Robustness Analysis for Value-Freezing Signal Temporal Logic

L. Brim, T. Vejpustek, D. Šafránek, and J. Fabriková [*]

Faculty of Informatics
Masaryk University
Botanická 68a, Brno, Czech Republic
safranek@fi.muni.cz

In our previous work we have introduced the logic STL*, an extension of Signal Temporal Logic (STL) that allows value freezing. In this paper, we define robustness measures for STL* by adapting the robustness measures previously introduced for Metric Temporal Logic (MTL). Furthermore, we present an algorithm for STL* robustness computation, which is implemented in the tool Parasim. Application of STL* robustness analysis is demonstrated on case studies.

1 Introduction

A particular place among formalisms adopted by systems biology is occupied by temporal logics, which serve as a language for description of biological systems behaviour. Resulting temporal formulae can be used during computer-aided system analysis, such as model checking [5], which automatically verifies whether a model satisfies given temporal formula. Methods based on temporal logics have been successfully employed to study biological phenomena [28, 25, 16] (see [3] for review).

Since most of current models developed in computational systems biology have the form of ordinary differential equations, model checking cannot be directly employed and is typically replaced with a non-exhaustive procedure of monitoring [24]. In this setting, a (finite) set of signals representing individual time-courses of the model is monitored wrt a given temporal specification. In particular, the respective temporal logics are interpreted over individual signals that are most typically simplified to discrete timed state sequences (time series) approximating the continuous trajectories by means of numerical simulation. Temporal logics fitting this interpretation are Metric Temporal Logic (MTL) [21] and Signal Temporal Logic (STL) [24], which allow quantifying modalities with the time frame represented by a closed time interval. MTL possesses both discrete and continuous semantics, as it can be interpreted over both infinite timed state sequences and continuous signals. STL is practically focused and is defined for piece-wise linear approximations of continuous signals.

Temporal logics are satisfactorily used in systems biology to express statements about a single instance of system behaviour such as *in five minutes, concentration of glucose will be greater than 0.8*. However, many biological hypotheses contain relative temporal references, e.g., *after protein P reaches the maximum concentration, a steady concentration of P is reached which is less than half of the maximum*. Such a scenario can be found, e.g., in feed-forward genetic regulatory circuits generating pulses in expression signals [18]. In common temporal logics, such a general query cannot be expressed. This is because the values in different time points cannot be compared, i.e., the property *in five minutes, concentration of glucose will rise by 0.2*, which relates glucose concentration at current time and in the

[*] The work has been supported by the Grant Agency of Czech Republic grant GAP202/11/0312 and by the EC OP project No. CZ.1.07/2.3.00/20.0256.

T. Dang and C. Piazza (Eds.): HSB 2013
EPTCS 125, 2013, pp. 20–36, doi:10.4204/EPTCS.125.2

future, cannot be specified. Of specific interest is oscillatory behaviour, e.g., a sequence of gradually increasing peaks followed by a limit cycle with a stable amplitude [15]. In order to express the increasing amplitude, it is necessary to detect local extremes in signals and compare respective signal values. This cannot be achieved using common temporal logics. Signals with a series of increasing local maxima have been observed, e.g., in response of FGF signalling pathways transferring stimuli from mutated FGFR3 receptors to target effectors affecting bone cells growth [22]. Since the mentioned behaviour correlates with the phenotype of dysplasia, it is necessary to develop models that mechanistically capture the respective signalling pathways and to analyse circumstances under which the undesired behaviour occurs. This makes a necessary step before designing a targeted medical treatment. To this end, temporal logics and verification procedures which allow to capture and analyse such complex phenotypes have to be developed.

In [7], we have introduced a new temporal logic STL* which alleviates limitations mentioned above. Expressiveness of STL* is enhanced by signal-value freeze operator which stores values at certain time, which may be referred to in the future. This allows STL* to specify and distinguish various dynamic aspects which occur in biological systems, in addition to the phenomena mentioned above, these can be, e.g., damped oscillations [17] or local extremes in species concentration. It is worth noting that some more complex queries can be expressed in traditional temporal logic by including signal derivatives into atomic propositions. However, this does not directly apply to queries mentioned above. One can express the presence and shape of a local extreme by using the first and second derivative, but still the values in particular time points have to be compared in order to express the complex queries.

An important concept associated with biological systems and temporal logics is *robustness*, the ability of a system to maintain its function against perturbations [20]. Since system function can be expressed in the terms of temporal logic, we speak of robustness with respect to a temporal logic formula, which can be quantified and computed [14, 26]. Robustness significantly enhances model analysis and gives an optimization goal for model parameter estimation/synthesis [11, 9, 27].

This paper introduces the notion of robustness in the value-freezing logic STL* setting. In particular, we extend the continuous and discrete measure defined for MTL by Fainekos et al. [14] to the semantic domain of STL*. Robustness of the input signal with respect to STL* formula delineates the robust neighbourhood of the signal (the maximal "tube" around the signal where the formula is satisfied). The robustness measure we propose (Section 3) is defined inductively wrt the formula structure and is based on a distance metrics employed on the signal domain extended with (multiple) dimensions representing the frozen time points. The theoretical framework is computationally supported with an algorithm based on solving the optimization problem (Section 4) provided that the logic is restricted to linear predicates. Special consideration is given to optimization of the formula to overcome unnecessary computational overhead.

Implementation of our algorithm is included as a part of Parasim [12], a tool aimed as a modular environment for monitoring and robustness analysis of kinetic models. To demonstrate the usage and evaluate the performance, we present case studies of two simple kinetic models (Section 5).

1.1 Related Work

Robustness measures have been defined for three temporal logics targeting deterministic continuous systems: STL [11], MTL [14] and QFLTL [26]. We adopt the concept of behaviour-based robustness introduced on a fragment of MTL by Fainekos et al. [14], who define robustness measure for MTL formulae with discrete [13] and continuous [14] semantics. In [14], Fainekos et. al prove a theorem connecting discrete and continuous robustness, which is valuable for robustness computation. A recent

tool [2] implements the method. Donzé et al. [24] use STL to define a distinct robustness measure, albeit constructed from [14], and propose its application for space exploration [11, 9], which was implemented in the Breach Toolbox [8]. The work is further improved from the computational point of view in [10]. Our implementation (Parasim) is based on a simplified version of the robustness analysis algorithm for STL where the sensitivity-based computation of local robustness is replaced with direct computation of trajectories distance. The extension for STL* as presented in Section 4 is implemented in this setting.

Fages et al. [26] introduced property-based approach to robustness that fixes input behaviour and examines the formula. Basically, it measures the extent to which the formula can be modified while preserving its satisfaction. The tool BioCham implements this idea [4]. Extended LTL logic with constraints over real numbers (quantifier-free LTL) is employed being defined for finite discrete time-series.

It is worth noting that the problem of formula satisfiability is undecidable for MTL [21]. To achieve decidability, Alur and Henzinger specified further conditions on intervals associated with temporal operators [1]. The result, metric interval temporal logic, requires all intervals to be non-singular and is interpreted over timed state-sequences where time points are replaced with consecutive time intervals. STL was introduced by Maler and Nickovic in [24] as a basis for their monitoring procedure. Technically, it comprises a variant of MITL interpreted over real signals. Because of its practical purpose, in [7] we selected STL as a good candidate for extension with value-freezing.

2 Background

STL* is evaluated over finite time continuous signals (finite signals for short).

