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

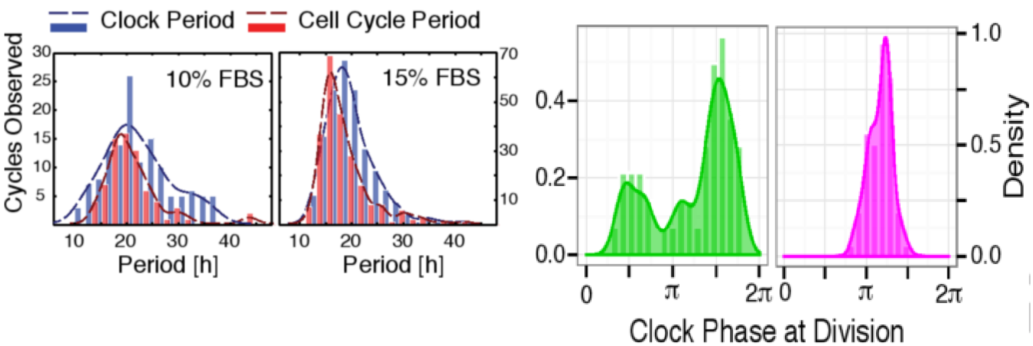
CMSB 2014

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?

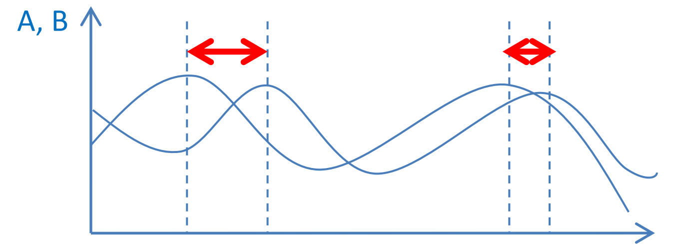
Model building assisted with formal methods (model calibration)

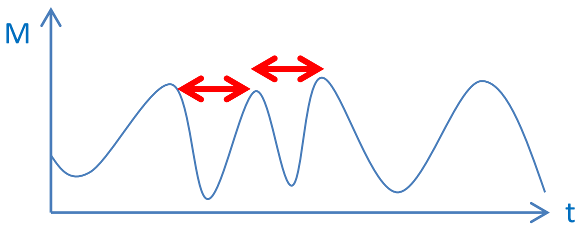
Predictions: mechanisms and perturbations, optimization

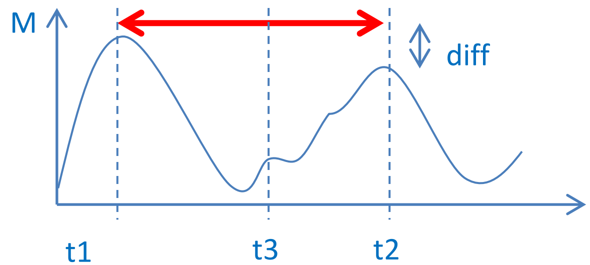
Bidirectional coupled model of the cell cycle and the circadian clock

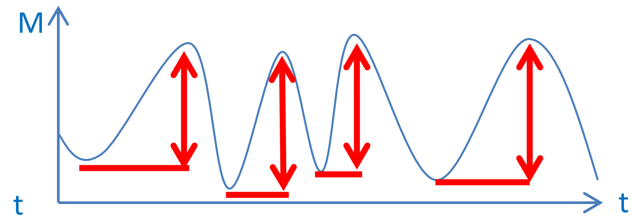
Temporal logic specifications

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









- **Formalised with temporal logic:**

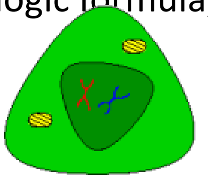
Ex: <u>period</u>

$$\phi = \exists(t1, t2) \mid p = t_2 - t_1 \wedge t_1 < t_2$$
$$\wedge\, \mathbf{F}\left(\frac{dA}{dt} > 0 \wedge \mathbf{X}\left(\frac{dA}{dt} \leq 0 \wedge Time = t1\right)\right)$$
$$\wedge\, \mathbf{F}\left(\frac{dA}{dt} > 0 \wedge \mathbf{X}\left(\frac{dA}{dt} \leq 0 \wedge Time = t2\right)\right)$$

<u>Result:</u>
-Possible values: p = 23 | p=24.5
- Satisfaction degree (with objective p=24): 0.1

- **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**

http://lifeware.inria.fr/Biocham/

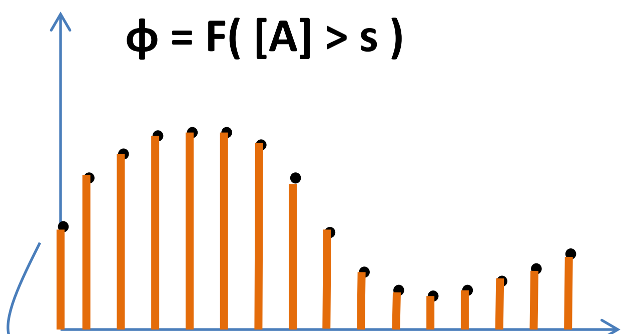


Validation domain computing algorithm

<u>Generic algorithm:</u>

- **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: $\mathcal{D}^T_{s_i,\mathbf{F}\phi} = \bigcup_{j=i}^{n} \mathcal{D}^T_{s_j,\phi}$
  - Operator **G** (globally) → intersection: $\mathcal{D}^T_{s_i,\mathbf{G}\phi} = \bigcap_{j=i}^{n} \mathcal{D}^T_{s_j,\phi}$
  - Operator **X** (next) → next domain if valid: $\mathcal{D}^T_{s_i,\mathbf{X}\phi} = \mathcal{D}^T_{s_{i+\mathbf{1}},\phi}$
  - Operator **U** (until) → union of intersections: $\mathcal{D}^T_{s_i,\phi\mathbf{U}\psi} = \bigcup_{j=i}^{n}(\mathcal{D}^T_{s_j,\psi} \cap \bigcap_{k=i}^{j-\mathbf{1}} \mathcal{D}^T_{s_k,\phi})$
- Domain for ϕ = combinated domain for the **first point** of the trace

