100]$			$]-6.1, -3.9[$		
y_3	$[-100, 100]$			$]48, 52[$		$]48, 51.61[$
$t_{2\cdot L}$	$]0, 0.1[$					

Refining the collecting interval. The idea of the collecting interval "terminating" the finite sequence of intervals representing the infinite trace is that this collecting interval covers all variable values that could eventually be reached from the last concrete interval. Computing such a reach set online is infeasible due to the computational cost. Instead, we reduce online computational cost by doing as much of the computation as possible beforehand in an offline fashion. In the offline process, we take advantage of existing reachability checkers for hybrid systems, like PHaver [Fre05] or HSolver [RS05].

Given that we only want to be able to predict whether some atomic predicates occurring in the LTL formula may ever become true, we proceed as follows. In the *offline* pre-processing, we take the reachability tools off-the-shelf and do compute a finite overapproximation of the transitive closure of the backward reachability relation:

1. Given the set P of atomic predicates occurring in the LTL formula ϕ, we partition $\mathcal{V}$ into the $2^{|P|}$ equivalence classes $V_1, \ldots, V_{2^{|P|}}$ spanned by these predicates.
2. For each equivalence class $V_i \in \{V_1, \ldots, V_{2^{|P|}}\}$, we compute and memorize a safe overapproximation R_i of its backward reach set using an off-the-shelf tool like —depending on the class of system— PHaver or HSolver.

Based on this, we use the following *online* algorithm for computing an adequate collecting interval terminating a finite interval trace:

3. Given a finite interval trace $tr : \{0, \ldots, l\} \to \mathcal{V}$ consisting of concrete intervals (in the sense of each $tr(n)$ representing just the possible values at time point n) $tr(0)$ to $tr(l-1)$ and some collecting interval $tr(l)$, we refine the collecting interval consistent with the dynamics as follows. We compute an overapproximation of the state set reachable form the last concrete interval valuation $tr(l-1)$ as

$$v_\infty = r(\bigcup_{i \in \{1,\ldots,2^{|P|} \mid R_i \cup tr(l-1) \neq \emptyset\}} V_i \cap tr(l)) \ ,$$

where r again represents interval closure wrt. machine-representable intervals. Then we replace tr by its refinement $tr' : \{0, \ldots, l\} \to \mathcal{V}$ with

$$tr'(n) = \begin{cases} tr(n) & \text{if } n \leq l, \\ v_\infty & \text{if } n = l. \end{cases}$$

Refining a finite representation of a infinite interval trace. Based on this, we use the following online algorithm for refining the finitely represented interval trace:

1. Initially, we start with the finite interval trace $tr_0 = \{0\} \to \mathcal{V}$ of length 1 given by

$$tr_0(i)(o) = [l_o, u_o] \text{ for each } o \in \mathcal{O} \text{ and } i = 0.$$

2. Whenever computation time is available, we do an online refinement of the finitely represented interval trace tr by one of the following refinement steps:
   (a) Take some $i < \max \mathrm{dom}(tr) - 1$ and apply ICP wrt. constraint $Flow^j(\mathcal{O}, \mathcal{O}', t)$, where t is the sampling period, to the pair $(tr(i), tr(i+1))$ to narrow the trace, obtaining a refinement tr' of tr.
   (b) We narrow the collecting interval $tr(\max \mathrm{dom}(tr))$ by replacing it with $v_\infty = r(\bigcup_{i \in \{1,\ldots,2^{|P|} \mid R_i \cup tr(\max \mathrm{dom}(tr)-1) \neq \emptyset\}} V_i \cap tr(\max \mathrm{dom}(tr)))$, obtaining a refinement tr' of tr.
   (c) Apply ICP wrt. constraint $Flow^j(tr(l-1), tr(l), t) \vee Flow^j(tr(l), tr(l), t)$, where $l = \max \mathrm{dom}(tr)$ and t is the sampling period, to the pair $(tr(l-1), tr(l))$ to narrow the trace, obtaining a refinement tr' of tr.
   (d) Select $l' > \max \mathrm{dom}(tr)$ and replace tr by its (non-strict) refinement $tr' : \{0, \ldots, l'\} \to \mathcal{V}$ of length l' defined by

$$tr'(j) = \begin{cases} tr(j) & \text{if } j \leq \max \mathrm{dom}(tr), \\ tr(\max \mathrm{dom}(tr)) & \text{if } \max \mathrm{dom}(tr) \leq j \leq l', \\ \text{undefined} & \text{otherwise.} \end{cases}$$

3. Whenever a new measurement $m \in [l_o, u_o]$ for observable o with associated time stamp $k \in \mathbb{N}$ arrives, we refine the finite interval trace tr to

$$tr'(j)(x) = \begin{cases} tr(i)(x) & \text{iff } i \neq k \text{ or } x \neq o, \\ [m, m] & \mathit{otherwise}, \end{cases}$$

provided $k < l$. If $k \geq l$, we do first extend the finite representation by rule (2d) such that $k < l$.