Definition 2.1 *Let $n \in \mathbb{N}$ and $T = [0, r]$ where $r \in \mathbb{R}^+$. Then $s : T \to \mathbb{R}^n$ is a* bounded continuous-time signal *and T its* time domain. *We denote $l(s) = r$ the length of signal s.*

Signal value freezing is facilitated by the following structure which is used to store time values at various time points which then can be referred to in predicates.

Definition 2.2 *Let $\mathscr{I}$ be a finite index set.* Frozen time vector *is a function:*

$$t^* : \mathscr{I} \to \mathbb{R}_0^+$$

The symbol $t_i^* = t^*(i)$ is referred to as i-th frozen time. For convenience reasons and without loss of generality, we will henceforth assume that an index set $\mathscr{I} = \{1, \ldots, m\}$ is given, where $m \in \mathbb{N}$.

Predicates comprise Boolean expressions over values of a signal s at time t and each frozen time t_i^*, where x_j denotes the j-the component of the signal at time t, i.e. $s(t) = (x_1, \ldots, x_j, \ldots, x_n)$, and x_j^{*i} the j-th component at time t_i^*. When $|\mathscr{I}| = 1$, we usually omit the index of asterisk, e.g. $x_i^* = x_i^{*1}$.

We consider only predicates given by linear inequalities, so that analytic expressions of predicate robustness is possible.

Definition 2.3 *Let $n \in \mathbb{N}$, $b \in \mathbb{R}$ and $a_{ij} \in \mathbb{R}$ where $i \in \{0\} \cup \mathscr{I}$, $j \in \{1, \ldots, n\}$ and not all a_{ij} are zero. A* predicate *is defined as a subset of $\mathbb{R}^n \times (\mathbb{R}^n)^{\mathscr{I}}$ such that:*

$$\sum_{j=1}^{n} a_{0j} x_j + \sum_{i=1}^{|\mathscr{I}|} \sum_{j=1}^{n} a_{ij} x_j^{*i} + b \geq 0$$

Predicates are specified by the set of associated coefficients a_{ij}, b (where coefficients a_{0j} are connected with the current time t). Therefore, for convenience reasons, we will use these coefficients to represent predicates. Predicates with all coefficients a_{ij} zero were omitted since they are of the form $b \geq 0$ and, therefore, trivially true or false.

Predicates with equality (i.e. having $=$ in place of $\geq$), although theoretically possible, lack practical value, as they are not robust (small perturbation may invalidate the property). This has been already argued in [7], albeit without defining the concept of robustness. Since robustness of predicates with strict and non-strict inequalities does not differ, we consider only non-strict inequalities.

Freeze operator is used to store the time point into frozen time vector, thus facilitating signal value freezing. The following definition introduces an auxiliary concept of storing the current time t as the ith component of the frozen time vector.

Definition 2.4 *Let t^* be frozen time vector, $i, j \in \mathscr{I}$ and $t \in \mathbb{R}_0^+$.* Freezing ith component of t^* in t *is denoted as $t^*[i \leftarrow t]$ and defined:*

$$t^*[i \leftarrow t](j) = \begin{cases} t & i = j \\ t_j^* & i \neq j \end{cases}$$

Definition 2.5 *Syntax of STL* is defined by the following grammar:*

$$\varphi ::= \mu \mid \top \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \boldsymbol{U}_I \varphi_2 \mid *_i \varphi$$

where $i \in \mathscr{I}$, $\top$ denotes the true constant, μ is a predicate as of Definition 2.3 and $I \subseteq \mathbb{R}_0^+$ a closed non-singular interval.

Note that all Boolean connectives and temporal operators **F** and **G** can be defined using the basic operators defined above. Similarly to predicates, when $|\mathscr{I}| = 1$, we usually omit the index of freeze operator, as in $*\mathbf{G}_I(x > x^*) = *_1 \mathbf{G}_I(x > x^{*_1})$. Henceforth, let $i, \mu, \varphi, \varphi_1, \varphi_2$ be the same as in Definition 2.5.

Definition 2.6 *Let $s \in (\mathbb{R}^n)^T$ be a signal, $t \in T$ a time point and $t^* \in T^{\mathscr{I}}$ a frozen time vector. Formula satisfaction is defined inductively:*

$$\begin{array}{lcl}
(s,t,t^*) \models \top & & \\
(s,t,t^*) \models \mu & \iff & (s(t), s \circ t^*) \in \mu \\
(s,t,t^*) \models \neg\varphi & \iff & (s,t,t^*) \not\models \varphi \\
(s,t,t^*) \models \varphi_1 \vee \varphi_2 & \iff & (s,t,t^*) \models \varphi_1 \vee (s,t,t^*) \models \varphi_2 \\
(s,t,t^*) \models \varphi_1 \boldsymbol{U}_I \varphi_2 & \iff & \exists\, t' \in t \oplus I : (s,t',t^*) \models \varphi_2 \wedge \\
 & & \forall\, t'' \in [t,t'] : (s,t'',t^*) \models \varphi_1 \\
(s,t,t^*) \models *_i \varphi & \iff & (s,t,t^*[i \leftarrow t]) \models \varphi
\end{array}$$

Operator $\circ$ is used to denote function composition, i.e. $(s \circ t^*) \in (\mathbb{R}^n)^{\mathscr{I}}$ and $(s \circ t^*)(i) = s(t_i^*)$ and $t \oplus I$ stands for $\{t + u \mid u \in I\}$.

Definition 2.7 *Let $s \in (\mathbb{R}^n)^T$ be signal and φ formula. Formula satisfaction by signal is given:*

$$s \models \varphi \iff (s, 0, \mathbf{0}) \models \varphi$$

where $\mathbf{0}$ *denotes the* zero frozen time vector*, i.e.* $\{(i,0) | i \in \mathscr{I}\}$.

Intuitively, interpretation of $*_i \varphi$ is the following: freeze operator stores signal values at the time of $*_i \varphi$ evaluation, which can then be referred to using index i in predicates of φ. An example property, *"in the next five time units, x increases by* 8*"* can be specified as:

$$*\mathbf{F}_{[0,5]}(x \geq x^* + 8)$$

where x^* refers to value of x at time 0.

When intervals associated with until operators are bounded, satisfaction of a given formula can be decided on any finite signal of sufficient length. This length can be determined from the formula structure in a way similar to [24] and corresponds to the furthest time point (among all possible signals) which has to be examined in order to determine formula satisfaction. This clearly also holds for frozen time values.

Definition 2.8 *Let φ be a formula. The* necessary input length *for φ, $l(\varphi)$ is defined inductively:*

$$\begin{aligned} l(\top) = l(\mu) &= 0 \\ l(\neg\varphi) = l(*_i \varphi) &= l(\varphi) \\ l(\varphi_1 \vee \varphi_2) &= \max(l(\varphi_1), l(\varphi_2)) \\ l(\varphi_1 \, \boldsymbol{U}_I \, \varphi_2) &= \max(l(\varphi_1), l(\varphi_2)) + \sup I \end{aligned}$$

When $l(s) < l(\varphi)$ we state that $s \not\models \varphi$.

Frozen time indices and freeze operators share some similarities with variables and quantifiers of predicate logic. We may distinguish free and bound indices, where index i is free if it is used in a predicate (i.e. coefficient a_{ij} is not zero for some j) and is not in the scope of operator $*_i$.

