Trace Simplifications preserving Temporal Logic Formulae with Case Study in a Coupled Model of the Cell Cycle and the Circadian Clock

Pauline Traynard and François Fages and Sylvain Soliman

Inria Paris-Rocquencourt, Team Lifeware, France

Abstract. Calibrating dynamical models on experimental data time series is a central task in computational systems biology. When numerical values for model parameters can be found to fit the data, the model can be used to make predictions, whereas the absence of any good fit may suggest to revisit the structure of the model and gain new insights in the biology of the system. Temporal logic provides a formal framework to deal with imprecise data and specify a wide variety of dynamical behaviors. It can be used to extract information from numerical traces coming from either experimental data or model simulations, and to specify the expected behaviors for model calibration. The computation time of the different methods depends on the number of points in the trace so the question of trace simplification is important to improve their performance. In this paper we study this problem and provide a series of trace simplifications which are correct to perform for some common temporal logic formulae. We give some general soundness theorems, and apply this approach to period and phase constraints on the circadian clock and the cell cycle. In this application, temporal logic patterns are used to compute the relevant characteristics of the experimental traces, and to measure the adequacy of the model to its specification on simulation traces. Speed-ups by several orders of magnitude are obtained by trace simplification even when produced by smart numerical integration methods.

1 Introduction

Calibrating dynamical models on experimental data time series is a central task in computational systems biology. When numerical values for model parameters can be found to fit the data, the model can be used to make predictions, whereas the absence of any good fit may suggest to revisit the structure of the model and gain new insights in the biology of the system, see for instance [23,15].

Temporal logic provides a formal framework to deal with imprecise data and specify a wide variety of dynamical behaviors. In the early days of systems biology, propositional temporal logic was proposed by computer scientists to formalize the Boolean properties of the behavior of biochemical reaction systems [11,5] or gene regulatory networks [4,3]. Generalizing these techniques to

quantitative models can be done in two ways: either by discretizing the different regimes of the dynamics in piece-wise linear or affine models [8,2], or by relying on numerical simulations and taking a first-order version of temporal logic with constraints on concentrations, as query language for the numerical traces [1,13,14]. Such language can be used not only to extract information from numerical traces coming from either experimental data or model simulations, but also to specify the expected behaviors as constraints for model calibration and robustness measure [20,21,9].

The general idea of model-checking a single finite trace has been well known for years, notably in the framework of Runtime Verification [17]. It usually relies on the classical bottom-up algorithm, which is bilinear [22]. This extends even to quantitative model-checking like the continuous interpretation of Signal Temporal Logic [10] since the combination of two booleans or two reals by min/-max is cheap. However, when using the full power of First-Order Linear Time Logic (FO-LTL) to compute validity domains, the dependency of the complexity on the size of the trace is no longer linear but exponential in the number of variables [13], reflecting the computational cost of combining complex domains. The question of trace simplification [14] is therefore important to improve the performance of FO-LTL constraint solving, and with it of the corresponding calibration methods.

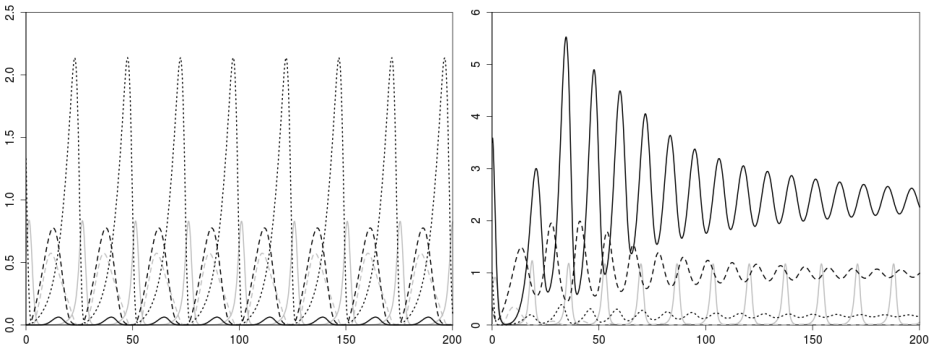

Fig. 1. Traces of some elements of the coupled cell cycle (*MPF* and *Wee1* in grey, respectively solid and dashed lines) and circadian clock (*PerCry*, *Bmal1* and $RevErb_\alpha$ in black, respectively solid, dashed and dotted lines) models with different parameter sets.

In this paper we provide a series of trace simplifications which are correct to perform for some common temporal logic formulae. We give some general soundness theorems, and apply this approach to period and phase constraints on the circadian clock and the cell cycle. The traces shown in Fig. 1, and detailed in Sect. 6, contain each several thousands of time-points. Computing the domains of the formula describing the period between each pair of successive peaks by polyhedral methods [13] becomes quite computationally expensive. In this appli-

cation, temporal logic patterns are used to compute the relevant characteristics of the experimental traces, and to measure the adequacy of the model to its specification on simulation traces. Speed-ups by several orders of magnitude are obtained by trace simplification, even when produced by smart numerical integration methods (e.g. Rosenbrock's implicit method), making trace simplification comparable with ad-hoc solvers.

2 Temporal Logic Patterns

The *Linear Time Logic* LTL is a temporal logic [6] which extends classical logic with modal operators for qualifying when a formula is true in a series of timed states. The temporal operators are **X** ("next", for at the next time point), **F** ("finally", for at some time point in the future), **G** ("globally", for at all time points in the future), **U** ("until", for a first formula must be true until a second one becomes true), and **W** (" weak until", a dual operator of **U**). These operators enjoy some simple duality properties, $\neg\mathbf{X}\phi = \mathbf{X}\neg\phi$, $\neg\mathbf{F}\phi = \mathbf{G}\neg\phi$, $\neg\mathbf{G}\phi = \mathbf{F}\neg\phi$, $\neg(\psi\ \mathbf{U}\ \phi) = (\neg\phi\ \mathbf{W}\ \neg\psi)$, $\neg(\psi\ \mathbf{W}\ \phi) = (\neg\psi\ \mathbf{U}\ \neg\phi)$, and we have $\mathbf{F}\phi = \text{true}\ \mathbf{U}\ \phi$, $\mathbf{G}\phi = \phi\ \mathbf{W}$ false.