As all these refinements preserve overapproximations of the traces consistent with the dynamics, they can be applied in an online fashion at any time and in any sequence whenever time permits.

5 A scenario case study using ICP approximations

The subject of the case study introduced in Section 2 is the determination of the truth value of the formula based on safely predicting the evolution of the observable state variables into the future segment. While the exact positions of the vehicles within this segment are unknown, a safe interval of possible positions at a given instant can be computed by exploiting knowledge about the car dynamics, like the differential equations from Newtonian mechanics plus suitable bounds on the velocity and the acceleration. Using the mean-value form from Section 4, we can use ICP for such a safe extrapolation. In the sequel, we employ ICP on the mean-value forms $Flow^1$ and $Flow^2$, resp., of the given ODEs $v = \frac{\mathrm{d}s}{\mathrm{d}t}$ and $a = \frac{\mathrm{d}v}{\mathrm{d}t}$, and compare the results obtained. In this benchmark, the computation was performed by a stripped-down verion of the iSAT tool [FHT+07,TF08]. iSAT originally is a tight integration of the Davis-Putnam-Loveland-Logeman algorithm for SAT solving [DLL62] algorithm with interval constraint propagation enriched by enhancements like conflict-driven clause learning and nonchronological backtracking. For the sake of our benchmarks, we have stripped the DPLL branching rules, confining the reasoning to branching-free inference sequences resorting to ICP and unit propagation only. The effect is a considerably reduced variability in runtime, making the tool suitable for online prediction, however at the price of reduced reasoning power compared to the original iSAT.

Within the benchmark scenario, the initial positions $EgoCar_p \in [0, 400]m$ and $LeadCar_p \in [0, 400]m$ of the $EgoCar$ and the $LeadCar$ and their initial velocities with $EgoCar_v \in [0, 55]\frac{m}{s}$, $LeadCar_v \in [0, 50]\frac{m}{s}$ are given, while the latter evolution of these observables is to be extrapolated. The interweaving lane is defined by its beginning $a = 0m$ and its end point $b = 200m$, such that a vehicle has passed the interweaving lane when $L_l > b$ or $E_l > b$, respectively. We perform our experiments with a sampling step (also preserved within the extrapolatioon) of $\Delta t = 0.1s$. Instantiating the $flow = Flow^1$ constraint from Lemma 5 for the given ODEs, the following constraint system characterizing the dynamics of the traffic agents is obtained:

$$EgoCar'_p - EgoCar_p = EgoCar_{v,z} * \Delta t \tag{32}$$

$$EgoCar'_v - EgoCar_v = EgoCar_{a,z} * \Delta t \tag{33}$$

$$LeadCar'_p - LeadCar_p = LeadCar_{v,z} * \Delta t \tag{34}$$

$$LeadCar'_v - LeadCar_v = LeadCar_{a,z} * \Delta t \tag{35}$$

$$EgoCar_{a,z} \in [-11, 5]\frac{m}{s^2} \ , \ LeadCar_{a,z} \in [-9, 3]\frac{m}{s^2} \tag{36}$$

$$EgoCar_{v,z} \in [0, 55]\frac{m}{s} \ , \ LeadCar_{v,z} \in [0, 50]\frac{m}{s} \tag{37}$$

Given the initial positions $EgoCar_p = 65m$ and $LeadCar_p = 157m$ plus the initial velocities $EgoCar_v = 20\frac{m}{s}$ and $LeadCar_v = 29\frac{m}{s}$, the iSAT tool extrapolates the future intervals by UCP as depicted in Fig. 3. The propagation was performed on a 2.5 GHz Intel Core 2 Duo CPU with 2 GByte physical memory, running Linux, with an overall solving time t_{solve} of $0.39s$ for 28 time steps. The figure shows that within the observed time period, $L_l < b$ holds throughout. This implies that the $LeadCar$ cannot be guaranteed to leave the interweaving lane before the end of the time frame (actually independently of the length of the time frame). Furthermore, for $t \geq 1.7$ the $EgoCar$ has position intervals that intersect with those of the lead car such that this extrapolation cannot exclude collisions.