ϕ = F([A] > s)



D_ϕ = { s < max[A] }

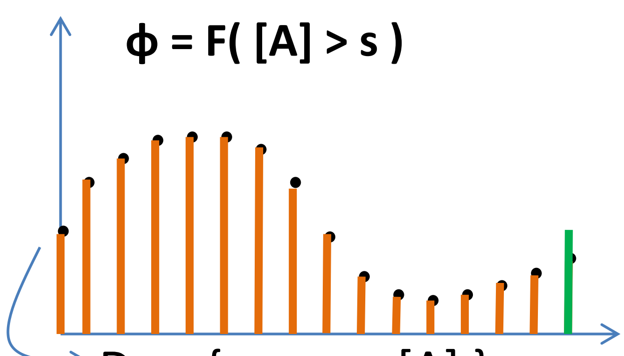
Computational cost: up to O(n^v)
(v = number of variables)

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

Validation domain computing algorithm

<u>Generic algorithm:</u>

- **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: $\mathcal{D}^{T}_{s_i,\mathbf{F}\phi} = \bigcup_{j=i}^{n} \mathcal{D}^{T}_{s_j,\phi}$
  - Operator **G** (globally) → intersection: $\mathcal{D}^{T}_{s_i,\mathbf{G}\phi} = \bigcap_{j=i}^{n} \mathcal{D}^{T}_{s_j,\phi}$
  - Operator **X** (next) → next domain if valid: $\mathcal{D}^{T}_{s_i,\mathbf{X}\phi} = \mathcal{D}^{T}_{s_{i+\mathbf{1}},\phi}$
  - Operator **U** (until) → union of intersections: $\mathcal{D}^{T}_{s_i,\phi\mathbf{U}\psi} = \bigcup_{j=i}^{n} (\mathcal{D}^{T}_{s_j,\psi} \cap \bigcap_{k=i}^{j-\mathbf{1}} \mathcal{D}^{T}_{s_k,\phi})$
- Domain for ϕ = combinated domain for the **first point** of the trace



D_{ϕ} = { s < max[A] }

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

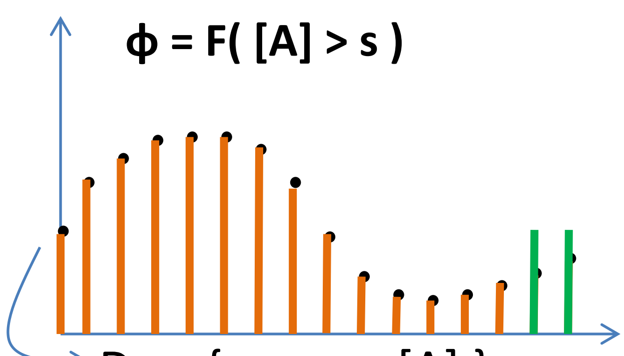
How to find a simplified trace that will **keep the same validity domain?**

Validation domain computing algorithm

Generic algorithm:

- **Decomposition** of 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: $\mathcal{D}^T_{s_i,\mathbf{F}\phi} = \bigcup_{j=i}^n \mathcal{D}^T_{s_j,\phi}$
  - Operator **G** (globally) → intersection: $\mathcal{D}^T_{s_i,\mathbf{G}\phi} = \bigcap_{j=i}^n \mathcal{D}^T_{s_j,\phi}$
  - Operator **X** (next) → next domain if valid: $\mathcal{D}^T_{s_i,\mathbf{X}\phi} = \mathcal{D}^T_{s_{i+1},\phi}$
  - Operator **U** (until) → union of intersections: $\mathcal{D}^T_{s_i,\phi\mathbf{U}\psi} = \bigcup_{j=i}^n (\mathcal{D}^T_{s_j,\psi} \cap \bigcap_{k=i}^{j-1} \mathcal{D}^T_{s_k,\phi})$
- Domain for ϕ = combinated domain for the **first point** of the trace

ϕ = F([A] > s)



D_ϕ = { s < max[A] }

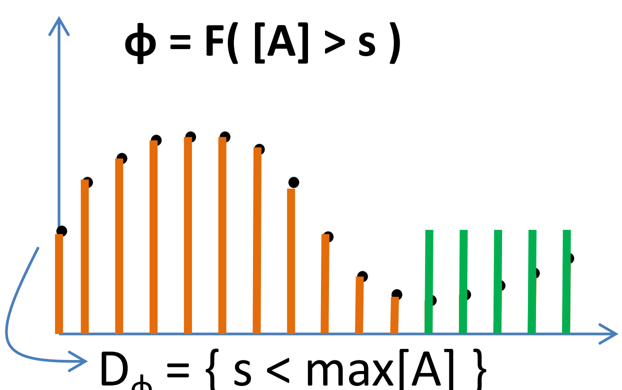
Computational cost: up to O(n^v)
(v = number of variables)

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

Validation domain computing algorithm

Generic algorithm:

- **Decomposition** of 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: $\mathcal{D}^{T}_{s_i,\mathbf{F}\phi} = \bigcup_{j=i}^{n} \mathcal{D}^{T}_{s_j,\phi}$
  - Operator **G** (globally) → intersection: $\mathcal{D}^{T}_{s_i,\mathbf{G}\phi} = \bigcap_{j=i}^{n} \mathcal{D}^{T}_{s_j,\phi}$
  - Operator **X** (next) → next domain if valid: $\mathcal{D}^{T}_{s_i,\mathbf{X}\phi} = \mathcal{D}^{T}_{s_{i+\mathbf{1}},\phi}$
  - Operator **U** (until) → union of intersections: $\mathcal{D}^{T}_{s_i,\phi\mathbf{U}\psi} = \bigcup_{j=i}^{n}(\mathcal{D}^{T}_{s_j,\psi} \cap \bigcap_{k=i}^{j-\mathbf{1}} \mathcal{D}^{T}_{s_k,\phi})$
- Domain for ϕ = combinated domain for the **first point** of the trace