In this paper we consider a first-order version of LTL, denoted by FO-LTL($\mathbb{R}_{\text{lin}}$), with variables and linear constraints over $\mathbb{R}$, and quantifiers. The grammar of FO-LTL($\mathbb{R}_{\text{lin}}$) formulae is defined as follows:
$\phi ::= \text{c} \mid \neg\phi \mid \phi \Rightarrow \psi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \exists x\ \phi \mid \forall x\ \phi \mid \mathbf{X}\phi \mid \mathbf{F}\phi \mid \mathbf{G}\phi \mid \phi\mathbf{U}\phi \mid \phi\mathbf{W}\phi$
where c denotes linear constraints between molecular concentrations (written with upper case letters) their first derivative (written dA/dt), free variables (written with lower case letters), real numbers, and the state time variable, denoted by *Time*; e.g., $\mathbf{F}(A < v)$ is an FO-LTL($\mathbb{R}_{\text{lin}}$) formula. To denote the value of state variable A in the state s_i we shall use a subscript notation such as A_{s_i}.

Temporal logic formulae are classically interpreted in a Kripke structure, i.e. a transition relation over a set of states such that each state has at least one successor [6]. In this paper, we consider finite traces obtained either by biological experiments, or by numerical integration. To give meaning to LTL formulae, a finite trace $(s_0, ..., s_n)$ is thus complemented in an infinite trace by adding a loop on the last state, $(s_0, ..., s_n, s_n, ...)$. The practical assumption behind this classical convention for interpreting temporal logic on finite traces [22] is that the time horizon considered is sufficiently long for properly evaluating the formulas of interest. We also replace the computed value of $\frac{dA}{dt}$ by 0 in the last state, in order to maintain the coherence between the concentrations and their derivatives. In this interpretation over finite traces, the formula $\mathbf{G}\phi$ is thus true in the last state if ϕ is true in the last state. The semantics of formulae containing free variables is given by the validity domains of the variables.

Definition 1. *The* validity domain $\mathcal{D}_{(s_0,...,s_n),\phi}$ *of the free variables of an FO-LTL($\mathbb{R}_{lin}$) formula ϕ on a finite trace $T = (s_0, ..., s_n)$, is a vector of least domains for the variables, noted $\mathcal{D}_{(s_0,...,s_n),\phi}$, satisfying the following equations:*

- $\mathcal{D}_{T,\phi} = \mathcal{D}^T_{s_0,\phi}$,

– $\mathcal{D}^T_{s_i,c(\boldsymbol{x})} = \{\boldsymbol{v} \in \mathbb{R}^k \mid s_i \models c[\boldsymbol{v}/\boldsymbol{x}]\}$ *for a constraint* $c(\boldsymbol{x})$,
– $\mathcal{D}^T_{s_i,\phi\wedge\psi} = \mathcal{D}^T_{s_i,\phi} \cap \mathcal{D}^T_{s_i,\psi}$, *and* $\mathcal{D}^T_{s_i,\phi\vee\psi} = \mathcal{D}^T_{s_i,\phi} \cup \mathcal{D}^T_{s_i,\psi}$,
– $\mathcal{D}^T_{s_i,\neg\phi} = \complement\, \mathcal{D}^T_{s_i,\phi}$,
– $\mathcal{D}^T_{s_i,\exists x\phi} = \Pi_x \mathcal{D}^T_{s_i,\phi}$, *and* $\mathcal{D}^T_{s_i,\forall x\phi} = \mathcal{D}^T_{s_i,\neg\exists x\neg\phi}$,
– $\mathcal{D}^T_{s_i,\mathbf{X}\phi} = \mathcal{D}^T_{s_{i+1},\phi}$ *if* $i < n$, *and* $\mathcal{D}^T_{s_n,\mathbf{X}\phi} = \mathcal{D}^T_{s_n,\phi}$,
– $\mathcal{D}^T_{s_i,\mathbf{F}\phi} = \bigcup_{j=i}^{n} \mathcal{D}^T_{s_j,\phi}$, *and* $\mathcal{D}^T_{s_i,\mathbf{G}\phi} = \bigcap_{j=i}^{n} \mathcal{D}^T_{s_j,\phi}$,
– $\mathcal{D}^T_{s_i,\phi\mathbf{U}\psi} = \bigcup_{j=i}^{n} (\mathcal{D}^T_{s_j,\psi} \cap \bigcap_{k=i}^{j-1} \mathcal{D}^T_{s_k,\phi})$.

where $\complement$ *is the set complement operator over domains, and* Π_x *is the domain projection operator out of* x*, restoring domain* $\mathbb{R}$ *for* x*, and the other operators are defined by duality.*

3 Trace Simplifications

The usual computation of the validity domains involves computing domains for each subformula on each point of the trace s_i. When dealing with temporal data coming from numerical integration, especially of stiff systems, n can be very high, which induces a high computational cost, $\mathcal{O}(n^k)$, where k is the number of variables. As mentionned in [14], and justified in the following sections of this paper, a practical solution to this issue involves simplifying the numerical trace without changing the generic domain solving algorithm. In this section we therefore define more precisely the formal framework for defining such trace simplifications.