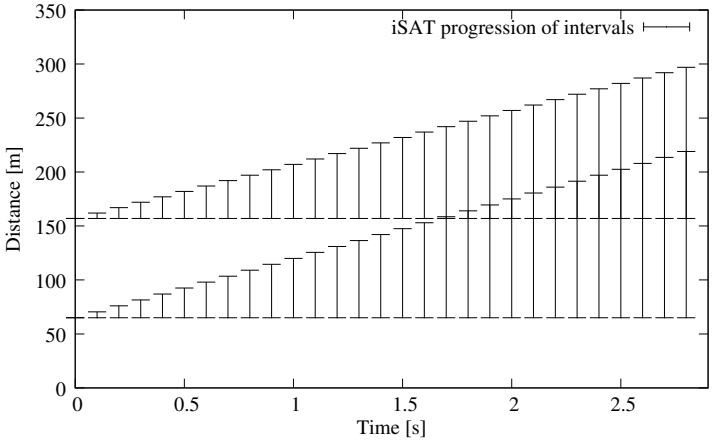

Fig. 3. The predicted trajectory of the traffic agents using $Flow^1$

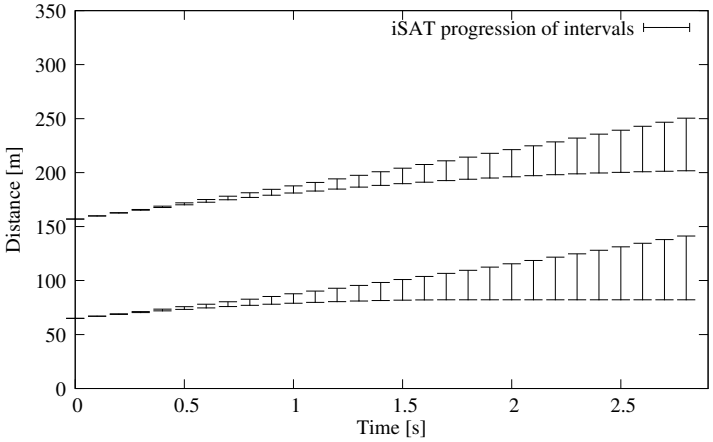

Fig. 4. The predicted trajectory of the traffic agents using $Flow^2$

Using Lemma 6 with $i = 2$ instead, the additional constraints (38) to (44) below arise, yielding the overall constraint system (32) to (44):

$$EgoCar_{p,z} - EgoCar_p = EgoCar_{v,z2\cdot L} * \Delta t_{2\cdot L} \tag{38}$$

$$EgoCar_{v,z} - EgoCar_v = EgoCar_{a,z2\cdot L} * \Delta t_{2\cdot L} \tag{39}$$

$$LeadCar_{p,z} - LeadCar_p = LeadCar_{v,z2\cdot L} * \Delta t_{2\cdot L} \tag{40}$$

$$LeadCar_{v,z} - LeadCar_v = LeadCar_{a,z2\cdot L} * \Delta t_{2\cdot L} \tag{41}$$

$$EgoCar_{a,z2\cdot L} \in [-11, 5]\frac{m}{s^2} \ , \ LeadCar_{a,z2\cdot L} \in [-9, 3]\frac{m}{s^2} \tag{42}$$

$$EgoCar_{v,z2\cdot L} \in [0, 55]\frac{m}{s} \ , \ LeadCar_{v,z2\cdot L} \in [0, 50]\frac{m}{s} \tag{43}$$

$$\Delta t_{2\cdot L} \in]0, 0.1[s \tag{44}$$

Using identical parameter as before, the additional constraints provide for a much more effective constraint propagation, yielding the intervals depicted in Fig. 3 within a solving time $t_{solve} = 0.95s$ for 28 extrapolation time steps. The tighter intervals obtained are immediately visible from the figures. Note that this time, the extrapolated intervals are sufficient for safely deciding absence of collision, as the lead car is guaranteed to be beyond the interweaving lane for $t \geq 2.5$ and as the position intervals of lead car and ego car do not intersect for $0 \leq t < 2.5$.

