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

CMSB 2014
Pauline Traynard and Francois Fages and Sylvain Soliman
Modeling the coupling between the cell cycle and the circadian clock

**Context:** Optimizing cancer treatment with chronotherapy

Experimental observations on periods and phases suggest bidirectional influence between cell divisions and the autonomous cellular circadian clock.

What are the mechanisms behind these observations?

Model building assisted with formal methods (model calibration)

Predictions: mechanisms and perturbations, optimization

Bidirectional coupled model of the cell cycle and the circadian clock
Temporal logic specifications

- Dynamical behaviors for oscillatory systems:
  - period, amplitude, phase
  - oscillations regularity

- Formalised with temporal logic:
  Ex: period \( \phi = \exists (t1, t2) \mid p = t2 - t1 \wedge t1 < t2 \)

\[
\begin{align*}
\wedge F\left( \frac{dA}{dt} > 0 \wedge X\left( \frac{dA}{dt} \leq 0 \wedge Time = t1 \right) \right) \\
\wedge F\left( \frac{dA}{dt} > 0 \wedge X\left( \frac{dA}{dt} \leq 0 \wedge Time = t2 \right) \right)
\end{align*}
\]

- Applications:
  - Data analysis: extracting meaningful information from a trace
  - Model checking: verifying that a model satisfies some constraints
  - Model analysis: comparing how the properties of a model evolve when some parameters vary
  - Parameter inference: continuous satisfaction degree of a temporal logic formula, powerful optimization algorithm CMA-ES

Result:
- Possible values: \( p = 23 \mid p=24.5 \)
- Satisfaction degree (with objective \( p=24 \)): 0.1

http://lifeware.inria.fr/Biocham/
Validation domain computing algorithm

Generic algorithm:

- **Decomposition** of $\phi$ in **sub-formulas**
- For each constraint and each time point, computing of a **domain** of possible variables
- **Combination** of the subdomains with the logical operators:
  - Operator $F$ (finally) $\rightarrow$ union: $\mathcal{D}_{s_i}^{T}F_{\phi} = \bigcup_{j=i}^{n} \mathcal{D}_{s_j}^{T} \phi$
  - Operator $G$ (globally) $\rightarrow$ intersection: $\mathcal{D}_{s_i}^{T}G_{\phi} = \bigcap_{j=i}^{n} \mathcal{D}_{s_j}^{T} \phi$
  - Operator $X$ (next) $\rightarrow$ next domain if valid: $\mathcal{D}_{s_i}^{T}X_{\phi} = \mathcal{D}_{s_{i+1}}^{T} \phi$
  - Operator $U$ (until) $\rightarrow$ union of intersections: $\mathcal{D}_{s_i}^{T}U_{\psi} = \bigcup_{j=i}^{n} (\mathcal{D}_{s_j}^{T} \psi \cap \bigcap_{k=i}^{j-1} \mathcal{D}_{s_k}^{T} \phi)$

- Domain for $\phi = \text{combedated domain for the first point of the trace}$

\[ \phi = F( [A] > s ) \]

\[ D_{\phi} = \{ s < \text{max}[A] \} \]

**Computational cost:** up to $O(n^v)$
\( (v = \text{number of variables}) \)

How to find a simplified trace that will keep the same validity domain?
Validation domain computing algorithm

Generic algorithm:

- **Decomposition** of $\phi$ in **sub-formulas**
- For each constraint and each time point, computing of a **domain** of possible variables
- **Combination** of the subdomains with the logical operators:
  - Operator $\mathbf{F}$ (finally) $\rightarrow$ union: $\mathcal{D}_{s_i}^T, \mathbf{F}\phi = \bigcup_{j=i}^{n} \mathcal{D}_{s_j}^T, \phi$
  - Operator $\mathbf{G}$ (globally) $\rightarrow$ intersection: $\mathcal{D}_{s_i}^T, \mathbf{G}\phi = \bigcap_{j=i}^{n} \mathcal{D}_{s_j}^T, \phi$
  - Operator $\mathbf{X}$ (next) $\rightarrow$ next domain if valid: $\mathcal{D}_{s_i}^T, \mathbf{X}\phi = \mathcal{D}_{s_{i+1}}^T, \phi$
  - Operator $\mathbf{U}$ (until) $\rightarrow$ union of intersections: $\mathcal{D}_{s_i}^T, \phi \mathbf{U} \psi = \bigcup_{j=i}^{n} (\mathcal{D}_{s_j}^T, \psi \cap \bigcap_{k=i}^{j-1} \mathcal{D}_{s_k}^T, \phi)$
- Domain for $\phi =$ combined domain for the **first point** of the trace