Definition 2 (Trace simplification). *Let* T *be a finite trace* $(s_0, \ldots, s_n)$ *and* ϕ *an FO-LTL*($\mathbb{R}_{lin}$) *formula with constraints over the states of* T*.*

T' *is a* simplification *of* T *for* ϕ *at* i*, written* $T' \preceq^i_\phi T$ *if:*

– $T' = (s_{j_0}, \ldots, s_{j_k})$ *for* $J = \{j_0, \ldots, j_k\}$ *a subset of the indices* $\{0, \ldots, n\}$ *such that* $j_0 < \ldots < j_k$*, i.e.,* T' *is a subtrace of* T*;*
– $\mathcal{D}^T_{s_i,\phi} = \mathcal{D}^{T'}_{s_{j_i},\phi}$*, where* j_i *is the smallest index in* J *such that* $j_i \geq i$*, i.e. the validity domains on* T *at* i *and* T' *at* j_i *are equal.*

T' *is a simplification of* T *for* ϕ*, written* $T' \preceq_\phi T$ *when it is a simplification of* T *at* s_0*, i.e.,* $\mathcal{D}_{T,\phi} = \mathcal{D}_{T',\phi}$*.*

T' *is a* strict simplification *of* T *for* ϕ*, written* $T' \prec T$ *if* $J \subsetneq \{0, \ldots, n\}$*.*

T' *is an* optimal simplification *of* T *for* ϕ *if its cardinal is minimal in the set of the simplifications of* T *for* ϕ*.*

Property-driven reduction of the system under analysis is a technique that has been addressed many times in the history of computer science. In the framework of abstract interpretation [7], not only the states but also the transitions can be abstracted in a new system for simplifying the analysis of some given properties. The definition above can be seen as a particular instance of this framework where a subset of states on the trace is preserved without abstraction, and the transitions are abstracted accordingly to this subset. This abstraction reflects our motivation of computing exact validity domains for formula variables (no state domain abstraction) more efficiently (transition abstraction).

4 Examples

Most of the equations for $\mathcal{D}^T_{s_i,\phi}$ in Definition 1 are local, in the sense that they only need information about the state at s_i. One obvious case of simplification is when the unions or intersections involved in the domains for **F**, **G** and **U** can be computed on a strict subset of the points, sometimes even a singleton. Since it will come up often in the following examples, let us define a simple subtrace containing all the local extrema and the initial point of the trace.

Definition 3 (Extrema Subtrace). *Let $T = (s_0, \ldots, s_n)$ be a trace, T^e_x is the subtrace of T defined as follows:*

$$T^e_x = \{s_i \in T \mid (dx/dt)_{i-1} > 0 \wedge (dx/dt)_i \leq 0\} \\ \cup \{s_i \in T \mid (dx/dt)_{i-1} < 0 \wedge (dx/dt)_i \geq 0\} \cup \{s_0\}$$

We shall write $T^e = \bigcup_x T^e_x$

In the following examples, we will use the formulae given in [14] plus a few other ones, and for each, we will compute the corresponding domain and examine possible trace simplifications.

Example 1 (Minimal Amplitude).

Formula: $\phi = \exists v \mid \mathbf{F}(A < v) \wedge \mathbf{F}(A > v + a)$

Validity Domain Let s_{minA} and s_{maxA} be some points of the trace where A is respectively minimum and maximum.

$$\begin{aligned} \mathcal{D}_{T,\phi} &= \Pi_a(\mathcal{D}^T_{s_0,\mathbf{F}(A<v)} \cap \mathcal{D}^T_{s_0,\mathbf{F}(A>v+a)}) \\ &= \Pi_a((\bigcup_{i=0}^{n} \mathcal{D}^T_{s_i,A<v}) \cap (\bigcup_{i=0}^{n} \mathcal{D}^T_{s_j,A>v+a})) \qquad (*) \\ &= \Pi_a(\mathcal{D}^T_{s_{minA},A<v} \cap \mathcal{D}^T_{s_{maxA},A>v+a}) \qquad (*) \end{aligned}$$

Trace Simplification From the computation of the domain, equations marked with a $(*)$, one can see that both unions are actually equal to a single domain, only dependent on the state but not on T. Therefore any choice of s_{minA}, s_{maxA} leads to an optimal trace simplification T_J where $J = \{minA, maxA\}$.

Note that because of the semantic link between A and $\frac{dA}{dt}$, T^e_A contains s_{minA} and s_{maxA} and therefore will result in the same unions in the computation of the domain, hence T^e_A is a simplification of T for ϕ.

Example 2 (Threshold).

Formula: $\phi = \mathbf{F}(\mathit{Time} > 20 \wedge A < v)$

Validity Domain Let T be a trace $(s_0, \ldots, s_n)$ and $T_{>20}$ its subtrace on the points $J = \{0 \leq i \leq n \mid Time_{s_i} > 20\}$. As before, we chose some $s_{minA_{>20}}$, a point where A is minimum on $T_{>20}$.

$$\begin{aligned}
\mathcal{D}_{T,\phi} = \mathcal{D}^T_{s_0, \mathbf{F}(Time>20 \wedge A<v)} &= \bigcup_{i=0}^{n} \mathcal{D}^T_{s_i, Time>20 \wedge A<v} \\
&= \bigcup_{i=0}^{n} (\mathcal{D}^T_{s_i, Time>20} \cap \mathcal{D}^T_{s_i, A<v}) \qquad (*) \\
&= \bigcup_{i \in J} \mathcal{D}^T_{s_i, A<v} = \mathcal{D}^T_{s_{minA_{>20}}, A<v} \qquad (*)
\end{aligned}$$

Trace Simplification As shown by the marked equations, the single point $\{s_{minA_{>20}}\}$ is enough to compute the big union of the domain, it defines an optimal trace simplification of T for ϕ.

Notice that T^e_A is not a simplification unless it does contain a local minimum such that $Time > 20$: if that is not the case, e.g. always increasing trace, $s_{minA_{>20}}$ will be the first state after $Time = 20$, which is not a local extremum.

Example 3 (Crossing).

Formula: $\phi = \mathbf{F}(A > B \wedge \mathbf{X}(A \leq B \wedge Time = t))$

Validity Domain
$$\begin{aligned}
\mathcal{D}_{T,\phi} &= \bigcup_{i=0}^{n} (\mathcal{D}^T_{s_i, A_{s_i} > B_{s_i}} \cap (\mathcal{D}^T_{s_{i+1}, A_{s_i} \leq B_{s_i}} \cap \mathcal{D}^T_{s_{i+1}, Time=t})) \\
&= \bigcup_{i \in \{0,\ldots,n\}] \mid A_{s_i} > B_{s_i} \wedge A_{s_{i+1}} \leq B_{s_{i+1}}} \{Time_{s_{i+1}}\}
\end{aligned}$$

The computation above simply discards from the union the trace points where the intersection is empty because one of the two first members is empty.