Naturally, whenever i is free in φ, then $s \models \varphi$ iff $s \models *_i \varphi$, since t_i^* is zero in both cases.

Additionally, we may substitute for free indices of a formula in a manner similar to variable substitution. However, it only makes sense to substitute one index for another, which we will denote *index renaming* and express as $\varphi[\pi]$ where π is a total function on $\mathscr{I}$ (but not necessarily a permutation – two indices can be renamed to one) or $\varphi[k/l]$, where k is renamed to l. To preserve formula semantics, renaming is only safe when no free index becomes bound after renaming in any subformula.

3 Robustness Measures for STL*

Following from STL* semantics, robustness of signal s with respect to formula φ is given for each time point t and frozen time vector t^* and denoted by $\rho(\varphi, s, t, t^*)$. We also define $\rho(\varphi, s) = \rho(\varphi, s, 0, \mathbf{0})$. Robustness of signal s with respect to formula φ is a value, which under-approximates the distance of s from the set of signals where φ has different truth value [14]. To express this formally, we first need to define certain basic concepts (where S is a set of signals):

- Distance of signals is given by their maximum pointwise distance: $d(s,s') = \max_{t \in \mathbb{R}_0^+} d(s(t), s'(t))$
- *Set distance* is given by minimum distance to the set: $\mathbf{dist}(s,S) = \min\{d(s,s') \mid s' \in S\}$
- *Set depth* is given by set distance to the complement: $\mathbf{depth}(s,S) = \mathbf{dist}\left(s, \overline{S}\right)$
- *Signed distance* is given: $\mathbf{Dist}(s,S) = \begin{cases} -\mathbf{dist}(s,S) & s \notin S \\ \mathbf{depth}(s,S) & s \in S \end{cases}$

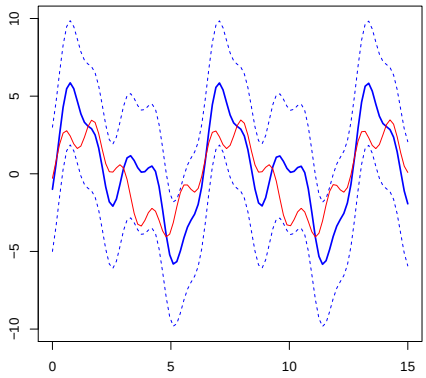


Figure 1: Signal s (blue, thick) and borders of its robust neighbourhood (blue, dashed) with an example of a signal (red) contained in the robust neighbourhood (adapted from [14]).

The value $\rho(\varphi,s)$ underapproximates the signed distance of s from the set of all signals satisfying φ, $\mathscr{L}(\varphi)$, i.e. $|\rho(\varphi,s)| \leq |\mathbf{Dist}(s,\mathscr{L}(\varphi))|$ holds while their signs are identical. The absolute value of $\rho(\varphi,s)$ thus delineates an equidistant tube where all signals satisfy φ if and only if s does – the *robust neighbourhood* of s (see Figure 1).

It would be desirable to define the robustness equal to the signed distance; however, by [14], the robustness computation would not be feasible then. In order to be sound, the robustness definition has to satisfy the following property (for any φ, s, t and t^*):

$$-\mathbf{dist}\,(s,\mathscr{L}_{t,t^*}(\varphi)) \leq \rho(\varphi,s,t,t^*) \leq \mathbf{depth}\,(s,\mathscr{L}_{t,t^*}(\varphi))\,, \tag{1}$$

where $\mathscr{L}_{t,t^*}(\varphi) = \{s \mid (s,t,t^*) \models \varphi\}$. Since $\mathbf{depth}(s,\mathscr{L}_{t,t^*}(\varphi)) = 0$ when $(s,t,t^*) \not\models \varphi$ (and analogously for **dist**), this actually requires that:

1. $s \models \varphi \implies 0 \leq \rho(\varphi,s,t,t^*) \leq \mathbf{depth}\,(s,\mathscr{L}_{t,t^*}(\varphi))$,
2. $s \not\models \varphi \implies -\mathbf{dist}\,(s,\mathscr{L}_{t,t^*}(\varphi)) \leq \rho(\varphi,s,t,t^*) \leq 0$.

Robustness is defined inductively for each logical connective from its semantics in such manner that Boolean functions $\wedge$ and $\vee$ are replaced by real functions min and max (respectively). Quantifiers in the semantics of operator **U** can then be expressed by infinite disjunction or conjunction. Robustness wrt predicate μ is defined as $\mathbf{Dist}(s,\mathscr{L}_{t,t^*}(\mu))$, i.e. the ideal value without underapproximation. If $\rho(\mu,s)$ was lower, it would diminish resulting robustness value, for robustness wrt formula cannot be greater than robustness wrt any of its predicates. Soundness of this definition (property (1)) is, naturally, proved inductively wrt formula structure.

This has already been established by Fainekos et al. in [14], albeit for MTL which does not allow signal value freezing. Nevertheless, their definition can be directly extended for STL*. Intuitively, this is due to frozen time values being only stored by freeze operators and retrieved in predicates, which does not affect other logical connectives. The full proof can be found in [29] (page 83).

Consequently, we have to define robustness for the freeze operator. It follows from its semantics:

$$\mathscr{L}_{t,t^*}(*_i\,\varphi) = \{s \mid (s,t,t^*) \models *_i\,\varphi\} = \{s \mid (s,t,t^*[i \leftarrow t]) \models \varphi\} = \mathscr{L}_{t,t^*[i\leftarrow t]}(\varphi)$$

Thus, robustness of freeze operator can be defined in the following manner:

$$\rho(*_i \varphi, s, t, t^*) = \rho(\varphi, s, t, t^*[i \leftarrow t])$$

Assume $-\mathbf{dist}(s, \mathscr{L}_{t,t^*}(\varphi)) \leq \rho(\varphi, s, t, t^*) \leq \mathbf{depth}(s, \mathscr{L}_{t,t^*}(\varphi))$ for any t, t^*. Therefore, it also holds for t and $t^*[i \leftarrow t]$ and thus:

$$-\mathbf{dist}(s, \mathscr{L}_{t,t^*[i\leftarrow t]}(\varphi)) \leq \rho(\varphi, s, t, t^*[i \leftarrow t]) \leq \mathbf{depth}(s, \mathscr{L}_{t,t^*[i\leftarrow t]}(\varphi))$$

From which follows the validity of (1) for $\rho(*_i \varphi, s, t, t^*)$. STL* robustness for logical connectives is presented in Figure 2.

$$\begin{aligned}
&\rho(\top, s, t, t^*) && = +\infty \\
&\rho(\neg\varphi, s, t, t^*) && = -\rho(\varphi, s, t, t^*) \\
&\rho(\varphi_1 \vee \varphi_2, s, t, t^*) && = \max\left(\rho(\varphi_1, s, t, t^*), \rho(\varphi_2, s, t, t^*)\right) \\
&\rho(\varphi_1 \,\mathbf{U}_I\, \varphi_2, s, t, t^*) && = \max_{t' \in t \oplus I} \min\left(\rho(\varphi_2, s, t', t^*), \min_{t'' \in [t,t']} \rho(\varphi_1, s, t'', t^*)\right) \\
&\rho(*_i \varphi, s, t, t^*) && = \rho(\varphi, s, t, t^*[i \leftarrow t])
\end{aligned}$$

Figure 2: Robustness of STL* logical connectives.