$\phi = \mathbf{F}( [A] > s )$

$\mathcal{D}_\phi = \{ s < \text{max}[A] \}$

**Computational cost:** up to $O(n^v)$
($v =$ number of variables)

How to find a simplified trace that will keep the same validity domain?
Validation domain computing algorithm

Generic algorithm:

- **Decomposition** of φ in *sub-formulas*
- For each constraint and each time point, computing of a **domain** of possible variables
- **Combination** of the subdomains with the logical operators:
  - Operator \( F \) (finally) → union: \( \mathcal{D}_{s_i}^T, F \phi = \bigcup_{j=i}^{n} \mathcal{D}_{s_j}^T, \phi \)
  - Operator \( G \) (globally) → intersection: \( \mathcal{D}_{s_i}^T, G \phi = \bigcap_{j=i}^{n} \mathcal{D}_{s_j}^T, \phi \)
  - Operator \( X \) (next) → next domain if valid: \( \mathcal{D}_{s_i}^T, X \phi = \mathcal{D}_{s_{i+1}}^T, \phi \)
  - Operator \( U \) (until) → union of intersections: \( \mathcal{D}_{s_i}^T, \phi U \psi = \bigcup_{j=i}^{n} (\mathcal{D}_{s_j}^T, \psi \cap \bigcap_{k=i}^{j-1} \mathcal{D}_{s_k}^T, \phi) \)
- Domain for \( \phi = \) combinated domain for the **first point** of the trace

\[ \phi = F(\ [A] > s ) \]
\[ D_\phi = \{ s < \text{max}[A] \} \]

**Computational cost:** up to \( O(n^v) \)
(\( v = \) number of variables)

How to find a simplified trace that will keep the same validity domain?
Validation domain computing algorithm

Generic algorithm:

• **Decomposition** of $\phi$ in **sub-formulas**
• For each constraint and each time point, computing of a **domain** of possible variables
• **Combination** of the subdomains with the logical operators:
  - Operator $F$ (finally) $\rightarrow$ union: $\mathcal{D}^T_{s_i}, F_\phi = \bigcup_{j=i}^n \mathcal{D}^T_{s_j}, \phi$
  - Operator $G$ (globally) $\rightarrow$ intersection: $\mathcal{D}^T_{s_i}, G_\phi = \bigcap_{j=i}^n \mathcal{D}^T_{s_j}, \phi$
  - Operator $X$ (next) $\rightarrow$ next domain if valid: $\mathcal{D}^T_{s_i}, X_\phi = \mathcal{D}^T_{s_{i+1}}, \phi$
  - Operator $U$ (until) $\rightarrow$ union of intersections: $\mathcal{D}^T_{s_i}, \phi U_\psi = \bigcup_{j=i}^n (\mathcal{D}^T_{s_j}, \psi \cap \bigcap_{k=i}^{j-1} \mathcal{D}^T_{s_k}, \phi)$
• Domain for $\phi = \text{combined domain for the first point of the trace}$

$\phi = F([A] > s)$

$D_\phi = \{ s < \max[A] \}$

**Computational cost:** up to $O(n^v)$
($v =$ number of variables)

How to find a simplified trace that will keep the same validity domain?
**Validation domain computing algorithm**