Trace Simplification Once again, for any trace $T = (s_0, \ldots, s_n)$, the validity domain is a big union that can be restricted to the points of $J = \{i, i+1 \in \{0, \ldots, n\}] \mid A_{s_i} > B_{s_i} \wedge A_{s_{i+1}} \leq B_{s_{i+1}}\}$, which defines a simplification T_J of T for ϕ. As in Example 2, T^e_A is not a simplification of T for ϕ since it obviously misses the points at which *Time* has to be computed.

Example 4 (Peak).

Formula: $\phi = \mathbf{F}(\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge Time = t))$

Validity Domain The reasoning is the same as for Example 3.

$$\mathcal{D}_{T,\phi} = \mathcal{D}^T_{s_0,\phi} = \bigcup_{i=0}^{n} (\mathcal{D}^T_{s_i, \frac{dA}{dt}>0} \cap (\mathcal{D}^T_{s_{i+1}, \frac{dA}{dt}\leq 0} \cap \mathcal{D}^T_{s_{i+1}, Time=t}))$$
$$= \bigcup_{i\in\{0,\ldots,n\}]|(\frac{dA}{dt})_{s_i}>0\wedge(\frac{dA}{dt})_{s_{i+1}}\leq 0} \mathcal{D}^T_{s_{i+1}, Time=t}$$
$$= \bigcup_{i\in\{0,\ldots,n\}]|(\frac{dA}{dt})_{s_i}>0\wedge(\frac{dA}{dt})_{s_{i+1}}\leq 0} \{Time_{s_{i+1}}\}$$

Trace Simplification As above, for any trace $T = (s_0, \ldots, s_n)$, $J = \{i, i+1 \in \{0, \ldots, n\}] \mid \frac{dA}{dt}_{s_i} > 0 \wedge \frac{dA}{dt}_{s_{i+1}} \leq 0\}$ defines a simplification T_J of T for ϕ.

Note that T^e_A is also a simplification of T for ϕ since it contains all $i+1$ at which A_{s_i} is used and a predecessor with the right sign of the derivative, either s_0 or a nadir preceding the peak. Note also that $|T^e_A| \leq |T_J| + 2$ since there can be one nadir more than there are peaks, plus the origin s_0.

Example 5 (Period).

Formula:
$$\begin{aligned}\phi = &\exists(t1, t2) \mid p = t_2 - t_1 \wedge t_1 < t_2 \\ &\wedge \mathbf{F}(\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge Time = t1)) \\ &\wedge \mathbf{F}(\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge Time = t2)) \\ &\wedge \neg\exists t_3 \mid t_1 < t_3 < t_2 \wedge \mathbf{F}(\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge Time = t3))\end{aligned}$$

ϕ encodes the fact that t_1 and t_2 are peaks, with no peak in between.

Trace Simplification One can notice that the domain is formed of the same kind of union as in Example 4, repeated three times, and under top-level projections/intersections/complementations. Now, remark that a simplification for the formula of Example 4 will, by definition, allow to compute correctly the domains for all three **F** formulae, and therefore is a simplification for the compound ϕ. This is a special case of Theorem 1 detailed in the next section.

It follows that T_J of Example 4 and T^e_A are simplifications of T for ϕ.

Equivalent Formula:
$$\begin{aligned}\phi = \exists(t_1, t_2) \mid p = t_2 - t_1 \wedge \mathbf{F}(&\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge Time = t_1 \\ &\wedge (\frac{dA}{dt} \leq 0)\mathbf{U}(\frac{dA}{dt} > 0 \\ &\wedge ((\frac{dA}{dt} > 0)\mathbf{U}(\frac{dA}{dt} \leq 0 \wedge Time = t_2)))))\end{aligned}$$

Validity Domain Note first that the validity domain of the subformula $\psi = \frac{dA}{dt} > 0 \wedge ((\frac{dA}{dt} > 0)\mathbf{U}(\frac{dA}{dt} \leq 0 \wedge Time = t_2))$ is computed at each time point s_i like this:

$$\mathcal{D}^T_{s_i,\psi} = \mathcal{D}^T_{s_i,\frac{dA}{dt}>0} \cap \bigcup_{j=i}^{n} (\mathcal{D}^T_{s_j,\frac{dA}{dt}\leq 0 \wedge Time=t_2} \cap (\bigcap_{k=i}^{j-1} (\mathcal{D}^T_{s_k,\frac{dA}{dt}>0}))$$

Since $\mathcal{D}^T_{s_i,\frac{dA}{dt}>0}$ is either empty or equal to the whole space when $\frac{dA}{dt}_{s_i}$ is respectively negative or strictly positive, it holds that $D^T_{s_i,\psi}$ is empty if $\frac{dA}{dt}_{s_i} \leq 0$, otherwise:

$$\begin{aligned}
\mathcal{D}^T_{s_i,\psi} &= \bigcup_{j=i}^{n} (\mathcal{D}^T_{s_j,\frac{dA}{dt}\leq 0 \wedge Time=t_2} \cap (\bigcap_{k=i}^{j-1} (\mathcal{D}^T_{s_k,\frac{dA}{dt}>0})) \\
&= \bigcup_{j=i}^{n} (\mathcal{D}^T_{s_j,\frac{dA}{dt}\leq 0} \cap \mathcal{D}^T_{s_j,Time=t_2} \cap (\bigcap_{k=i}^{j-1} (\mathcal{D}^T_{s_k,\frac{dA}{dt}>0})) \\
&= \bigcup_{j\in\{i,\ldots,n\} | (\frac{dA}{dt})_{s_j}\leq 0 \wedge \forall k\in\{i,\ldots,j-1\},(\frac{dA}{dt})_{s_k}>0} \mathcal{D}^T_{s_j,Time=t_2} \\
&= \bigcup_{j\in\{i,\ldots,n\} | (\frac{dA}{dt})_{s_j}\leq 0 \wedge \forall k\in\{i,\ldots,j-1\},(\frac{dA}{dt})_{s_k}>0} \{Time_{s_j}\}
\end{aligned}$$

This union is in fact restricted to the first point s_j after $s_i$where $\frac{dA}{dt}$ is no longer strictly positive.