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

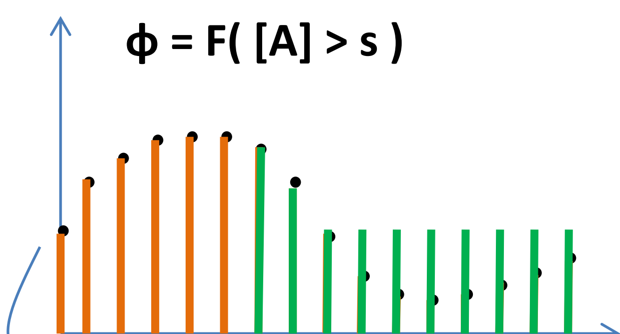
How to find a simplified trace that will **keep the same validity domain?**

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: $\mathcal{D}^T_{s_i,\mathbf{F}\phi} = \bigcup_{j=i}^n \mathcal{D}^T_{s_j,\phi}$
  - Operator **G** (globally) → intersection: $\mathcal{D}^T_{s_i,\mathbf{G}\phi} = \bigcap_{j=i}^n \mathcal{D}^T_{s_j,\phi}$
  - Operator **X** (next) → next domain if valid: $\mathcal{D}^T_{s_i,\mathbf{X}\phi} = \mathcal{D}^T_{s_{i+\mathbf{1}},\phi}$
  - Operator **U** (until) → union of intersections: $\mathcal{D}^T_{s_i,\phi\mathbf{U}\psi} = \bigcup_{j=i}^n(\mathcal{D}^T_{s_j,\psi} \cap \bigcap_{k=i}^{j-\mathbf{1}} \mathcal{D}^T_{s_k,\phi})$
- Domain for ϕ = combinated domain for the **first point** of the trace

ϕ = F([A] > s)



D_ϕ = { s < max[A] }

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

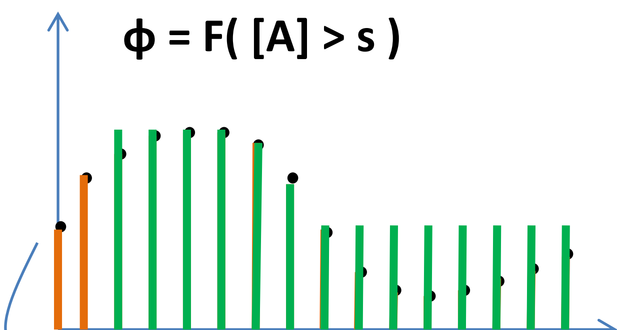
How to find a simplified trace that will **keep the same validity domain?**

Validation domain computing algorithm

Generic algorithm:

- **Decomposition** of 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: $\mathcal{D}^T_{s_i,\mathbf{F}\phi} = \bigcup_{j=i}^{n} \mathcal{D}^T_{s_j,\phi}$
  - Operator **G** (globally) → intersection: $\mathcal{D}^T_{s_i,\mathbf{G}\phi} = \bigcap_{j=i}^{n} \mathcal{D}^T_{s_j,\phi}$
  - Operator **X** (next) → next domain if valid: $\mathcal{D}^T_{s_i,\mathbf{X}\phi} = \mathcal{D}^T_{s_{i+\mathbf{1}},\phi}$
  - Operator **U** (until) → union of intersections: $\mathcal{D}^T_{s_i,\phi\mathbf{U}\psi} = \bigcup_{j=i}^{n}(\mathcal{D}^T_{s_j,\psi} \cap \bigcap_{k=i}^{j-\mathbf{1}} \mathcal{D}^T_{s_k,\phi})$
- Domain for ϕ = combinated domain for the **first point** of the trace

ϕ = F([A] > s)



D_ϕ = { s < max[A] }

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

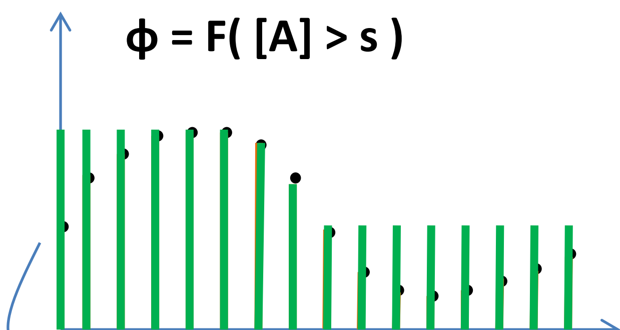
➡ How to find a simplified trace that will **keep the same validity domain?**

Validation domain computing algorithm

Generic algorithm:

- **Decomposition** of 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: $\mathcal{D}^{T}_{s_i,\mathbf{F}\phi} = \bigcup_{j=i}^{n} \mathcal{D}^{T}_{s_j,\phi}$
  - Operator **G** (globally) → intersection: $\mathcal{D}^{T}_{s_i,\mathbf{G}\phi} = \bigcap_{j=i}^{n} \mathcal{D}^{T}_{s_j,\phi}$
  - Operator **X** (next) → next domain if valid: $\mathcal{D}^{T}_{s_i,\mathbf{X}\phi} = \mathcal{D}^{T}_{s_{i+\mathbf{1}},\phi}$
  - Operator **U** (until) → union of intersections: $\mathcal{D}^{T}_{s_i,\phi\mathbf{U}\psi} = \bigcup_{j=i}^{n}(\mathcal{D}^{T}_{s_j,\psi} \cap \bigcap_{k=i}^{j-\mathbf{1}} \mathcal{D}^{T}_{s_k,\phi})$
- Domain for ϕ = combinated domain for the **first point** of the trace

ϕ = F([A] > s)



D_ϕ = { s < max[A] }

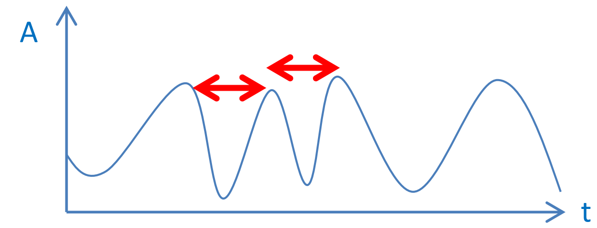
Computational cost: up to $O(n^v)$
(v = number of variables)