**Generic algorithm:**

- **Decomposition** of $\phi$ in **sub-formulas**
- For each constraint and each time point, computing of a **domain** of possible variables
- **Combination** of the subdomains with the logical operators:
  - Operator $F$ (finally) $\rightarrow$ union: $D_{s_i}^T, F\phi = \bigcup_{j=i}^n D_{s_j}^T, \phi$
  - Operator $G$ (globally) $\rightarrow$ intersection: $D_{s_i}^T, G\phi = \bigcap_{j=i}^n D_{s_j}^T, \phi$
  - Operator $X$ (next) $\rightarrow$ next domain if valid: $D_{s_i}^T, X\phi = D_{s_{i+1}}^T, \phi$
  - Operator $U$ (until) $\rightarrow$ union of intersections: $D_{s_i}^T, \phi U \psi = \bigcup_{j=i}^n (D_{s_j}^T, \psi \cap \bigcap_{k=i}^{j-1} D_{s_k}^T, \phi)$
- Domain for $\phi = \text{combined domain for the first point of the trace}$

**Computational cost:** up to $O(n^v)$
($v =$ number of variables)

How to find a simplified trace that will keep the same validity domain?
Generic algorithm:

- **Decomposition** of $\phi$ in **sub-formulas**
- For each constraint and each time point, computing of a **domain** of possible variables
- **Combination** of the subdomains with the logical operators:
  - Operator $F$ (finally) $\rightarrow$ union: $\bigcup_{j=i}^{n} D_{s_j, \phi}$
  - Operator $G$ (globally) $\rightarrow$ intersection: $D_{s_i, \phi} = \bigcap_{j=i}^{n} D_{s_j, \phi}$
  - Operator $X$ (next) $\rightarrow$ next domain if valid: $D_{s_i, \phi} = D_{s_{i+1}, \phi}$
  - Operator $U$ (until) $\rightarrow$ union of intersections: $D_{s_i, \phi} = \bigcup_{j=i}^{n} (D_{s_j, \psi} \cap \bigcap_{k=i}^{j-1} D_{s_k, \phi})$
- Domain for $\phi = \text{combed domain for the first point of the trace}$

**Validation domain computing algorithm**

$\phi = F([A] > s)$

$D_\phi = \{ s < \text{max}[A] \}$

Computational cost: up to $O(n^v)$
($v = \text{number of variables}$)

How to find a simplified trace that will keep the same validity domain?
Validation domain computing algorithm

Generic algorithm:

- **Decomposition** of \( \phi \) in **sub-formulas**
- For each constraint and each time point, computing of a **domain** of possible variables
- **Combination** of the subdomains with the logical operators:
  - Operator \( F \) (finally) → union: \( D_{s_i}^T, F_{\phi} = \bigcup_{j=i}^{n} D_{s_j}^T, \phi \)
  - Operator \( G \) (globally) → intersection: \( D_{s_i}^T, G_{\phi} = \bigcap_{j=i}^{n} D_{s_j}^T, \phi \)
  - Operator \( X \) (next) → next domain if valid: \( D_{s_i}^T, X_{\phi} = D_{s_{i+1}}^T, \phi \)
  - Operator \( U \) (until) → union of intersections: \( D_{s_i}^T, \phi U_{\psi} = \bigcup_{j=i}^{n} (D_{s_j}^T, \psi \cap \bigcap_{k=i}^{j-1} D_{s_k}^T, \phi) \)

- Domain for \( \phi = \) combined domain for the **first point** of the trace

\[ \phi = F([A] > s) \]

\[ D_{\phi} = \{ s < \max[A] \} \]

**Computational cost:** up to \( O(n^v) \)

\( v = \) number of variables

How to find a simplified trace that will keep the same validity domain?
Validation domain computing algorithm

Dedicated solvers:

• Specific function for a dynamical behavior
• Direct computing of the validity domain on the trace

Specification:

$$\phi = \exists (t_1, t_2) \mid p = t_2 - t_1$$
$$\wedge F\left(\frac{dA}{dt} > 0 \wedge X\left(\frac{dA}{dt} \leq 0 \wedge Time = t_1\right)\right)$$
$$\wedge (\frac{dA}{dt} \leq 0)U(\frac{dA}{dt} > 0)$$
$$\wedge ((\frac{dA}{dt} > 0)U(\frac{dA}{dt} \leq 0 \wedge Time = t_2)))$$

Result:

$$p = 23 \ | \ p=24.5$$

Computational cost:

$$O(n^2)$$

Trace simplification

For the case when there is no dedicated solver, how to make the generic algorithm more efficient?