3.1 Robustness of Predicates

Finding $\mathbf{Dist}(s, \mathscr{L}_{t,t^*}(\mu))$ generally constitutes a convex analysis problem [14]. Thus, it could be solved using convex programming for each t and t^*, which would, however, greatly increase computation time, and therefore, analytic solution is preferable. To this end, we have restricted STL* predicates to be linear.

For predicate μ with coefficients a_{ij}, b, the problem of finding $\mathbf{Dist}(s, \mathscr{L}_{t,t^*}(\mu))$ can be reduced to optimization of $f(\mathbf{d}) = \max_i \sum_j d_{ij}^2$ (where $i \in \mathscr{I}$ and $j \in \{1, \ldots, n\}$) under the constraint $\sum_i \sum_j a_{ij} d_{ij} + \varepsilon = 0$ for some positive ε. This is a non-trivial problem, since f is not differentiable at point $\mathbf{d}$ where $f(\mathbf{d}) = \sum_j d_{kj}^2 = \sum_j d_{lj}^2$ for some $k \neq l$. To solve it, generalized method of Lagrange multipliers from [6] was used, resulting in the following definition of the robustness ρ (detailed derivation can be found in [29] (page 47)).

Definition 3.1 *Let μ be a predicate with coefficients a_{ij}, b. Then*

$$\rho(\mu, s, t, t^*) = \frac{\sum_j a_{0j} s_j(t) + \sum_i \sum_j a_{ij} s_j(t_i^*) + b}{\sum_i \sqrt{\sum_j a_{ij}^2}}$$

for arbitrary s, t, t^*, i ranging over $\mathscr{I}$, j ranging over $\{1, \ldots, n\}$.

The numerator corresponds to the left-hand side value of the predicate.

It holds that $\rho(\mu, s, t, t^*) = \mathbf{Dist}(s, \mathscr{L}_{t,t^*}(\mu))$, unless some time points given by t and t^* are equal. This originates from the optimization problem, where $t_k^* = t_l^*$ (or $t = t_k^*$) would constitute another constraint, which might change the solution.

Suppose that $t_k^* = t_l^*$ (reasoning for $t = t_k^*$ is similar). We can merge (sum) coefficients a_{kj} and a_{lj} for any given j, which effectively reduces the number of considered frozen times. Robustness of

predicates with merged coefficients is greater, since the denominator of definition 3.1 becomes smaller as $\sqrt{\sum_j \left(a_{kj}+a_{lj}\right)^2} \leq \sqrt{\sum_j a_{kj}^2}+\sqrt{\sum_j a_{lj}^2}$ due to triangle inequality. Therefore, even if we disregard possible time point equality, property (1) still holds. However, the greater the value of $\rho(\mu,s,t,t^*)$ is, the better approximation of $\mathbf{Dist}(s,\mathscr{L}_{t,t^*}(\varphi))$ is obtained. Therefore, we will investigate two distinct cases when time points can be equal:

1. It happens consistently for given formula φ and predicate μ, i.e. φ is built in such way that the same time value is stored by freeze operator associated with both indices, such as:

$$\psi = \mathbf{G}_{I_1}(*_i \neg *_j \mathbf{F}_{I_2}(x^{*_i}+x^{*_j} \geq x))$$

2. It is a result of $\varphi \equiv *_i(\varphi_1 \mathbf{U}_I \varphi_2)$ (or similar formula) evaluation:

$$(s,t,t^*) \models \varphi \iff (s,t,t^*[i \leftarrow t]) \models \varphi_1 \mathbf{U}_{[a,b]} \varphi_2 \iff$$
$$\exists t' \in [a+t,b+t] : (s,t',t^*[i \leftarrow t]) \models \varphi_2 \wedge \forall t'' \in [t,t'] : (s,t'',t^*[i \leftarrow t]) \models \varphi_1$$

   When $a=0$, it may occur that $t'=t$. Additionally, $t'' \in [t,t']$, therefore, satisfaction of φ_1 by $(s,t,t^*[i \leftarrow t])$ has to be evaluated. The equality of t and i-th frozen time may be propagated to predicates. We have decided to omit this case in order to simplify robustness computation.

3.2 Improving Approximation

The formula ψ (see above) is obviously badly written, since it can be reformulated with only one frozen time index: $\mathbf{G}_{I_1}(\neg * \mathbf{F}_{I_2}(2x^* \geq x))$. This eliminates time point equality and thus improves robustness approximation. We have formulated three rules which can be used to automatically rewrite formula so that it does not induce consistent time point equality (while preserving its meaning):

1. Freeze operator is distributive over Boolean connectives. Consequently, freeze operators can be moved down along the formula syntax tree until they reach a temporal operator, predicate or another freeze operator.
2. Freeze operator preceding predicate can be merged with the predicate (associated coefficients being merged with coefficients for unfrozen time).
3. Two consecutive freeze operators and their associated indices can be merged. However, in order to preserve the formula meaning, a completely new index has to be chosen as the result of merging.

Subsequently, all STL* formulae can be written in such manner that each freeze operator is followed by until operator, which also ensures that all frozen time indices generally refer to distinct time points. Indeed, all meaningful formulae (i.e. not serving to illustrate semantic peculiarities) in [7] are specified in this manner.

This reinforces the connection between temporal operators and freeze operators expressiveness. Subsequently, it may be practical to define an alternate STL* syntax, where signal value freezing is directly tied to the until operator, such as $\varphi_1 \mathbf{U}_I^{*_i} \varphi_2 \equiv *_i(\varphi_1 \mathbf{U}_I \varphi_2)$. However, we do not deem it necessary, seeing that it entails no expressiveness gain. Moreover, the current syntax of STL* may permit shorter and more transparent formulae.

It should be noted that although application of previous rules may increase number of indices used in a formula (due to the rule (3) which introduces one new index), it does not increase the number of free indices in each subformula. On the contrary, the number of free indices may decrease.

4 Computation

To compute (or monitor) robustness of continuous signal, we use the approach of Fainekos et al. [14], which is based on discrete robustness semantics. The following procedure is used:

1. Sample input signal $s : T \to \mathbb{R}^m$ into a *timed state sequence* $(\tau, \sigma) : \mathbb{N} \to T \times \mathbb{R}^m$.
2. Compute robustness over points of the resulting timed state sequence (i.e. the discrete robustness).

This only approximates continuous robustness of s. When MTL robustness is concerned, Fainekos et al. give bound for error introduced by this approximation under certain conditions, which can be summarized as signal sampling being sufficiently dense with respect to given formula. We assume this strong theorem translates to STL* (as STL* robustness extends MTL robustness) and deem the previous procedure good approximation for an input signal with large enough sampling rate.

Before the robustness monitoring algorithm is described, we should note that it can also be used to decide formula satisfaction, since positive robustness implies formula satisfaction (and negative its invalidity). However, when $\rho(\varphi, s) = 0$ no information about formula satisfaction can be derived. Additionally, robustness measure only underapproximates the robust neighbourhood, and so the robustness value may be zero even if clearly s satisfies φ. Consequently, classical monitoring may produce more precise results.

Algorithm 1 computes robustness for a STL* formula and sufficiently long timed state sequence (which may constitute a sampled signal). It copies inductive definition of robustness with recursive calls of procedure MONITOR (line 4), which computes robustness only in the points of given state sequence. Therefore, instead of frozen time vector $t^* : \left(\mathbb{R}_0^+\right)^{\mathscr{J}}$, *frozen state vector* $\iota^* : \mathbb{N}^{\mathscr{J}}$ is used. The computation starts at zero index and zero frozen state vector (line 3), which ensures only robustness values needed for resulting robustness evaluation are computed.