⇒ 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:

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

→ distanceSuccPeaks(A,B,dist)

Result: p = 23 | p=24.5 → p = 23 | p=24.5

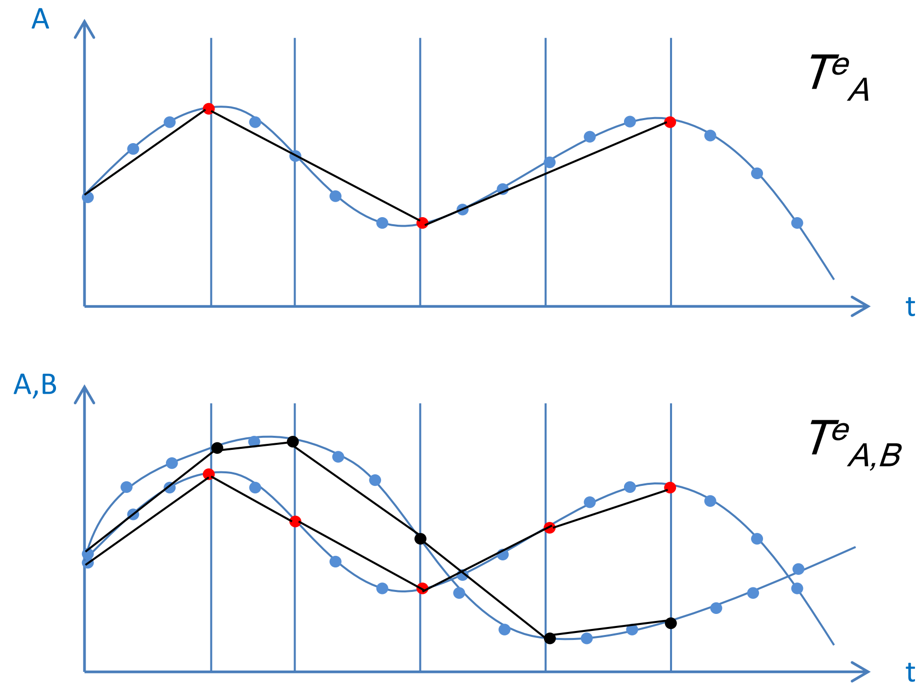
Computational cost: $O(n^2)$ → O(n)

F. Fages, P. Traynard. Temporal Logic Modeling of Dynamical Behaviors: First-Order Patterns and Solvers. In *Logical Modeling of Biological Systems*, pages 307–338. ISTE Ltd, Eds. L. Farinas del Cerro et K. Inoue., 2014

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 ?

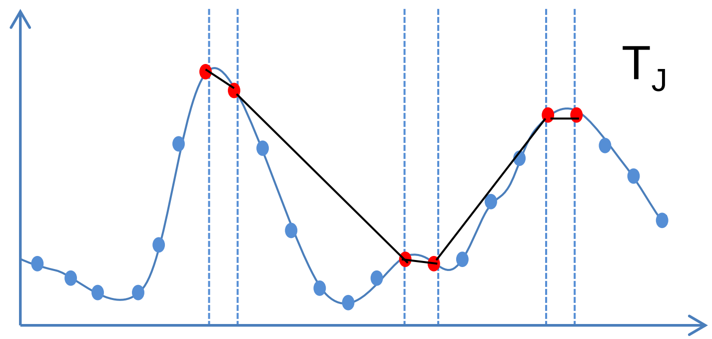
Proof of validity: peak and period

Peak

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

Validity Domain:

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



Trace simplification:

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

$T^e{}_A$ is a simplification of T for φ.

Period

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

$$\wedge\, \mathbf{F}(\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge Time = t1))$$

$$\wedge\, \mathbf{F}(\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge Time = t2))$$

$$\wedge\, \neg\exists t_3 \mid t_1 < t_3 < t_2 \wedge \mathbf{F}(\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge Time = t3))$$

Same trace simplification

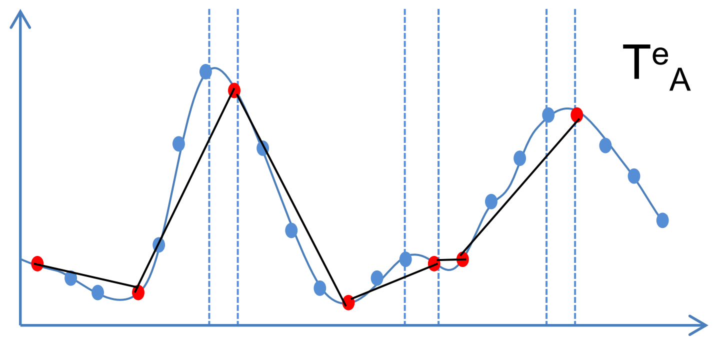
Proof of validity: peak and period

Peak

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

Validity Domain:

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



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

T^e_A is a simplification of T for φ.

Period

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

$$\wedge \mathbf{F}(\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge Time = t1))$$

$$\wedge \mathbf{F}(\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge Time = t2))$$

$$\wedge \neg\exists t_3 \mid t_1 < t_3 < t_2 \wedge \mathbf{F}(\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge Time = t3))$$