Trace simplification: local extrema

Under which condition on the constraints is it safe to use this simplification?
Proof of validity: peak and period

**Peak**

**Formula:** $\phi = F\Big(\frac{dA}{dt} > 0 \land X\Big(\frac{dA}{dt} \leq 0 \land \text{Time} = t\Big)\Big)$

**Validity Domain:**

$D_{T,\phi} = D_{s_0,\phi}^{T}$

$\forall e = t \rightarrow$

$$\bigcup_{i=0}^{n} \left( D_{s_i,\frac{dA}{dt} > 0} \cap \bigcap_{i=0}^{n} D_{s_{i+1},\frac{dA}{dt} \leq 0} \cap D_{s_{i+e},\text{Time}} \right)$$

**Trace simplification:**

The optimal trace simplification is $T_J$ with $J = \{ i, i + 1 \in \{0, \ldots, n\} \mid \frac{dA}{dt}_{s_i} > 0 \land \frac{dA}{dt}_{s_{i+1}} \leq 0 \}$

$T^e_A$ is a simplification of $T$ for $\phi$.

**Period**

**Formula:** $\phi = \exists(t1, t2) \mid p = t2 - t1 \land t1 < t2$

$$\land F\Big(\frac{dA}{dt} > 0 \land X\Big(\frac{dA}{dt} \leq 0 \land \text{Time} = t1\Big)\Big)$$

$$\land F\Big(\frac{dA}{dt} > 0 \land X\Big(\frac{dA}{dt} \leq 0 \land \text{Time} = t2\Big)\Big)$$

$$\land \neg \exists t3 \mid t1 < t3 < t2 \land F\Big(\frac{dA}{dt} > 0 \land X\Big(\frac{dA}{dt} \leq 0 \land \text{Time} = t3\Big)\Big)$$
Proof of validity: peak and period

Peak

Formula: \( \phi = F \left( \frac{dA}{dt} > 0 \land X \left( \frac{dA}{dt} \leq 0 \land \text{Time} = t \right) \right) \)

Validity Domain:
\[ \mathcal{D}_T, \phi = \mathcal{D}_{s_0, \phi} \]

Trace simplification:
The optimal trace simplification is \( T_J \) with \( J = \{ i, i + 1 \in \{0, \ldots, n\} \mid \frac{dA}{dt}_{s_i} > 0 \land \frac{dA}{dt}_{s_{i+1}} \leq 0 \} \)

\( T^e_A \) is a simplification of \( T \) for \( \phi \).

Period

Formula: \( \phi = \exists (t1, t2) \mid p = t_2 - t_1 \land t_1 < t_2 \)

\[ \land F \left( \frac{dA}{dt} > 0 \land X \left( \frac{dA}{dt} \leq 0 \land \text{Time} = t1 \right) \right) \]

\[ \land F \left( \frac{dA}{dt} > 0 \land X \left( \frac{dA}{dt} \leq 0 \land \text{Time} = t2 \right) \right) \]

\[ \land \neg \exists t_3 \mid t_1 < t_3 < t_2 \land F \left( \frac{dA}{dt} > 0 \land X \left( \frac{dA}{dt} \leq 0 \land \text{Time} = t3 \right) \right) \]
Proof of validity: phase and amplitude

**Peak**

**Formula:**

\[ \phi = \exists(t_1, t_2) \mid p = t_2 - t_1 \land t_1 < t_2 \]

\[ \land F\left( \frac{dA}{dt} > 0 \land X\left( \frac{dA}{dt} \leq 0 \land \text{Time} = t_1 \right) \right) \]

\[ \land F\left( \frac{dB}{dt} > 0 \land X\left( \frac{dB}{dt} \leq 0 \land \text{Time} = t_2 \right) \right) \]

**Trace simplification:**

The optimal trace simplification is \( T_j \) with \( J = \{i, i+1 \in \{0, \ldots, n\}\} \)

\[ \frac{dA}{dt} s_i > 0 \land \frac{dA}{dt} s_{i+1} \leq 0 \]

\( T_{e_{A,B}} \) is a simplification of \( T \) for \( \phi \).