Robustness values with respect to subformulae of input formula are not stored. Instead, they are computed every time procedure MONITOR is called on a given subformula. The reasoning behind this practise is the following: For the majority of formulae, the value of robustness for given ι and ι^* is obtained by a simple – constant-time – operation on just a single value of robustness (or two in the case of $\vee$). Additionally, the robustness with respect to predicates can be computed in constant time.

The only operator where robustness depends on robustness values over an interval is the until operator (and by extension all derived temporal operators). Consequently, robustness values associated with until operators are stored. Furthermore, when MONITOR($\varphi_1 \mathbf{U}_I \varphi_2, \iota, \iota^*$) is called for the first time, robustness values with respect to $\varphi_1 \mathbf{U}_I \varphi_2$ for ι^* and all ι' are precomputed (see lines 10–17) by the procedure PRECOMPUTEUNTIL, which constitutes an algorithmic version of robustness definition for until operator. These precomputed values are expected to be referred to later, since robustness computation is restricted to time interval $[0, l(\varphi)]$ which comprises all input values necessary to evaluate $\rho(\varphi, (\tau, \sigma))$.

4.1 Complexity

Apparently, the most time-consuming task of Algorithm 1 is the PRECOMPUTEUNTIL procedure, which is quadratic to the number of states in the input timed state sequence. In the worst case it is called for each ι^*. Therefore, the complexity of Algorithm 1 is in $\mathscr{O}\left(|\varphi| \cdot n^{2|\mathscr{J}|}\right)$ where n is the size of input timed state sequence. For sampled signals, it may be expressed using necessary length, resulting in alternate complexity formulation: $\mathscr{O}\left(|\varphi| \cdot l(\varphi)^{2|\mathscr{J}|} \cdot f^{2|\mathscr{J}|}\right)$ where f is the sampling rate of input signal, which correlates with the precision of robustness computation. Space complexity can be bounded by the same function.

Algorithm 1 Robustness Monitoring for STL*

Input: STL* formula φ and timed state sequence (τ,σ) of length greater than $l(\varphi)$ (see Definition 2.8).
Output: The value of $\rho(\varphi,(\tau,\sigma))$.

For any i free in φ, $\varphi \leftarrow *_i \varphi$.

$P \leftarrow \emptyset$ ▷ Precomupted robustness values.
return MONITOR$(\varphi, 0, \mathbf{0})$

procedure MONITOR$(\varphi, \imath, \imath^*)$
  if $\varphi \equiv \top$ **then return** $+\infty$
  else if $\varphi \equiv \mu$ **then return** $\rho(\mu,(\tau,\sigma),\imath,\imath^*)$ ▷ According to Definition 3.1.
  else if $\varphi \equiv \neg\varphi_1$ **then return** $-$MONITOR$(\varphi_1,\imath,\imath^*)$
  else if $\varphi \equiv \varphi_1 \vee \varphi_2$ **then return** $\max($MONITOR$(\varphi_1,\imath,\imath^*),$MONITOR$(\varphi_2,\imath,\imath^*))$
  else if $\varphi \equiv *_i \varphi_1$ **then return** MONITOR$(\varphi_1,\imath,\imath^*[i \leftarrow \imath])$
  else if $\varphi \equiv \varphi_1 \mathbf{U}_{[a,b]} \varphi_2$ **then**
    if $(\varphi,\imath^*) \in \mathbf{dom}(P)$ **then**
      return $P(\varphi,\imath^*)(\imath)$
    else
      $\rho \leftarrow$ PRECOMPUTEUNTIL$(\varphi_1,\varphi_2,a,b,\imath^*)$
      $P \leftarrow P \cup ((\varphi,\imath^*),\rho)$
      return $\rho_\imath$
    end if
  end if
end procedure

procedure PRECOMPUTEUNTIL$(\varphi_1,\varphi_2,a,b,\imath^*)$
  $i \leftarrow 0$
  $l \leftarrow \max(l(\varphi_1),l(\varphi_2))$
  $\rho \leftarrow \emptyset$ ▷ Sequence of robutness values.
  while $\tau_i + b + l \le l(\tau)$ **do**
    $j \leftarrow 0$
    $r_1 \leftarrow$ MONITOR$(\varphi_1,i,\imath^*)$
    while $\tau_{i+j} < \tau_i + a$ **do** ▷ Before $[\tau_i+a,\tau_i+b]$.
      $r_1 \leftarrow \min(r_1,$MONITOR$(\varphi_1,i+j,\imath^*))$
      $j \leftarrow j+1$
    end while
    $r \leftarrow r_1$
    while $\tau_{i+j} \le \tau_i + b$ **do** ▷ Inside $[\tau_i+a,\tau_i+b]$.
      $r_1 \leftarrow \min(r_1,$MONITOR$(\varphi_1,i+j,\imath^*))$
      $r_2 \leftarrow$ MONITOR$(\varphi_2,i+j,\imath^*)$
      $r \leftarrow \max(r,\min(r_1,r_2))$
      $j \leftarrow j+1$
    end while
    $\rho \leftarrow \rho \cup \{(i,r)\}$ ▷ Set the value of ρ_i.
    $i \leftarrow i+1$
  end while
end procedure

The parameter most adversely affecting the algorithm complexity is the size of frozen time index set $|\mathscr{I}|$. Naturally, $\mathscr{I}$ can be restricted to indices used in input formula. In most practical cases, their number will be small. This is supported by the following result:

Theorem 4.1 *Any formula φ can be rewritten into a semantically equivalent formula which uses only so many indices as is the maximum number of free indices in subformulae of φ.*

Note that the number of free indices may increase as we descend into subformulae.

This statement derives from the fact that an index only serves to associate one freeze operator with a set of coefficients in one or more predicates and it is free on all paths between this freeze operator and all associated predicates. Therefore, indices which are never simultaneously free need not be different.

The result of this theorem can be realized by an automatic procedure which renames frozen time indices in a formula while traversing its syntax tree (using DFS). This procedure stores pairs of indices $[k/l]$ corresponding to the renaming of source index k in the original formula φ to destination index l in its optimized version φ'. When the procedure encounters freeze operator $*_i$, new pair $[i/m]$ is introduced where m is the smallest unused destination index and the operator is changed to $*_m$. Whenever k becomes free in φ, the pair $[k/l]$ is removed and l can be reused. Upon reaching a predicate, all stored pairs are applied as a renaming.

This procedure is described in greater detail in [29] (page 44) where additional justification of its correctness can also be found.

Together with freeze operator merging described in Section 3.2 (which does not increase number of free indices), this can considerably decrease the number of indices used in a formula and thus the time complexity of robustness monitoring. Although intelligent formula specification may result in already optimal formula, the existence of automatic optimization procedures reduces demands on writers of formulae.

4.2 Implementation

The algorithm has been implemented as an extension of the tool Parasim [12]. Parasim is a highly modular Java-based open-source tool with graphical user interface for computing robustness of a model with respect to perturbations. Integrating the algorithm presented in this paper into an already existing tool has an additional advantage of facilitating the use of STL* robustness in practise.