With the same reasoning, the validity domain for the whole formula becomes:

$$\mathcal{D}_{T,\phi} = \bigcup_{(i,j)\in P}^{n} \{Time_{s_{j+1}} - Time_{s_{t+1}}\}$$

where P is the set of pairs of successive peaks:

$$\begin{aligned}
P = \{(i,j) \mid &(\frac{dA}{dt})_{s_i} > 0 \wedge (\frac{dA}{dt})_{s_{i+1}} \leq 0 \wedge (\frac{dA}{dt})_{s_j} > 0 \wedge (\frac{dA}{dt})_{s_{j+1}} \leq 0 \\
&\wedge \neg\exists i < k < j \mid (\frac{dA}{dt})_{s_k} > 0 \wedge (\frac{dA}{dt})_{s_{k+1}} \leq 0\}
\end{aligned}$$

T^e_A is a simplification of T for ϕ since it contains all the peaks of the trace.

5 General Simplification Results

Example 5 shows that if one can simplify subformulae, one might obtain a simplification for the whole formula. Indeed, with some hypotheses, the patterns described in the previous section can actually be composed.

The first theorem simply notices that if the highest-level temporal subformulae have a simplification, it also holds for the compound formula.

Theorem 1. *Let T be a trace containing a state s_i, ϕ and ψ two formulae and T' such that $T' \preceq^i_\phi T$ and $T' \preceq^i_\psi T$. Then $T' \preceq^i_\mu T$ for μ equal to*

$$\phi \wedge \psi \text{ or } \phi \vee \psi \text{ or } \neg\phi \text{ or } \exists x\phi \text{ or } \forall x\phi$$

Proof. We have $\mathcal{D}^T_{s_i,\phi} = \mathcal{D}^{T'}_{s_{j_i},\phi}$ and the same for ψ, therefore $\mathcal{D}^T_{s_i,\phi\wedge\psi} = \mathcal{D}^T_{s_i,\phi} \cap \mathcal{D}^T_{s_i,\psi} = \mathcal{D}^{T'}_{s_{j_i},\phi} \cap \mathcal{D}^{T'}_{s_{j_i},\psi} = \mathcal{D}^{T'}_{s_{j_i},\phi\wedge\psi}$ and the same for the other operators. □

Note that it is not true that if T' is a simplification for ϕ and T'' a simplification for ψ, then the union of the points in T' and T'' defines a simplification for $\phi \vee \psi$: indeed, adding points to a simplification can invalidate it, for instance if the formula contains $\mathbf{X}$. Now, remark that if a subtrace contains extreme domains, it is a simplification for $\mathbf{F}$ and $\mathbf{G}$:

Theorem 2. *Let $T = (s_0, \ldots, s_n)$ be a trace, ϕ a formula and $T' = T_J$ a substrace of T such that:*
$\forall j \in J, T' \preceq^j_\phi T$ and $\forall 0 \leq i \leq n, \exists j \in J,\ \mathcal{D}^T_{s_i,\phi} \subset \mathcal{D}^{T'}_{s_j,\phi}$ (resp. $\mathcal{D}^T_{s_i,\phi} \supset \mathcal{D}^{T'}_{s_j,\phi}$)
then: $T' \preceq_{\mathbf{F}\phi} T$ (resp. $T' \preceq_{\mathbf{G}\phi} T$)

Proof. We have, $\forall 0 \leq i \leq n,\ \mathcal{D}^T_{s_i,\phi} \subset \mathcal{D}^{T'}_{s_j,\phi}$ it follows that $\bigcup_{i=0}^n \mathcal{D}^T_{s_i,\phi} \subset \bigcup_{j\in J} \mathcal{D}^{T'}_{s_j,\phi}$. The other inclusion is immediate since J is a subset of the indices $\{0, \ldots, n\}$ and we have simplification for ϕ at those indices. The result for $\mathbf{G}$ is obtained similarly. □

Consider now the case of formulae without free variables, their domain is either empty or full, which can be taken advantage of:

Corollary 1. *Let $T = (s_0, \ldots, s_n)$ be a trace, ϕ a formula, c a constraint without free variables and J_c be the subset of indices defined by $J_c = \{0 \leq i \leq n \mid s_i \models c\}$*
If $\forall i \in J_c,\ T_{J_c} \preceq^i_\phi T$ then $T_{J_c} \preceq_{\mathbf{F}(c\wedge\phi)} T$ and $T_{J_c} \preceq_{\mathbf{G}(\neg c\vee\phi)} T$

Proof. Let us prove the result for $\mathbf{F}$, then Thm. 1 can give it for $\mathbf{G}$. We will simply apply the above theorem to $c \wedge \phi$. The first hypothesis of Thm. 2 is satisfied by T_{J_c} since $T_{J_c} \preceq^i_\phi T \Rightarrow T_{J_c} \preceq^i_{c\wedge\phi} T$. For the second hypothesis, it is enough to notice that if $i \notin J_c$ then $\mathcal{D}^T_{s_i,c\wedge\phi} = \mathcal{D}^T_{s_i,c} \cap \mathcal{D}^T_{s_i,\phi} = \emptyset$. □

Note that in general $\mathbf{F}\phi\wedge\psi$ is not easy to simplify. On the contrary $\mathcal{D}_{T,\mathbf{F}(\phi\vee\psi)} = \mathcal{D}_{T,\mathbf{F}(\phi)\vee\mathbf{F}(\psi)}$ which can benefit from Theorem 1.

In many cases it is worth noticing that T^e_A satisfies the hypothesis of Thm. 2 for any formula $\mathbf{F}(\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge c))$.

Proposition 1. *Let $\phi = \mathbf{F}(\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge c))$ be a formula, $T^e_A \preceq_\phi T$*

Proof. We will apply Thm. 2. First note that for any extremum j in T^e_A we have $T^e_A \preceq^j_\phi T$. Indeed, s_0 is in T^e_A but will not be used to compute $\mathcal{D}_c$, on the other hand it ensures that even the first extremum does have a predecessor of the correct sign for the derivative. Now, notice that $\mathcal{D}^T_{s_i,\phi}$ will be empty at each point not a predecessor of a state of T^e_A. At those points the domain on T is the same as that at the preceding extremum (or s_0 for the first) on T^e_A. This enforces the inclusion needed for the second hypothesis of Thm. 2. □

Taken together, these results prove all the simplifications of the previous examples except the second formula of Example 5, which is a deeply nested formula with **U** that relies on the semantics of the *Time* variable.