**Minimal amplitude**

**Formula:** \[ \phi = \exists v \mid F(A < v) \land F(A > v + a) \]

**Validity Domain:**

\[ \mathcal{D}_{T, \phi} = \Pi_a(\mathcal{D}_{s_0,0}^T, F(A < v) \cap \mathcal{D}_{s_0,0}^T, F(A > v + a)) \]

\[ = \Pi_a((\bigcup_{i=0}^{n} \mathcal{D}_{s_i, A < v}^T) \cap (\bigcup_{i=0}^{n} \mathcal{D}_{s_i, A > v + a}^T)) \]

\[ = \Pi_a(\mathcal{D}_{s_{\text{minA}}, A < v}^T \cap \mathcal{D}_{s_{\text{maxA}}, A > v + a}^T) \]

**Trace simplification:**

The optimal trace simplification is \( T_j \) where \( J = \{\text{minA, maxA}\} \).

\( T_{e_A} \) is a simplification of \( T \) for \( \phi \).
**First theorem:** If a simplification trace is correct for $\phi$ and $\psi$ then it is correct for the logical combinations of $\phi$ and $\psi$.

**Proof:**

\[
\mathcal{D}_{s_i,\phi \land \psi}^T = \mathcal{D}_{s_i,\phi}^T \cap \mathcal{D}_{s_i,\psi}^T = \mathcal{D}_{s_j,\phi}^T \cap \mathcal{D}_{s_j,\psi}^T = \mathcal{D}_{s_j,\phi \land \psi}^T
\]

**Second theorem:**

If a subtrace contains extreme domains, it is a simplification for $F$.

**Proof:**

\[
D_{\phi}^T = \bigcup_i D_{s_i,\phi} \cap \bigcup_j D_{s_j,\phi}
\]

Similar result for $G$: A simplification trace of $G\phi$ is the set of points $s_j$ whose $D_{s_j,\phi}$ is contained in all the $D_{s_i,\phi}$

**Corollary:** A simplified trace on $T$ for $F(c \land \phi)$ can be found by discarding all the points where $c$ is false, if this defines a simplified trace on $T$ for $\phi$. 
**First theorem:*** If a simplification trace is correct for \( \phi \) and \( \psi \) then it is correct for the logical combinations of \( \phi \) and \( \psi \).

**Proof:**

\[
D^T_{s_i, \phi \land \psi} = D^T_{s_i, \phi} \cap D^T_{s_i, \psi} = D^T_{s_j, \phi} \cap D^T_{s_j, \psi} = D^T_{s_j, \phi \land \psi}
\]

**Minimal Amplitude**

**Formula:**

\[
\phi = \exists v \mid F(A < v) \land F(A > v + a)
\]

**Validity Domain:**

\[
D_T, \phi = \prod_a (D^T_{s_0, F(A < v)} \cap D^T_{s_0, F(A > v + a)})
\]

\[
= \prod_a ((\bigcup_{i=0}^n D^T_{s_i, A < v}) \cap (\bigcup_{i=0}^n D^T_{s_j, A > v + a}))
\]

\[
= \prod_a (D^T_{s_{\min A}, A < v} \cap D^T_{s_{\max A}, A > v + a})
\]

**Trace simplification:**

The optimal trace simplification is \( T_J \) where \( J = \{\min A, \max A\} \).

\( T^e_A \) is a simplification of \( T \) for \( \phi \).
Second theorem:
If a subtrace contains extreme domains, it is a simplification for $F$.

Proof: $D^T \phi = \bigcup_i D_{s_j,\phi} \cap \bigcup_j D_{s_j,\phi}$

Similar result for $G$: A simplification trace of $G\phi$ is the set of points $s_j$ whose $D_{s_j,\phi}$ is contained in all the $D_{s_i,\phi}$.
Second theorem:
If a subtrace contains extreme domains, it is a simplification for $F$.

Proof: $D^T_\phi = U_i D_{s_j,\phi} \cap U_j D_{s_j,\phi}$

Similar result for $G$: A simplification trace of $G\phi$ is the set of points $s_j$ whose $D_{s_j,\phi}$ is contained in all the $D_{s_i,\phi}$.