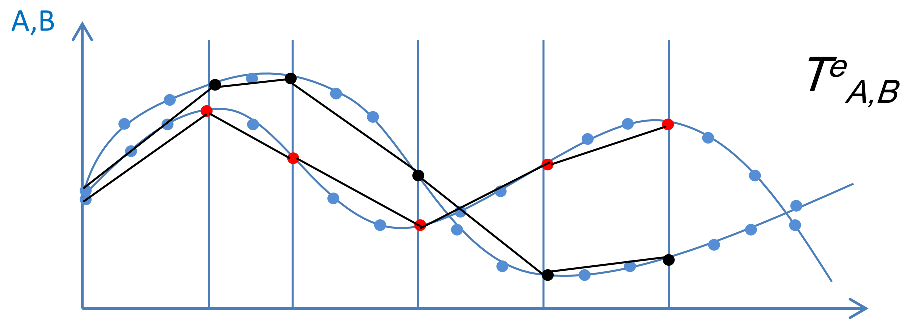
Same trace simplification

Proof of validity: phase and amplitude

Peak

Formula:

$$\phi = \exists(t1, t2) \mid p = t_2 - t_1 \wedge t_1 < t_2$$
$$\wedge \mathbf{F}(\frac{dA}{dt} > 0 \wedge \mathbf{X}(\frac{dA}{dt} \leq 0 \wedge Time = t1))$$
$$\wedge \mathbf{F}(\frac{dB}{dt} > 0 \wedge \mathbf{X}(\frac{dB}{dt} \leq 0 \wedge Time = t2))$$



Trace simplification:

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

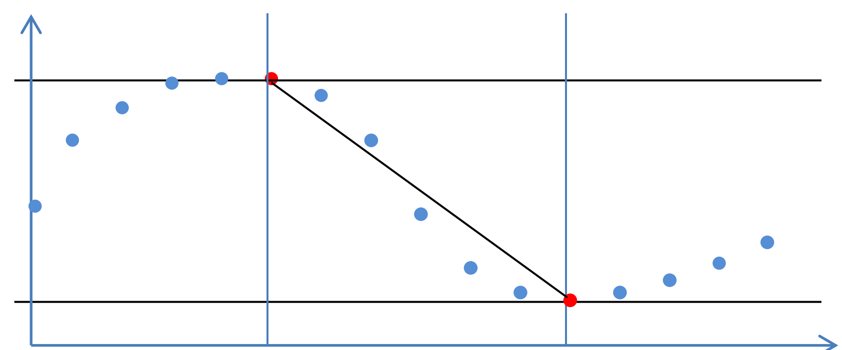
$T^e_{A,B}$ is a simplification of T for φ.

Minimal amplitude

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

Validity Domain:

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



Trace simplification:

The optimal trace simplification is T_J where J = {minA, maxA}.

T^e_A is a simplification of T for φ.

General Theorems

First theorem: If a simplification trace is correct for ϕ and ψ then it is correct for the logical combinations of ϕ and ψ.

Proof: $\mathcal{D}^{T}_{s_i,\phi\wedge\psi} = \mathcal{D}^{T}_{s_i,\phi} \cap \mathcal{D}^{T}_{s_i,\psi} = \mathcal{D}^{T'}_{s_{j_i},\phi} \cap \mathcal{D}^{T'}_{s_{j_i},\psi} = \mathcal{D}^{T'}_{s_{j_i},\phi\wedge\psi}$

Second theorem:
If a subtrace contains extreme domains, it is a simplification for **F**.

Proof: $D^{T}_{\phi} = U_i\, D_{sj,\phi}\ \ C\ \ U_j\, D_{sj,\phi}$

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

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

First theorem

First theorem: If a simplification trace is correct for ϕ and ψ then it is correct for the logical combinations of ϕ and ψ.

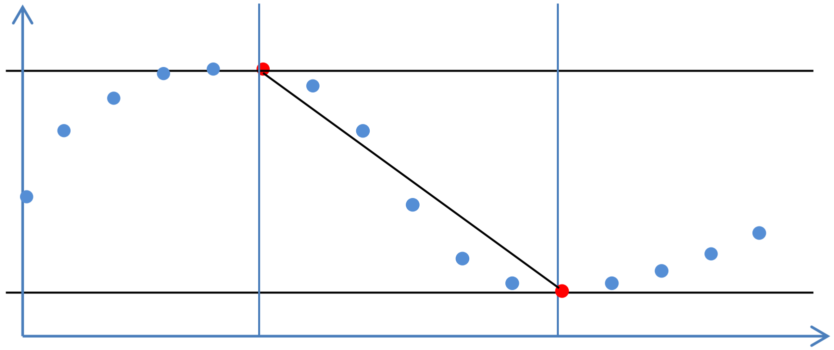
Proof: $\mathcal{D}^{T}_{s_i,\phi\wedge\psi} = \mathcal{D}^{T}_{s_i,\phi} \cap \mathcal{D}^{T}_{s_i,\psi} = \mathcal{D}^{T'}_{s_{j_i},\phi} \cap \mathcal{D}^{T'}_{s_{j_i},\psi} = \mathcal{D}^{T'}_{s_{j_i},\phi\wedge\psi}$

Minimal Amplitude

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

Validity Domain:

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



Trace simplification:

The optimal trace simplification is T_J where J = {minA, maxA}.

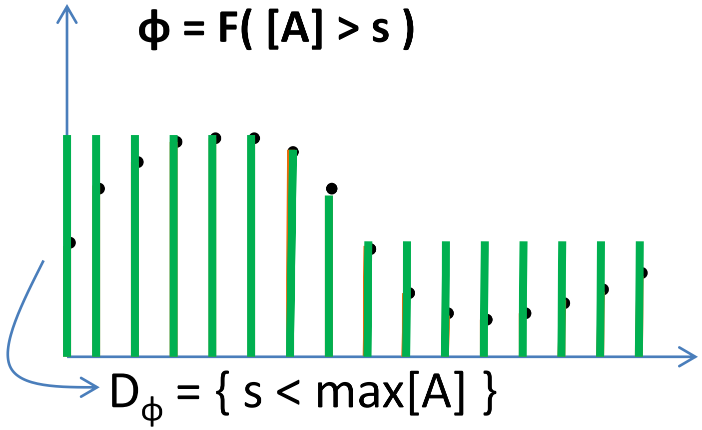
T^e_A is a simplification of T for φ.

Second theorem

Second theorem:
If a subtrace contains extreme domains, it is a simplification for **F**.