6 Evaluation on Oscillation Constraints between the Cell Cycle and Circadian Clock

Cellular rhythms represent an interesting field of research for systems biology, where models should satisfy qualitative properties like oscillations, synchronization among elements, and stability, as well as quantitative properties on the lengths of the oscillations and phases. FO-LTL($\mathbb{R}_{\text{lin}}$) formulae are particularly adequate to constraint biological oscillators models after these considerations.

We illustrate the use of FO-LTL($\mathbb{R}_{\text{lin}}$) constraints on a coupled model of the cell cycle and the circadian clock, which are two such biological oscillators also inter-regulated through clock-controlled cell cycle components. This gives rise to complex behaviors as suggested in a detailed study by Nagoshi et al. [12].

We use a reference model of the mammalian circadian clock [16] and a model of a generic cell cycle oscillator focusing on the G2/M transition [19]. A molecular link between the two systems is introduced with the regulation of the cell cycle kinase *Wee1* by the clock gene *bmal1* [18].

Figure 1 shows two examples of traces obtained with different sets of parameters values, simulated over a time horizon of 200 hours. They give different dynamical behaviors with correct oscillations of the components on the first one, and damped oscillations on the other. By applying specifications expressed with the temporal logic formalism on these traces, we investigate the behavior of the system, or evaluate how far each set of parameter values is from reproducing desired properties in a calibrating process.

The chosen FO-LTL($\mathbb{R}_{\text{lin}}$) formulae express constraints on the periods of each module, phases between the components, as well as stability constraints. Each formula accept T^e_M as a simplification of T, where M is the set of molecules appearing in the formula. They correspond to patterns associated to dedicated solvers defined in [14] and listed below with the corresponding properties. Detailed formulae are given in Appendix B with justifications for the simplifications.

- Constraints on the amplitude: *MinAmpl(A,min)*. This constraints the molecule A to an amplitude of at least *min*.
- Constraints on the period: *DistanceSuccPeaks(A,d)* specifies that there should be two successive peaks of the molecule A distant by *d*. The results for the evaluation on the first trace, computed either with the FO-LTL($\mathbb{R}_{\text{lin}}$) formula and the generic solver or with the *ad hoc* pattern and dedicated solver are the same and shown in Appendix A. This gives an example of information extraction from a trace with a FO-LTL($\mathbb{R}_{\text{lin}}$) formula.
- Constraints on the phases: *DistancePeaks(A,B,d)*. Here d take as values the possible distances between a peak of A and the following peak of B.

– Stability constraints on the oscillations: the specification *MaxDiffDistance-Peaks(A,d)* ensures that two successive peak-to-peak distances are not too different, with a maximum difference of d, so that the oscillations of the molecule have a relative regularity over time. A second stability constraint, *MaxDiffDistancePeaks(A,d)*, constraints the differences between the peak amplitudes, and is thus useful to filter out damped oscillations. The evaluation of *MaxDiffAmplPeaks(PerCry,d)* on the trace gives $[d > 8.48801e-05]$ as the validity domain for the first trace and $[d > 1.90466]$ for the second trace. Thus the evaluation of the constraint extracts the maximum difference in amplitudes between two successive peaks, and this result can be used as a penalty for the set of parameter values that result in damped oscillations.

We apply these constraints to the traces presented above, before and after performing the generic trace simplification where the trace T is replaced by the trace T^e_M, that is T^e_{PerCry} for all constrains in Table 1, except for *Distance-Peaks(MPF,PerCry)* where the simplified trace is $T^e_{MPF,PerCry}$.

The initial traces are obtained with two different integration methods:

– In Biocham the default simulation method is the Rosenbrock's numerical integration method. This implicit method with variable step-size avoids generating too many points and does an impressively good job in producing relatively sparse traces. With this method the first trace counts 971 point, 18 of which are kept in the simplified trace T^e_{PerCry} and 34 in $T^e_{MPF,PerCry}$. The second trace T counts 1047 points, T^e_{PerCry} counts 35 points and $T^e_{MPF,PerCry}$ counts 58 points. Since the initial traces have reasonable sizes the computing times for the simplifications are short: between 8 and 16ms.
– However in some cases, the Rosenbrock method is less adequate than other non-adaptive methods. For example, this is the case when the model involves events, since the approximation done for numerical integration with big steps, may not be valid for determining when an event becomes true. Therefore we also consider the fourth order Runge-Kutta method with a fixed step size. With this method, the trace optimisation is all the more beneficial since the traces originally count more points: 20002 points here for a time horizon of 200 hours. However the same trace simplifications take longer: around 160ms for T^e_{PerCry} and 250ms for $T^e_{MPF,PerCry}$.

The execution times are compared in Table 1 where each constraint is identified by the equivalent pattern. We compare the evaluation of the constraints on a trace with a high number of points (fixed Runge-Kutta method) or a reduced size (adaptive Rosenbrock method), and either complete or simplified. Furthermore the generic solver is compared to the dedicated solvers defined in [14].

Table 1 clearly shows that trace simplification provides a faster evaluation for all constraints on all traces, with a speed-up up to 100 fold for the more complex ones. The dedicated solvers benefit as well from this speed-up, however it has to be noted that applying the dedicated solver on the full trace is faster than the time needed for the trace simplification in this example. Although the simplification can be done just once on a trace that can be then evaluated repeatedly

Table 1. Computing time (in ms) for the validity domain of different formula patterns. Comparison between the first and second parameter sets, with variable or fixed step-size over 200h, before (**Bef.**) and after (**Aft.**) simplification.