$\phi = F( [A] > s )$

$D_\phi = \{ s < \max[A] \}$
**Corollary**

**Corollary:** A simplified trace on $T$ for $F(c \land \phi)$ can be found by discarding all the points where $c$ is false, if this defines a simplified trace on $T$ for $\phi$.

**Formula:** $\phi = F(Time > 20 \land A < v)$

**Validity Domain:** $D_{T,\phi} = D_{T,s0,F}(Time > 20 \land A < v)$

\[
= \bigcup_{i=0}^{n} D_{s_i, Time > 20 \land A < v}^{T}
\]

\[
= \bigcup_{i=0}^{n} (D_{s_i, Time > 20}^{T} \cap D_{s_i, A < v}^{T})
\]

\[
= \bigcup \{D_{s_i, A < v}^{T} \mid Time_{s_i} > 20\}
\]

\[
= D_{s_{minA > 20}, A < v}^{T}
\]

**Trace simplification:**

The single point $s_{minA > 20}$ defines an optimal trace simplification of $T$ for $\phi$. $T^e_A$ is not a simplification of $T$ for $\phi$ unless it does contain a local minimum such that $Time > 20$. 

**Diagram:**

A versus Time graph with a threshold at Time = 20 and points below the threshold being discarded.
**Corollary:** A simplified trace on $T$ for $F(c \land \phi)$ can be found by discarding all the points where $c$ is false, if this defines a simplified trace on $T$ for $\phi$.

**Formula:** $\phi = F(Time > 20 \land A < v)$  
**Validity Domain:** $D_{T,\phi} = D_{s_0,F(Time > 20 \land A < v)}$

$$= \bigcup_{i=0}^{n} D_{s_i,Time > 20 \land A < v}$$

$$= \bigcup_{i=0}^{n} (D_{s_i,Time > 20} \cap D_{s_i,A < v})$$

$$= \bigcup \{ D_{s_i,A < v} \mid Time_{s_i} > 20 \}$$

$$= D_{s_{minA > 20},A < v}$$

**Trace simplification:**  
The single point $s_{minA > 20}$ defines an optimal trace simplification of $T$ for $\phi$. $T_{e_A}$ *is not a simplification of* $T$ *for* $\phi$ *unless* it *does contain a local minimum such that* $Time > 20$. 

![Diagram](image-url)
**Corollary:** A simplified trace on T for $F(c \land \phi)$ can be found by discarding all the points where c is false, if this defines a simplified trace on T for $\phi$.

**Formula:** $\phi = F(Time > 20 \land A < v)$

**Validity Domain:** $D_{T,\phi} = D_{s_0,F(Time > 20 \land A < v)}$

\[
= \bigcup_{i=0}^{n} D_{s_i,Time > 20 \land A < v}
= \bigcup_{i=0}^{n} (D_{s_i,Time > 20} \cap D_{s_i,A < v})
= \bigcup_{\{i | Time_{s_i} > 20\}} D_{s_i,A < v}
= D_{s_{\text{minA > 20}},A < v}
\]

**Threshold**

**Trace simplification:**
The single point $s_{\text{minA > 20}}$ defines an optimal trace simplification of T for $\phi$.

$T^e_A$ is not a simplification of T for $\phi$ unless it does contain a local minimum such that $\text{Time} > 20$. 
Corollary: A simplified trace on $T$ for $F(c \land \phi)$ can be found by discarding all the points where $c$ is false, if this defines a simplified trace on $T$ for $\phi$.

**Formula:** $\phi = F(Time > 20 \land A < v)$

**Validity Domain:** $D_{T,\phi} = D^T_{s_0, F(Time > 20 \land A < v)}$

\[
= \bigcup_{i=0}^{n} D^T_{s_i, Time > 20 \land A < v}
\]

\[
= \bigcup_{i=0}^{n} (D^T_{s_i, Time > 20} \cap D^T_{s_i, A < v})
\]

\[
= \bigcup \{D^T_{s_i, A < v} \mid Time_{s_i} > 20\}
\]

\[
= D^T_{s_{\min A > 20}, A < v}
\]