Proof: $D^T_\phi = \cup_i D_{sj,\phi} \subset \cup_j D_{sj,\phi}$

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

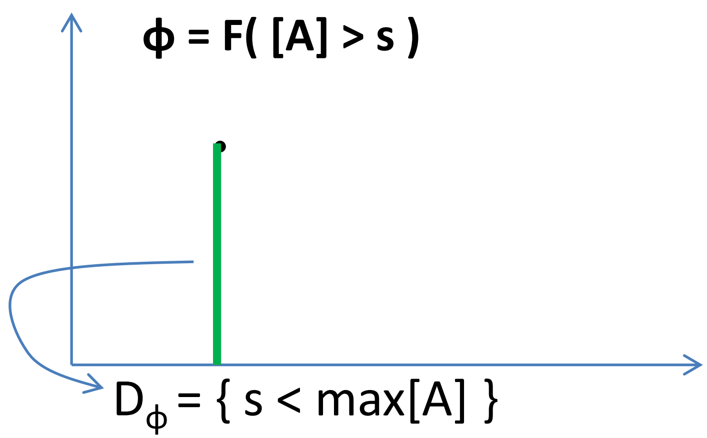


Second theorem

Second theorem:
If a subtrace contains extreme domains, it is a simplification for **F**.

Proof: $D^T_\phi = U_i\, D_{sj,\phi}\ \ C\ \ U_j\, D_{sj,\phi}$

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



Corolary

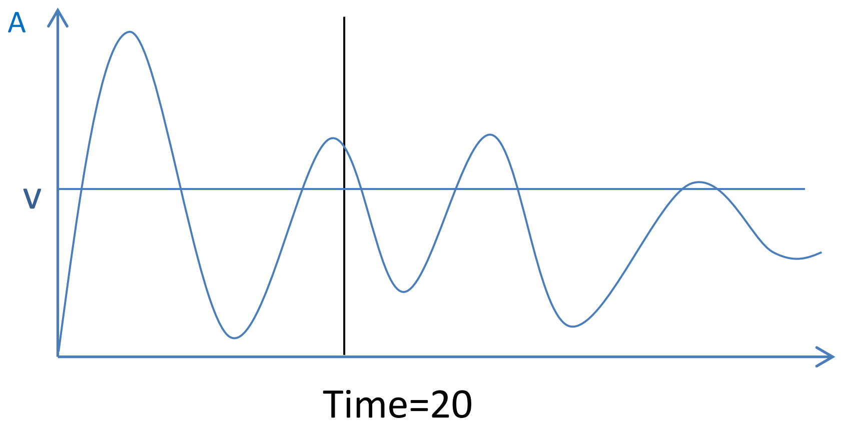
Corollary: A simplified trace on T for $\mathbf{F}(c \wedge \phi)$ can be found by discarding all the points where c is false, if this defines a simplified trace on T for ϕ.

Threshold

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

Validity Domain:

$$\mathcal{D}_{T,\phi} = \mathcal{D}^T_{s_0,\mathbf{F}(Time>20 \wedge A<v)}$$
$$= \bigcup_{i=0}^{n} \mathcal{D}^T_{s_i,\, Time>20 \wedge A<v}$$
$$= \bigcup_{i=0}^{n} (\mathcal{D}^T_{s_i,\, Time>20} \cap \mathcal{D}^T_{s_i,A<v})$$
$$= \bigcup_{\{i \mid Time_{s_i}>20\}} \mathcal{D}^T_{s_i,A<v}$$
$$= \mathcal{D}^T_{s_{minA>20},A<v}$$



Trace simplification:

The single point $s_{minA>20}$ defines an optimal trace simplification of T for ϕ.

T^e_A is not a simplification of T for φ unless it does contain a local minimum such that Time>20.

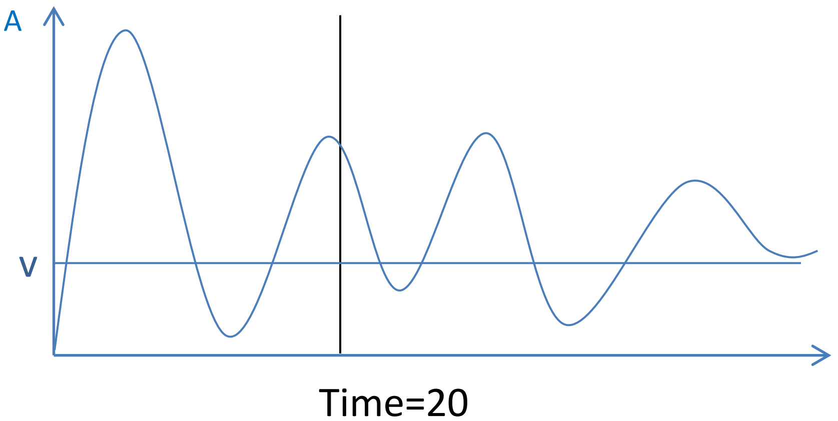
Corolary

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

Threshold

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

Validity Domain:
$$\mathcal{D}_{T,\phi} = \mathcal{D}^T_{s_0, \mathbf{F}(Time>20 \wedge A<v)}$$
$$= \bigcup_{i=0}^{n} \mathcal{D}^T_{s_i, Time>20 \wedge A<v}$$
$$= \bigcup_{i=0}^{n} (\mathcal{D}^T_{s_i, Time>20} \cap \mathcal{D}^T_{s_i, A<v})$$
$$= \bigcup_{\{i \mid Time_{s_i}>20\}} \mathcal{D}^T_{s_i, A<v}$$
$$= \mathcal{D}^T_{s_{minA>20}, A<v}$$