Finally, it is interesting to study the relationship between solving time and the quality of extrapolation obtained, as this is a crucial measure of suitability for online applications. Aiming at

Δt	Steps	L_l	t_{solve}
0.05	60	202.825	4.63
0.1	30	202.15	1.16
0.2	15	200.8	0.27
0.3	10	199.45	0.12
0.5	6	196.75	0.05

Table 1. Effect of step size Δt on the runtime t_{solve} of the extrapolation algorithm and on the quality of the result.

Δt	Steps	$L_l > b$	$t_{L_l>b}$	t_{solve}
0.05	27	200.645	1.35	0.98
0.1	14	201.15	1.4	0.26
0.2	7	200.52	1.4	0.07
0.3	5	201.35	1.5	0.04
0.4	4	202	1.6	0.02
0.5	3	200	1.5	0.01

Table 2. Effect of step size Δt on the number of extrapolation steps needed for guaranteeing that lead car has left interweaving lane, plus extrapolated time $t_{L_l>b}$ of leave and corresponding computation time for extrapolation.

extrapolation over a time frame of $t = 3s$, we do in Table 1 compare the runtimes and the lower bound on the lead car position obtained for different step sizes, i.e. different sampling rates. The initial conditions used throughout this experiment are as before. As expected, the performance gain achieved by enlarging Δt is bought at the cost of more conservative approximations. Table 2 investigates this issue more closely by stating the computation times needed for obtaining a certificate of leave of the lead car rather than the computation times for a constant duration of the extrapolation.

6 Conclusion

The contribution of this paper is twofold. On the one hand, we have presented a definition of a notion of a robust interval semantic of LTL formulae which permits interpretation of unbounded future operators over finite interval-state sequences terminated by an collecting interval representing the states reachable in future. On the other hand, we have provided algorithms for safely obtaining such finite interval-trace representations from finite samples of an observed system. The algorithms perform both a relatively accurate approximation into the finite future and the termination of the trace with an collecting interval. Being based on interval constraint propagation, the computational effort is low enough to facilitate online prediction while monitoring systems at moderate sampling frequencies in the range of some Hertz.

In a range of practical situations, this solves the problem with online monitoring of LTL noted in [MNP08]: a major difficulty in checking properties expressed in future LTL is due to the non-causal definition of the satisfaction relation, where satisfaction of ϕ at time t may depend on the value of ξ at some future time instant $t' \geq t$. Our semantics permits not only the direct accommodation of inexact measurements in online monitoring, but does furthermore capture and continuously react to the time-wise monotonic increase in information pertinent to online monitoring and thus in certainty about the requirements monitored: For all interesting formulae, i.e. all formulae being neither tautologic nor unsatisfiable, the "truth interval" reported by the monitor starts with a large interval covering both positive and negative values. In the course of monitoring, this truth interval will be monotonically refined upon advent of every new measurement until it finally may fall completely into the positive range (or, alternatively, fully into the negative), which implies that every possible continuation of the observed time series robustly satisfies (violates, resp.) the formula with a robustness margin at least equalling the infimum (supremum, resp.) of the reported interval. Whenever that situation arises, the satisfaction of ϕ at time t has been safely determined despite possible references within ϕ to yet unknown future events. From the users perspective, the corresponding interval refinements signal a natural sequence of states of a monitored requirement, namely starting undecided, then turning critically sensitive to the next measurements coming in, and finally being decided.

References

[AFH96] R. Alur, T. Feder, and T.A. Henzinger. The benefits of relaxing punctuality. *Journal of the ACM*, 43:116–146, 1996.

[Ben96] F. Benhamou. Heterogeneous constraint solving. In *Proc. of the Fifth International Conf. on Algebraic and Logic Programming*, volume 1139 of *LNCS*. Springer, 1996.

[BG06] F. Benhamou and L. Granvilliers. Continuous and interval constraints. In F. Rossi, P. van Beek, and T. Walsh, editors, *Handbook of Constraint Programming*, Foundations of Artificial Intelligence, chapter 16, pages 571–603. Elsevier, Amsterdam, 2006.

[BMVH94] F. Benhamou, D. McAllester, and Pascal Van Hentenryck. CLP(Intervals) revisited. In *Int. Symp. on Logic Progr.*, pages 124–138, Ithaca, NY, USA, 1994. MIT Press.

[Cle87] J. G. Cleary. Logical arithmetic. *Future Computing Systems*, 2(2):125–149, 1987.