Given a model, STL* formula and perturbation set, Parasim samples the perturbation set into points and for each point simulates the model and computes robustness of the resulting signal with respect to STL* robustness measure. In the neighbourhood of signals with low robustness, additional points are sampled. Formula optimizing algorithms are implemented to maximize efficiency.

5 Case Study

By employing the Parasim tool we have conducted several experiments on two simple population dynamics models. The experiments have also served us to briefly evaluate the algorithm performance (in the setting of the Parasim tool).

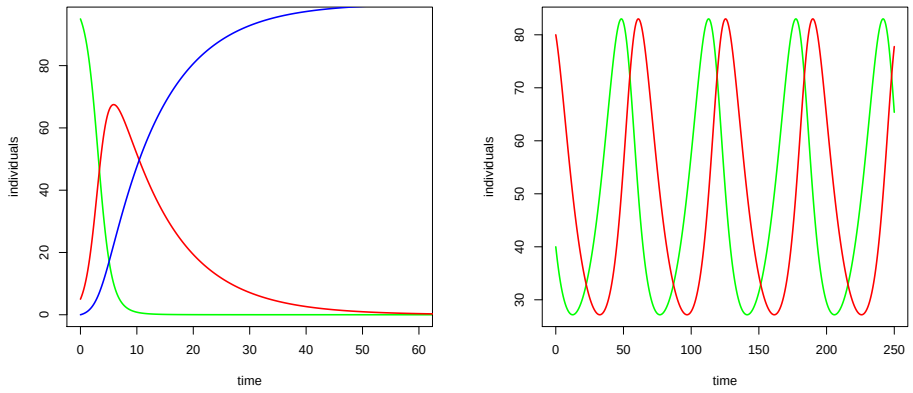
5.1 SIR Model

First, we demonstrate the robustness analysis on the model simulating an outbreak of an infectious disease in a population [19]. The simulated population is divided into three categories: *susceptible* (*S*),

infected (I) and *recovered* (R). A susceptible individual can become infected by contact with another infected individual and an infected individual may recover. The ODE model is the following:

$$\frac{dS}{dt} = -\alpha SI \qquad \frac{dI}{dt} = \alpha SI - \beta I \qquad \frac{dR}{dt} = \beta I$$

Where α is the *contact rate* which correlates to probability of disease transmission, while β, the *recovery rate*, takes into account the standard length of recovery. A typical simulation of this model (see Figure 3a) includes a rapid increase in infected individuals, which is then followed by their gradual recovery.



(a) (b)

Figure 3: (a) Typical development of SIR model, showing the number of susceptible (green), infected (red) and recovered (blue) individuals. (b) Typical development of populations in predator-prey model, showing number of prey (green) and predator (red).

In this case study, we compare robustness analysis based on a formula containing value-freezing with respect to a freezing-free formula analysis exploiting a similar behavioural pattern. In particular, we consider the following formulae:

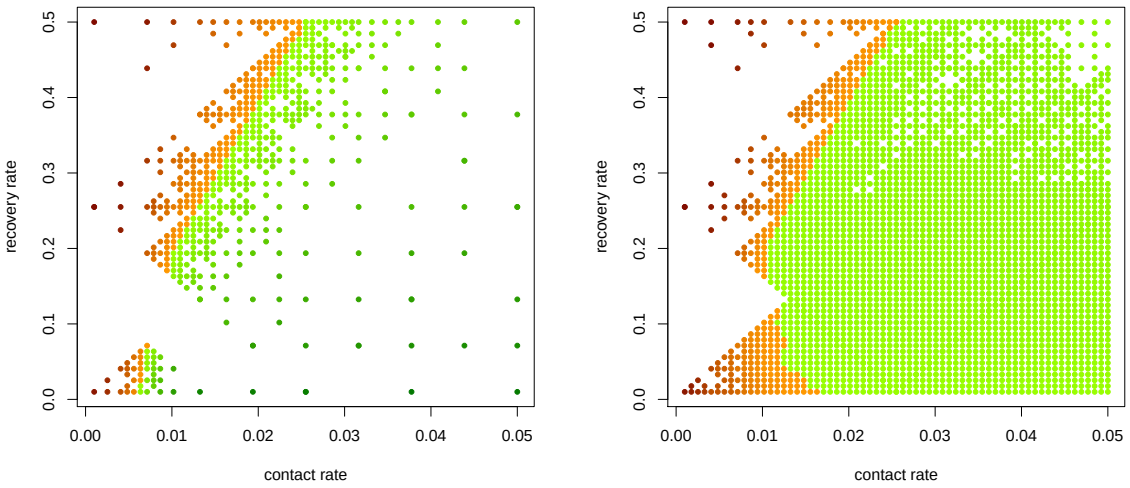
$$\text{STL} : \varphi_1 = \mathbf{F}_{[1,5]}(I \geq 50) \qquad \text{STL}^* : \varphi_2 = \mathbf{F}_{[1,5]}\left(I \geq 50 \wedge * \mathbf{G}_{[0.25,5]}(I^* \geq I)\right)$$

Both formulae require the number of infected individuals to be greater than 50 at some time in the interval $[1,5]$, while φ_2 also requires this number to be the local maximum (the number of infected individuals is required to decrease after reaching this maximum).

The robustness with respect to both properties was analysed on perturbations of both contact rate and recovery rate. Results are presented in Figure 4.

While the satisfaction sets of φ_1 and φ_2 (delineated by positive robustness) are essentially identical, the actual robustness values show a significant difference. Generally, when they are positive, the value of robustness with respect to φ_1 at given point is considerably greater than the corresponding value of robustness with respect to φ_2. In Figure 4, this can be seen as lighter shade of green points in 4b. Also, lower robustness causes the apparent increase in the number of points.

The reason for the rapid change in robustness comes from evaluation of the subformula $*\mathbf{G}_{[0.25,5]}(I^* \geq I)$ that describes the local extreme. When evaluated in time t, robustness is proportional to the difference $(I[t] - I[t+0.25])$ (by Definition 3.1). In practise, the difference is small provided that the descent of I is not extremely steep. This causes such formulae to have typically low robustness values on common signals.



(a) Robustness wrt φ_1.

(b) Robustness wrt φ_2.

Figure 4: Robustness of SIR model with respect to φ_1 and φ_2 for variable contact and recovery rates. Robustness was positive in green points and negative in orange points. Darker colour represents greater absolute value of robustness.

5.2 Predator-Prey Model

In the second case study we analyse the predator-prey model [23, 30], which attains oscillating behaviour for a wide variety of parameters. We use a variant of the Lotka-Volterra model represented by the following ordinary differential equations:

$$\frac{dX}{dt} = \nu X - \alpha XY \qquad \frac{dY}{dt} = \alpha XY - \mu Y$$

The model simulates a situation where a prey species X is hunted by a predator species Y with the simplifying assumption that predator birth rate and prey death rate are equal and proportional to the probability of prey and predator contact, and thus to the product of both species populations. We use the following coefficients: prey natality (ν), predator mortality (μ) and predation rate (α). Typical behaviour of this models constitutes periodic oscillations (see Figure 3b).