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

Corolary

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

Threshold

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

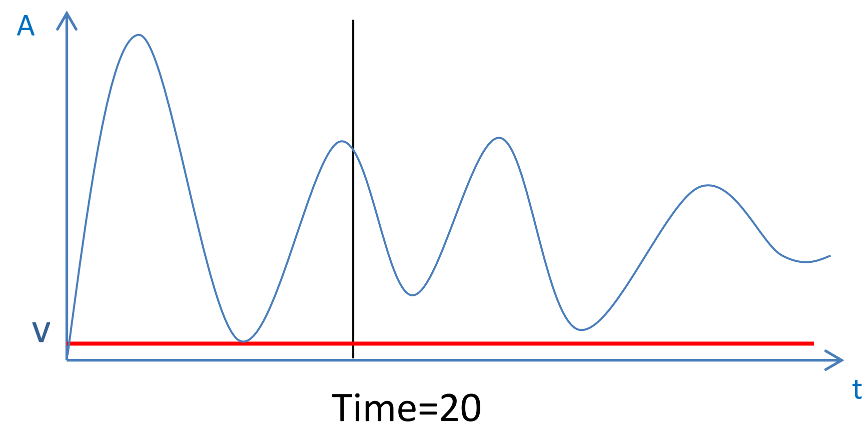
Validity Domain: $\mathcal{D}_{T,\phi} = \mathcal{D}^T_{s_0, \mathbf{F}(Time>20 \wedge A<v)}$

$$= \bigcup_{i=0}^{n} \mathcal{D}^T_{s_i, Time>20 \wedge A<v}$$

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

$$= \bigcup_{\{i | Time_{s_i} > 20\}} \mathcal{D}^T_{s_i, A<v}$$

$$= \mathcal{D}^T_{s_{minA>20}, A<v}$$



Trace simplification:

The single point $s_{minA>20}$ defines an optimal trace simplification of T for ϕ.

T^e_A is not a simplification of T for φ unless it does contain a local minimum such that Time>20.

Corolary

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

Threshold

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

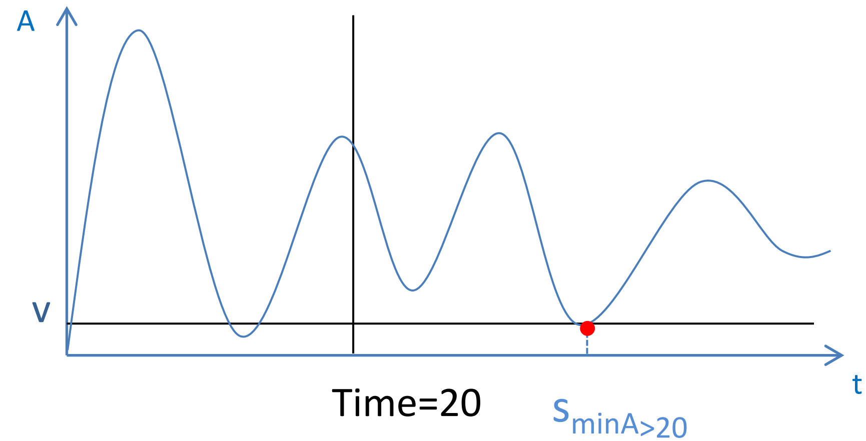
Validity Domain: $\mathcal{D}_{T,\phi} = \mathcal{D}^{T}_{s_0, \mathbf{F}(Time > 20 \wedge A < v)}$

$$= \bigcup_{i=0}^{n} \mathcal{D}^{T}_{s_i, Time > 20 \wedge A < v}$$

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

$$= \bigcup_{\{i \mid Time_{s_i} > 20\}} \mathcal{D}^{T}_{s_i, A < v}$$

$$= \mathcal{D}^{T}_{s_{minA>20}, A < v}$$



Trace simplification:

The single point $s_{minA>20}$ defines an optimal trace simplification of T for ϕ.

T^e_A is not a simplification of T for φ unless it does contain a local minimum such that Time>20.

Crossing

Crossing

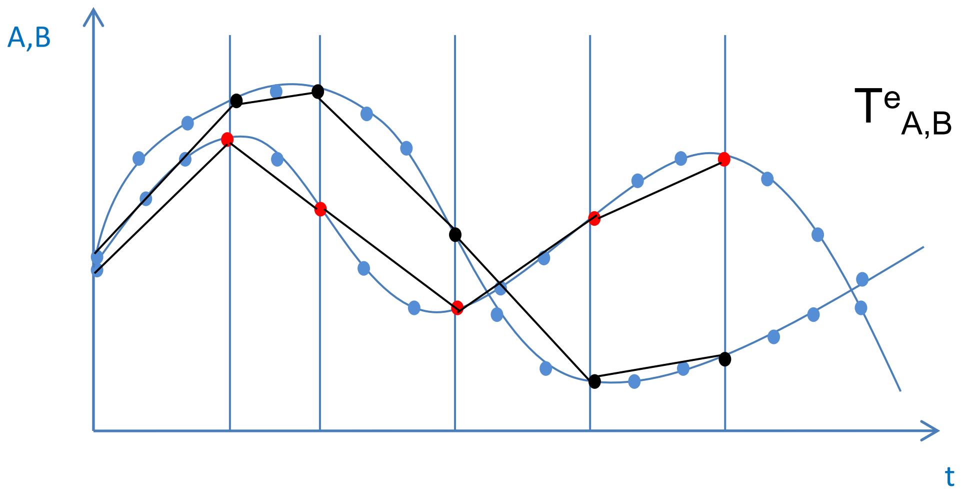
Formula: $\phi = \mathbf{F}(A > B \wedge \mathbf{X}(A \leq B \wedge Time = t))$

Validity Domain:

$$\mathcal{D}_{T,\phi} = \bigcup_{i=0}^{n} (\mathcal{D}^{T}_{s_i, A_{s_i} > B_{s_i}} \cap (\mathcal{D}^{T}_{s_{i+1}, A_{s_i} \leq B_{s_i}} \cap \mathcal{D}^{T}_{s_{i+1}, Time=t}))$$

$$= \bigcup_{i \in \{0,\ldots,n\}] | A_{s_i} > B_{s_i} \wedge A_{s_{i+1}} \leq B_{s_{i+1}}} \{Time_{s_{i+1}}\}$$

Here $T^{e}_{A,B}$ is NOT a simplification of T for φ.