		First trace				**Second trace**			
		variable		fixed		variable		fixed	
Formula	**Solver**	**Bef.**	**Aft.**	**Bef.**	**Aft.**	**Bef.**	**Aft.**	**Bef.**	**Aft.**
Reached(PerCry)	generic	12	0	260	4	12	0	204	0
	dedicated	0	0	16	0	4	0	16	0
MinAmpl(PerCry)	generic	132	0	2728	0	132	4	2516	4
	dedicated	0	0	16	0	4	0	16	0
LocalMax(PerCry)	generic	64	0	1308	4	72	4	1316	4
	dedicated	0	0	36	8	4	0	44	4
DistancePeaks(PerCry)	generic	512	12	9584	12	708	80	12373	104
	dedicated	4	4	40	8	32	28	80	48
DistanceSuccPeaks(PerCry)	generic	532	12	10980	12	1188	36	23101	156
	dedicated	4	0	40	8	4	0	28	4
MaxDiffDistancePeaks(PerCry)	generic	1700	32	34818	32	3056	96	60776	108
	dedicated	0	0	36	0	4	0	52	20
DistancePeaks(MPF,PerCry)	generic	456	16	9332	16	496	32	9365	32
	dedicated	4	4	68	12	4	0	76	20

for different patterns, the number of evaluations would have to be unlikely high for any real benefit. In contrast, the time gain obtained with the combined use of the trace simplification and the generic solver is clear. This suggests that the trace simplification is a good strategy when the desired constraint is not covered by the patterns with dedicated solvers, provided that the FO-LTL($\mathbb{R}_{\text{lin}}$) formula accepts a good trace simplification accordingly with the theorems presented in Sect. 3.

7 Conclusion

We have shown that trace simplifications can result in speed-ups by several orders of magnitude for the evaluation of temporal logic constraints. In particular we have given some general conditions on the syntax of the formulae under which it is correct to keep in the trace only the time points corresponding to the local extrema of the molecules, or the crossing points between molecular concentrations.

On an application concerning the modeling of the coupling between the circadian clock and the cell cycle, we have shown that temporal logic patterns provide an elegant way to extract information on the periods and phases from numerical traces, and to use these formulae as constraints for parameter search. On simulation traces, the speedup obtained in computation time was by several orders of magnitude, even on relatively sparse simulation traces obtained by Rosenbrock's implicit method for numerical integration.

The trace simplifications described in this paper are implemented in Biocham release 3.6.

Acknowledgements This work has been supported by the French OSEO Biointelligence project. We acknowledge discussions with our partners at Dassault-Systèmes and the CMSB reviewers for their remarks.

References

1. M. Antoniotti, A. Policriti, N. Ugel, and B. Mishra. Model building and model checking for biochemical processes. *Cell Biochemistry and Biophysics*, 38:271–286, 2003.
2. G. Batt, M. Page, I. Cantone, G. Goessler, P. Monteiro, and H. de Jong. Efficient parameter search for qualitative models of regulatory networks using symbolic model checking. *Bioinformatics*, 26(18):i603–i610, 2010.
3. G. Batt, D. Ropers, H. de Jong, J. Geiselmann, R. Mateescu, M. Page, and D. Schneider. Validation of qualitative models of genetic regulatory networks by model checking : Analysis of the nutritional stress response in *Escherichia coli*. *Bioinformatics*, 21(Suppl.1):i19–i28, 2005.
4. G. Bernot, J.-P. Comet, A. Richard, and J. Guespin. A fruitful application of formal methods to biological regulatory networks: Extending Thomas' asynchronous logical approach with temporal logic. *Journal of Theoretical Biology*, 229(3):339–347, 2004.
5. N. Chabrier and F. Fages. Symbolic model checking of biochemical networks. In C. Priami, editor, *CMSB'03: Proceedings of the first workshop on Computational Methods in Systems Biology*, volume 2602 of *Lecture Notes in Computer Science*, pages 149–162, Rovereto, Italy, Mar. 2003. Springer-Verlag.
6. E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. MIT Press, 1999.
7. P. Cousot and R. Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *POPL'77: Proceedings of the 6th ACM Symposium on Principles of Programming Languages*, pages 238–252, New York, 1977. ACM Press. Los Angeles.
8. H. de Jong, J.-L. Gouzé, C. Hernandez, M. Page, T. Sari, and J. Geiselmann. Qualitative simulation of genetic regulatory networks using piecewise-linear models. *Bulletin of Mathematical Biology*, 66(2):301–340, 2004.
9. A. Donzé, T. Ferrère, and O. Maler. Efficient robust monitoring for STL. In *25th International Conference on Computer Aided Verification, CAV'13*, Berkeley, USA, 2013.
10. A. Donzé and O. Maler. Robust satisfaction of temporal logic over real-valued signals. In *FORMATS 2010*, volume 6246 of *Lecture Notes in Computer Science*, pages 92–106. Springer-Verlag, 2010.
11. S. Eker, M. Knapp, K. Laderoute, P. Lincoln, J. Meseguer, and M. K. Sönmez. Pathway logic: Symbolic analysis of biological signaling. In *Proceedings of the seventh Pacific Symposium on Biocomputing*, pages 400–412, Jan. 2002.
12. N. Emi, S. Camille, B. Christoph, L. Thierry, N. Felix, and U. Schibler. Circadian gene expression in individual fibroblasts: cell-autonomous and self-sustained oscillators pass time to daughter cells. *Cell*, 119:693–705, 2004.
13. F. Fages and A. Rizk. On temporal logic constraint solving for the analysis of numerical data time series. *Theoretical Computer Science*, 408(1):55–65, Nov. 2008.
14. F. Fages and P. Traynard. Temporal logic modeling of dynamical behaviors: First-order patterns and solvers. In L. F. del Cerro and K. Inoue, editors, *Logical Modeling of Biological Systems*, chapter 8, pages 307–338. ISTE Ltd, 2014.