We consider perturbation of two aforementioned coefficients, ν and α, and compute robustness with respect to two properties specified by the following formulae:

$$\psi_1 = \mathbf{G}_{[0,300]} * \mathbf{F}_{[0,100]} (X \geq Y^*)$$
$$\psi_2 = \mathbf{G}_{[0,300]} \left(X \geq 1 \wedge Y \geq 1 \wedge \mathbf{F}_{[0,50]} * \left(\mathbf{F}_{[0,75]} (X^* - X \geq 25) \wedge \mathbf{F}_{[0,75]} (X - X^* \geq 25)\right)\right)$$

The property ψ_1 requires that for each time point $t \in [0,300]$, there is a subsequent time point $t' \in [t, t+100]$ such that population of prey in t' is greater than population of predators in t. According to Definition 3.1 its corresponding robustness can be expressed as follows:

$$\rho(\varphi, s) = \min_{t \in [0,300]} \max_{t' \in [t, t+100]} \frac{X[t'] - Y[t]}{2}$$

where $X[t']$ and $Y[t]$ denote values of s associated with given species at given time. The robustness value is maximized with respect to t' and minimized with respect to t, therefore, it uses maximal values of both X and Y. Consequently, this property can be interpreted as maximum population of prey being greater then maximum population of predators (restricted to given intervals).

Formula ψ_2 is based on the similar principle. While rejecting aberrant behaviour where population of one of the species drops below one individual, intuitively, it requires that there always is time in the future when population of prey can increase or decrease by 25 individuals, which is stated by the subformula $\mathbf{F}_{[0,50]} * \left(\mathbf{F}_{[0,75]}\left(X^* - X \geq 25\right) \wedge \mathbf{F}_{[0,75]}\left(X - X^* \geq 25\right)\right)$. Therefore, ψ is satisfied when the difference between maximal and minimal prey population is greater than 50 and the associated robustness is proportional to this difference. Again, we have avoided use of the extreme property, which would adversely affect robustness value.

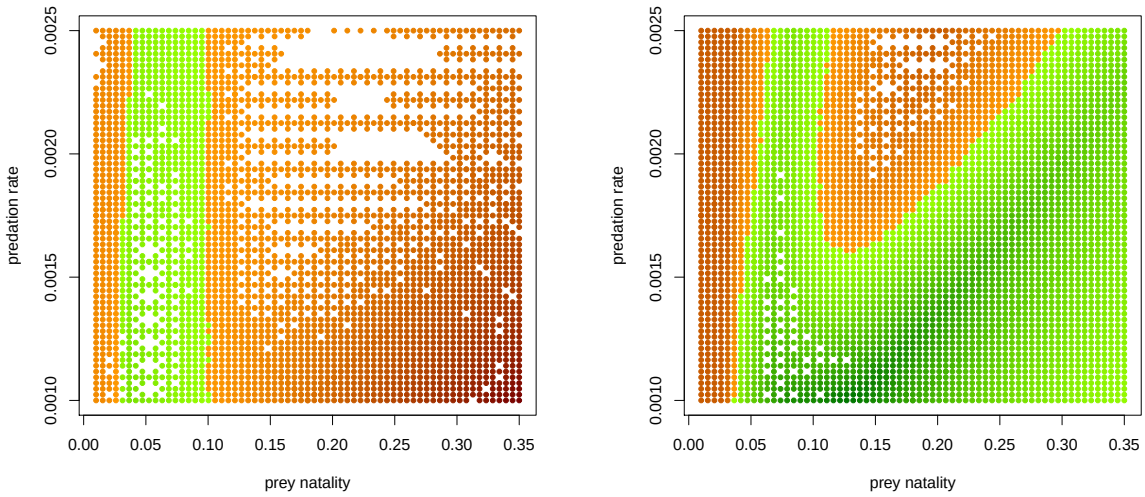


Figure 5: Robustness of predator-prey model with respect to ψ_1 (left) and ψ_2 (right) for variable prey natality and predation rate. Robustness was positive in green points and negative in orange points. Darker colour represents greater absolute value of robustness.

Results of this analysis are presented in Figure 5. Here, we should point out that small prey natality produced behaviour where predator population approached zero and period of oscillations was greatly increased. For such behaviour, intervals used in ψ_1 and ψ_2 were shorter than one period.

Apparently, satisfaction of ψ_1 is not affected by predation rate. More interestingly, when prey natality increases, predator population exceeds that of prey (see Figure 5 (left)). Figure 5 (right) shows that amplitude of prey population oscillation is affected by both prey natality and predation rate.

The above results have been confirmed by simulation.

5.3 Performance

Performance of robustness analysis is summarized in Table 1. All results have been obtained by executing the algorithm implementation on a 4 core 2 GHz CPU with 4 GB RAM. Each computation has been arranged into 8 threads. For each analysis we have set an optimal resolution of the trajectories (number of simulated points). The number of simulated trajectories has been bounded by the number of refinement iterations in the Parasim parameter space sampling procedure.

It is worth noting that all analysed properties consist only of **F** and **G** operators for which the procedure is optimized by employing Lemire queues in the same way as proposed in [10]. This is based on an optimal streaming algorithm for computing maxima (resp. minima) of a numerical sequence and allows to reduce the quadratic complexity wrt formula size to linear.

Property (model)	formula size	# trajectories	# points per a trajectory	time
φ_1 (SIR)	2	250	500	8.6 s
φ_2 (SIR)	6	1365	1000	15.2 s
ψ_1 (Predator-Prey)	4	831	400	85.4 s
ψ_2 (Predator-Prey)	12	1293	423	309.4 s

Table 1: Performance of the robustness computation measured on the prototype implementation.

The increase in computation time in the case of ψ_1 is caused by longer time intervals quantifying the temporal operators. Computation of the property ψ_2 has been slowed down due to insufficient memory.

6 Conclusion

In this paper we have set up a robustness measure for a value-freezing extension of STL. The robustness of a signal with respect to a given STL* property is based on the distance of the signal from signals violating the property. We have introduced a measure that is proved to fulfil requirements imposed on robustness measures as defined in [14]. This guarantees that the robustness measure is defined correctly. We have derived the algorithm for STL* robustness computation from the discrete robustness and implemented it as an extension of the tool Parasim [12].

Some of the properties from case studies required comparison of signal values at near frozen time points. Robustness of such properties is typically small. This is only natural as such properties represent stricter requirements on signals. However, this feature may also constitute a detriment for tools such as Parasim, which use robustness to direct perturbation set sampling. This is the exact case of analysed SIR model and property φ_2. It must be noted, though, that this problem is encompassed by the much broader issue of meaningful property design.

In [14] the authors quantify error in robustness value caused by the approximate computation. We have not yet explored this possibility for STL* robustness measures and leave this for future work. However, results in [14] imply this error is inversely proportional to the rate of input signal sampling.

References

[1] Rajeev Alur, Tomás Feder & Thomas A. Henzinger (1996): *The Benefits of Relaxing Punctuality*. *Journal of the ACM* 43(1), pp. 116–146, doi:10.1145/227595.227602.

[2] Yashwanth Singh Rahul Annapureddy, Che Liu, Georgios E. Fainekos & Sriram Sankaranarayanan (2011): *S-TaLiRo: A Tool for Temporal Logic Falsification for Hybrid Systems*. In: *Tools and Algorithms for the Construction and Analysis of Systems*, *Lecture Notes in Computer Science* 6605, Springer, pp. 254–257, doi:10.1007/978-3-642-19835-9_21.

[3] Luboš Brim, Milan Češka & David Šafránek (2013): *Model Checking of Biological Systems*. In Marco Bernardo, Erik Vink, Alessandra Pierro & Herbert Wiklicky, editors: *Formal Methods for Dynamical Systems*, *Lecture Notes in Computer Science* 7938, Springer Berlin Heidelberg, pp. 63–112, doi:10.1007/978-3-642-38874-3_3.