**Threshold**

$T^e_A$ is *not a simplification of $T$ for $\phi$* unless it contains a local minimum such that $Time > 20$.
Crossing

**Formula:** \( \phi = F(A > B \land X(A \leq B \land Time = t)) \)

**Validity Domain:**

\[
\mathcal{D}_T^e = \bigcap_{i=0}^{n} \mathcal{D}_T^e A_i > B_i \land \mathcal{D}_T^e A_{i+1} \leq B_{i+1}
\]

\[
= \bigcup \{ \text{Time}_{s+1} \mid \exists i \in \{0, \ldots, n\} \text{ s.t. } A_{s_i} > B_{s_i} \land A_{s_i+1} \leq B_{s_i+1} \}
\]

Here \( T^e_{A,B} \) is NOT a simplification of \( T \) for \( \phi \).

A simplification trace is defined by the points in:

\[
J = \{ i, i + 1 \in \{0, \ldots, n\} \mid A_{s_i} > B_{s_i} \land A_{s_i+1} \leq B_{s_i+1} \}
\]
Crossing

**Formula:** \( \phi = F(A > B \land X(A \leq B \land Time = t)) \)

**Validity Domain:**

\[
\mathcal{D}_T^\phi = \bigcap_{n} (\mathcal{D}^T_{A > B} \cap (\mathcal{D}^T_{A < B} \cap \mathcal{D}^T_{Time = t}))
\]

\[
= \bigcup_{C \subseteq \{1, \ldots, n\}} \{ \text{Time}_{s+1} \}
\end{equation}

Here \( T^e_{A,B} \) is NOT a simplification of \( T \) for \( \phi \).

A simplification trace is defined by the points in:

\[
J = \{ i, i + 1 \in \{0, \ldots, n\} \mid A_{s_i} > B_{s_i} \land A_{s_{i+1}} \leq B_{s_{i+1}} \}
\]
Evaluation on Oscillation Constraints between the Cell Cycle and Circadian Clock

- The cell cycle and the circadian clock: two coupled oscillators involving:
  - qualitative properties: oscillations, stability
  - quantitative properties: period of each oscillator, phase

- Constraints on one molecule:
  - Minimum amplitude
  - Distance between successive peaks
  - Regularity of the distances between peaks
  - Regularity of the peak amplitudes

- Constraints on two molecules:
  - Phase

Cell cycle: MPF, Wee1
Circadian clock: Bmal1, PerCry, Rev-erba
Evaluation on Oscillation Constraints between the Cell Cycle and Circadian Clock

Trace simplification:
- **Extrema subtrace** implemented in BIOCHAM
- Computing times:
  - Rosenbrock’s variable step-size simulation: 8-16 ms
  - 4th order Runge-Kutta fixed step-size simulation: 160-250 ms

<table>
<thead>
<tr>
<th>Formula</th>
<th>Nb of points Solver</th>
<th>First trace</th>
<th>Second trace</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>variable Bef.</td>
<td>variable Bef.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>971</td>
<td>18</td>
</tr>
<tr>
<td>Reachability of PerCry</td>
<td>generic</td>
<td>12</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>dedicated</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Minimum amplitude of PerCry</td>
<td>generic</td>
<td>132</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>dedicated</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Local maxima of PerCry</td>
<td>generic</td>
<td>64</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>dedicated</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Distance betw. PerCry peaks</td>
<td>generic</td>
<td>512</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>dedicated</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Distance betw. succ. PerCry peaks</td>
<td>generic</td>
<td>532</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>dedicated</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>Regularity of PerCry peaks</td>
<td>generic</td>
<td>1700</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>dedicated</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Phase betw. PerCry and MPF</td>
<td>generic</td>
<td>456</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>dedicated</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

• **Validity domain computing time** (in ms):
Conclusion

- Temporal logic patterns provide an elegant way to
  - extract meaningful information on the periods and phases from numerical traces
  - use these formulae as constraints for parameter search

- Simplifying the trace prior to the solving makes the generic solving algorithm more efficient

- Under some general conditions on the syntax of the formulae given as theorems 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 simulation traces, the speedup obtained in computation time was by several orders of magnitude: up to 1000 fold.

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