15. D. Heitzler, G. Durand, N. Gallay, A. Rizk, S. Ahn, J. Kim, J. D. Violin, L. Dupuy, C. Gauthier, V. Piketty, P. Crépieux, A. Poupon, F. Clément, F. Fages, R. J. Lefkowitz, and E. Reiter. Competing G protein-coupled receptor kinases balance G protein and β-arrestin signaling. *Molecular Systems Biology*, 8(590), June 2012.
16. J.-C. Leloup and A. Goldbeter. Toward a detailed computational model for the mammalian circadian clock. *Proceedings of the National Academy of Sciences*, 100:7051–7056, 2003.
17. N. Markey and P. Schnoebelen. Model checking a path. In *CONCUR 2003*, volume 2761 of *Lecture Notes in Computer Science*, pages 251–265. Springer-Verlag, 2003.
18. T. Matsuo, S. Yamaguchi, S. Mitsui, A. Emi, F. Shimoda, and H. Okamura. Control mechanism of the circadian clock for timing of cell division in vivo. *Science*, 302(5643):255–259, Oct. 2003.
19. Z. Qu, W. R. MacLellan, and J. N. Weiss. Dynamics of the cell cycle: checkpoints, sizers, and timers. *Biophysics Journal*, 85(6):3600–3611, 2003.
20. A. Rizk, G. Batt, F. Fages, and S. Soliman. A general computational method for robustness analysis with applications to synthetic gene networks. *Bioinformatics*, 12(25):il69–il78, June 2009.
21. A. Rizk, G. Batt, F. Fages, and S. Soliman. Continuous valuations of temporal logic specifications with applications to parameter optimization and robustness measures. *Theoretical Computer Science*, 412(26):2827–2839, 2011.
22. G. Roşu and K. Havelund. Rewriting-based techniques for runtime verification. *Automated Software Engineering*, 12(2):151–197, 2005.
23. S. Stoma, A. Donzé, F. Bertaux, O. Maler, and G. Batt. STL-based analysis of TRAIL-induced apoptosis challenges the notion of type I/type II cell line classification. *PLoS Computational Biology*, 9(5):e1003056, May 2013.

A Example of computation with both the generic solver and a dedicated one

```
domains(t2-t1=d & F(d([CRY_nucl-PER_nucl])/dt>0 & X(Time=t1 & d([CRY_nucl-PER_nucl
    ])/dt=<0 & (d([CRY_nucl-PER_nucl])/dt=<0) U (d([CRY_nucl-PER_nucl])/dt>0 & ((
    d([CRY_nucl-PER_nucl])/dt>0) U (d([CRY_nucl-PER_nucl])/dt=<0 & Time=t2))))))).
Domain computed in 532 ms
d = 24.6095, t1 = 15.2848, t2 = 39.8944
| d = 24.7193, t1 = 39.8944, t2 = 64.6137
| d = 25.1225, t1 = 64.6137, t2 = 89.7362
| d = 24.7623, t1 = 89.7362, t2 = 114.499
| d = 24.7984, t1 = 114.499, t2 = 139.297
| d = 24.8047, t1 = 139.297, t2 = 164.102
| d = 24.7704, t1 = 164.102, t2 = 188.872
```

```
domains(distanceSuccPeaks([CRY_nucl-PER_nucl],[d])).
Domain computed in 4 ms
d = 24.6095
| d = 24.7193
| d = 25.1225
| d = 24.7623
| d = 24.7984
| d = 24.8047
| d = 24.7704
```

B Oscillation constraints

Constraints on the amplitude As shown in Ex. 1, the following formula, accepting T_A^e as a simplification of T, ensures that a molecule A has an amplitude of at least *min*: $\phi = \exists v \mid \mathbf{F}(A < v) \wedge \mathbf{F}(A > v + min)$.

It is equivalent to the pattern *MinAmpl(A,min)* described in [14] and associated to a specific solver which computes the amplitude of A directly from the trace.

Constraints on the period This formula extracts the distances between successive peaks:

$$\begin{aligned}\phi = \exists(t_1, t_2) \mid d = t_2 - t_1 \wedge \mathbf{F}(\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge Time = t_1 \\ \wedge (\frac{dA}{dt} \leq 0)\mathbf{U}(\frac{dA}{dt} > 0 \\ \wedge ((\frac{dA}{dt} > 0)\mathbf{U}(\frac{dA}{dt} \leq 0 \wedge Time = t_2))))))\end{aligned}$$

This formula accepts T_A^e as a simplification of T, as shown in Ex. 4 and with Thm. 1. It is equivalent to the pattern *DistanceSuccPeaks(A,d)*. The specific solver associated to this pattern computes the list of peaks of A directly from the trace and exhibits the possible distances between two successive peaks. Computing the validity domain of this formula enables to extract each peak-to-peak distance from the trace, giving an estimation of the period of the oscillations.

Constraints on the phases

$$\begin{aligned}\phi = \exists(t_1, t_2) \mid t2 - t1 = d \wedge \mathbf{F}(\frac{dA}{dt} \geq 0 \wedge \mathbf{X}(\frac{dA}{dt} < 0 \wedge Time = t1)) \\ \wedge \mathbf{F}(\frac{dB}{dt} \geq 0 \wedge \mathbf{X}(\frac{dB}{dt} < 0 \wedge Time = t2))\end{aligned}$$

corresponds to *DistancePeaks([A,B],d)*. $T_A^e \cup T_B^e$ is a simplification of T for ϕ.

Stability constraints The following formula constraints two successive peak-to-peak distances to be similar by setting a maximum for the difference between

the two distances.

$$\begin{aligned}\phi = \exists(t_1, t_2, t_3)\ | & t2 - t1 = d1 \wedge t3 - t2 = d2 \wedge d2 - d1 \leq d \wedge d1 - d2 \leq d \\ & \wedge \mathbf{F}(\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge Time = t_1 \\ & \wedge (\frac{dA}{dt} \leq 0)\mathbf{U}(\frac{dA}{dt} > 0 \\ & \wedge ((\frac{dA}{dt} > 0)\mathbf{U}(\frac{dA}{dt} \leq 0 \wedge Time = t_2 \\ & \wedge (\frac{dA}{dt} \leq 0)\mathbf{U}(\frac{dA}{dt} > 0 \\ & \wedge ((\frac{dA}{dt} > 0)\mathbf{U}(\frac{dA}{dt} \leq 0 \wedge Time = t_3)))))))\end{aligned}$$

This formula accepts T_A^e as a simplification of T and the equivalent pattern is *MaxDiffDistancePeaks(A,d)*. A similar formula, useful to filter out damped oscillations, constraints the differences between the peak amplitudes, and is equivalent to the pattern *MaxDiffAmplPeaks(A,d)*.