[4] Laurence Calzone, François Fages & Sylvain Soliman (2006): *BIOCHAM: An Environment for Modeling Biological Systems and Formalizing Experimental Knowledge*. Bioinformatics 22(14), pp. 1805–1807, doi:10.1093/bioinformatics/btl172.

[5] Edmund M. Clarke, Orna Grumberg & Doron A. Peled (2000): *Model Checking*. MIT Press.

[6] Frank H. Clarke (1983): *Optimization and Nonsmooth Analysis*. S.I.A.M.

[7] Petr Dluhoš, Luboš Brim & David Šafránek (2012): *On Expressing and Monitoring Oscillatory Dynamics*. In: Proceedings First International Workshop on Hybrid Systems and Biology, Open Publ. Assoc., pp. 73–87, doi:10.4204/EPTCS.92.6.

[8] Alexandre Donzé (2010): *Breach, A Toolbox for Verification and Parameter Synthesis of Hybrid Systems*. In Tayssir Touili, Byron Cook & Paul Jackson, editors: Computer Aided Verification, Lecture Notes in Computer Science 6174, Springer Berlin Heidelberg, pp. 167–170, doi:10.1007/978-3-642-14295-6_17.

[9] Alexandre Donzé, Gilles Clermont, Axel Legay & Christopher J. Langmead (2009): *Parameter Synthesis in Nonlinear Dynamical Systems: Application to Systems Biology*. In: Research in Computational Molecular Biology, Lecture Notes in Computer Science 5541, Springer, pp. 155–169, doi:10.1007/978-3-642-02008-7_11.

[10] Alexandre Donzé, Thomas Ferrre & Oded Maler (2013): *Efficient Robust Monitoring for STL*. In Natasha Sharygina & Helmut Veith, editors: Computer Aided Verification, Lecture Notes in Computer Science 8044, Springer Berlin Heidelberg, pp. 264–279, doi:10.1007/978-3-642-39799-8_19.

[11] Alexandre Donzé & Oded Maler (2010): *Robust Satisfaction of Temporal Logic over Real-Valued Signals*. In: FORMATS 2010, Springer, pp. 92–106, doi:10.1007/978-3-642-15297-9_9.

[12] Faculty of Informatics, Masaryk University (2013): *Parasim: Tool for Parallel Simulations and Verification*. Available at `https://github.com/sybila/parasim/wiki`.

[13] Georgios Fainekos & George Pappas (2006): *Robustness of Temporal Logic Specifications*. In Klaus Havelund, Manuel Nunez, Grigore Rosu & Burkhart Wolff, editors: Formal Approaches to Software Testing and Runtime Verification, Lecture Notes in Computer Science 4262, Springer Berlin Heidelberg, pp. 178–192, doi:10.1007/11940197_12.

[14] Georgios E. Fainekos & George J. Pappas (2009): *Robustness of Temporal Logic Specifications For Continuous-Time Signals*. Theoretical Computer Science 410(42), pp. 4262–4291, doi:10.1016/j.tcs.2009.06.021.

[15] Leon Glass & JoelS. Pasternack (1978): *Prediction of limit cycles in mathematical models of biological oscillations*. Bulletin of Mathematical Biology 40(1), pp. 27–44, doi:10.1007/BF02463128.

[16] Radu Grosu, Gregory Batt, FlavioH. Fenton, James Glimm, Colas Guernic, ScottA. Smolka & Ezio Bartocci (2011): *From Cardiac Cells to Genetic Regulatory Networks*. In Ganesh Gopalakrishnan & Shaz Qadeer, editors: Computer Aided Verification, Lecture Notes in Computer Science 6806, Springer Berlin Heidelberg, pp. 396–411, doi:10.1007/978-3-642-22110-1_31.

[17] Benno Hess (2000): *Periodic Patterns in Biology*. Naturwissenschaften 87(5), pp. 199–211, doi:10.1007/s001140050704.

[18] Shai Kaplan, Anat Bren, Erez Dekel & Uri Alon (2008): *The incoherent feed-forward loop can generate non-monotonic input functions for genes*. Molecular Systems Biology 4(1), pp. –, doi:10.1038/msb.2008.43.

[19] William O. Kermack & Anderson G. McKendrick (1927): *A Contribution to the Mathematical Theory of Epidemics*. Proceedings of the Royal Society of London. Series A 115(772), pp. 700–721, doi:10.1098/rspa.1927.0118.

[20] Hiroaki Kitano (2004): *Biological Robustness*. Nature Reviews Genetics 5(11), pp. 826–837, doi:10.1038/nrg1471.

[21] Ron Koymans (1990): *Specifying Real-Time Properties with Metric Temporal Logic*. Real-Time Systems 2, pp. 255–299, doi:10.1007/BF01995674.

[22] Pavel Krejci, Vitezslav Bryja, Jiri Pachernik, Ales Hampl, Robert Pogue, Pertchoui Mekikian & William R Wilcox (2004): *FGF2 inhibits proliferation and alters the cartilage-like phenotype of RCS cells*. Experimental Cell Research 297(1), pp. 152 – 164, doi:10.1016/j.yexcr.2004.03.011.

[23] Alfred J. Lotka (1925): *Elements of Physical Biology*. Williams and Wilkins, Baltimore.

[24] Oded Maler & Dejan Nickovic (2004): *Monitoring Temporal Properties of Continuous Signals*. In Yassine Lakhnech & Sergio Yovine, editors: Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems, Lecture Notes in Computer Science 3253, Springer Berlin Heidelberg, pp. 152–166, doi:10.1007/978-3-540-30206-3_12.

[25] Elisabetta De Maria, François Fages, Aurélien Rizk & Sylvain Soliman (2011): *Design, Optimization and Predictions of a Coupled Model of the Cell Cycle, Circadian Clock, DNA Repair System, Irinotecan Metabolism and Exposure Control under Temporal Logic Constraints*. Theoretical Computer Science 412(21), pp. 2108–2127, doi:10.1016/j.tcs.2010.10.036.

[26] Aurélien Rizk, Grégory Batt, François Fages & Sylvain Soliman (2011): *Continuous valuations of temporal logic specifications with applications to parameter optimization and robustness measures*. Theor. Comput. Sci. 412(26), pp. 2827–2839, doi:10.1016/j.tcs.2010.05.008.

[27] Aurlien Rizk, Gregory Batt, Franois Fages & Sylvain Soliman (2009): *A general computational method for robustness analysis with applications to synthetic gene networks*. Bioinformatics 25(12), pp. i169–i178, doi:10.1093/bioinformatics/btp200.

[28] Szymon Stoma et al. (2013): *STL-based Analysis of TRAIL-induced Apoptosis Challenges the Notion of Type I/Type II Cell Line Classification*. PLoS Comput Biol 9(5), p. e1003056, doi:10.1371/journal.pcbi.1003056.

[29] Tomáš Vejpustek (2013): *Robustness Analysis of Extended Signal Temporal Logic STL**. Master's thesis, Masaryk University, Faculty of Informatics. Available at `http://is.muni.cz/th/324713/fi_m/`.

[30] Vito Volterra (1928): *Variations and Fluctuations of the Number of Individuals in Animal Species living together*. Journal du Conseil 3(1), pp. 3–51, doi:10.1093/icesjms/3.1.3.