[Dav87] E. Davis. Constraint propagation with interval labels. *Artif. Intell.*, 32(3):281–331, 1987.

[DLL62] M. Davis, G. Logemann, and D. Loveland. A machine program for theorem proving. *Communications of the ACM*, 5:394–397, 1962.

[FH05] M. Fränzle and M. R. Hansen. A robust interpretation of duration calculus. In Dang Van Hung and Martin Wirsing, editors, *Theoretical Aspects of Computing - ICTAC 2005*, volume 3722 of *LNCS*, pages 257–271. Springer-Verlag, 2005.

[FHT+07] M. Fränzle, C. Herde, T. Teige, S. Ratschan, and T. Schubert. Efficient Solving of Large Non-linear Arithmetic Constraint Systems with Complex Boolean Structure. *Journal on Satisfiability, Boolean Modeling and Computation – Special Issue on SAT/CP Integration*, 1:209–236, 2007.

[FLS08] M. Faella, A. Legay, and M.I.A. Stoelinga. Model checking quantitative linear time logic. In A. Aldini and C.Baier, editors, *Proceedings of the Sixth Workshop on Quantitative Aspects of Programming Languages (QAPL'08)*, ENTCS, 2008. to appear.

[Fre05] Goran Frehse. Phaver: Algorithmic verification of hybrid systems past hytech. In Morari and Thiele [MT05], pages 258–273.

[Gez09] Tayfun Gezgin. Observerbasierte on-the-fly Auswertung von QLTL-Formeln innerhalb eines HLA-Simulationsverbunds. Diplomarbeit, Carl von Ossietzky Universität Oldenburg, September 2009. To appear.

[HJvE01] T.J. Hickey, Qun Ju, and M.H. van Emden. Interval arithmetic: from principles to implementation. *Journal of the ACM*, 48(5):1038–1068, 2001.

[HW04] Timothy Hickey and David Wittenberg. Rigorous modeling of hybrid systems using interval arithmetic constraints. In Rajeev Alur and George J. Pappas, editors, *HSCC'04*, volume 2993 of *LNCS*, pages 402–416. Springer, 2004.

[MN04] Oded Maler and Dejan Nickovic. Monitoring temporal properties of continuous signals. In Yassine Lakhnech and Sergio Yovine, editors, *Proceedings of the Joint International Conferences on Formal Modelling and Analysis of Timed Systems, (FORMATS'04) and Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT'04)*, volume 3253 of *Lecture Notes in Computer Science*, pages 152–166. Springer-Verlag, September 2004.

[MNP08] Oded Maler, Dejan Nickovic, and Amir Pnueli. Checking temporal properties of discrete, timed and continuous behaviors. In Arnon Avron, Nachum Dershowitz, and Alexander Rabinovich, editors, *Pillars of Computer Science*, volume 4800 of *Lecture Notes in Computer Science*, pages 475–505. Springer, 2008.

[MT05] M. Morari and L. Thiele, editors. *Hybrid Systems: Computation and Control*, volume 3414 of *LNCS*. Springer, 2005.

[Neu90] A. Neumaier. *Interval Methods for Systems of Equations*. Cambridge Univ. Press, Cambridge, 1990.

[NM07] Dejan Nickovic and Oded Maler. AMT: A property-based monitoring tool for analog systems. In Jean-Francois Raskin and P. S. Thiagarajan, editors, *FORMATS*, volume 4763 of *Lecture Notes in Computer Science*, pages 304–319. Springer, 2007.

[Rat00] Stefan Ratschan. Uncertainty propagation in heterogeneous algebras for approximate quantified constraint solving. *Journal of Universal Computer Science*, 6(9), 2000.

[RS05] Stefan Ratschan and Zhikun She. Safety verification of hybrid systems by constraint propagation based abstraction refinement. In Morari and Thiele [MT05], pages 573–589.

[RS06] Stefan Ratschan and Zhikun She. Constraints for continuous reachability in the verification of hybrid systems. In *Proc. 8th Int. Conf. on Artif. Intell. and Symb. Comp., AISC'2006*, number 4120 in LNCS. Springer, 2006.

[TF08] T. Teige and M. Fränzle. Stochastic Satisfiability modulo Theories for Non-linear Arithmetic. In Laurent Perron and Michael A. Trick, editors, *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, 5th International Conference, CPAIOR 2008*, volume 5015 of *Lecture Notes in Computer Science*. Springer, 2008.