A simplification trace is defined by the points in:

$$J = \{i, i+1 \in \{0,\ldots,n\}] \mid A_{s_i} > B_{s_i} \wedge A_{s_{i+1}} \leq B_{s_{i+1}}\}$$

Crossing

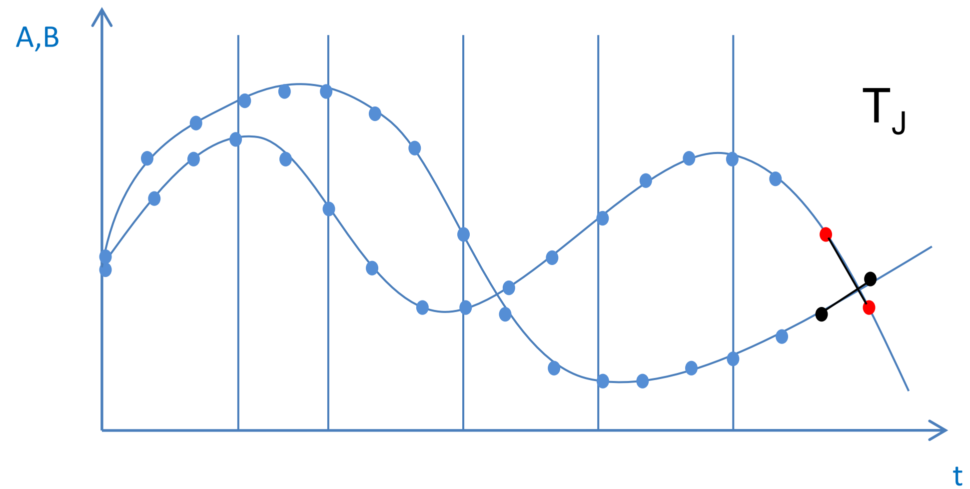
Crossing

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

$$\mathcal{D}_{T,\phi} = \bigcup_{i=0}^{n} (\mathcal{D}^{T}_{s_i, A_{s_i} > B_{s_i}} \cap (\mathcal{D}^{T}_{s_{i+1}, A_{s_i} \leq B_{s_i}} \cap \mathcal{D}^{T}_{s_{i+1}, Time=t}))$$

$$= \bigcup_{i \in \{0,\ldots,n\}] | A_{s_i} > B_{s_i} \wedge A_{s_{i+1}} \leq B_{s_{i+1}}} \{Time_{s_{i+1}}\}$$

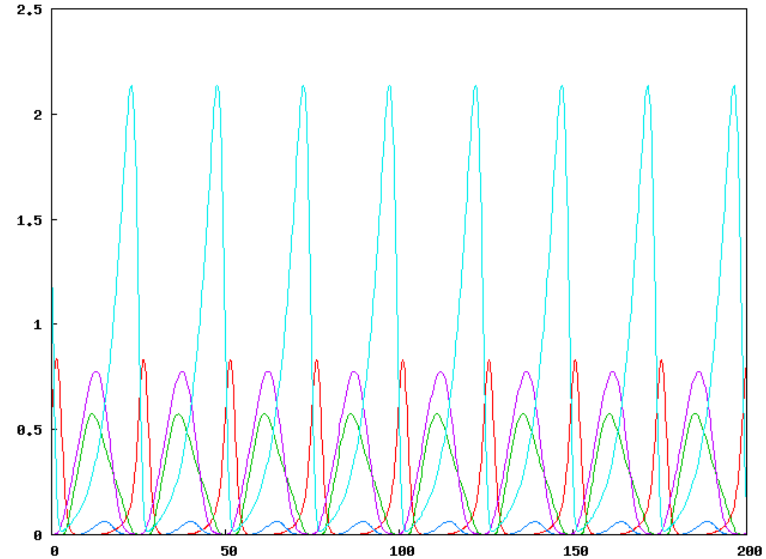
Here $T^{e}_{A,B}$ is NOT a simplification of T for φ.



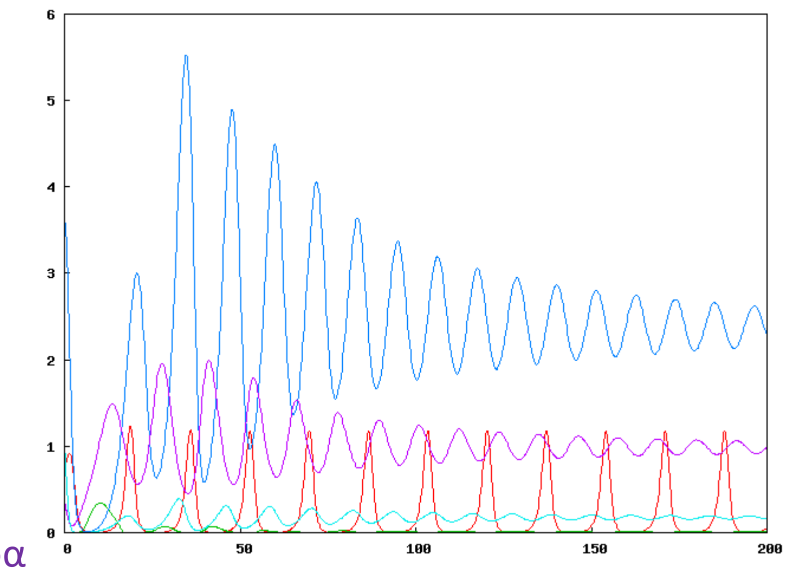
A simplification trace is defined by the points in:

$$J = \{i, i+1 \in \{0,\ldots,n\}] \mid A_{s_i} > B_{s_i} \wedge A_{s_{i+1}} \leq B_{s_{i+1}}\}$$

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-erbα

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

- **Validity domain computing time** (in ms):

		First trace				Second trace			
		variable		fixed		variable		fixed	
		Bef.	Aft.	Bef.	Aft.	Bef.	Aft.	Bef.	Aft.
	Nb of points	971	18	20002	18	1047	35	20002	35
Formula	Solver		34		34		58		58
Reachability of PerCry	generic	12	0	260	4	12	0	204	0
	dedicated	0	0	16	0	4	0	16	0
Minimum amplitude of PerCry	generic	132	0	2728	0	132	4	2516	4
	dedicated	0	0	16	0	4	0	16	0
Local maxima of PerCry	generic	64	0	1308	4	72	4	1316	4
	dedicated	0	0	36	8	4	0	44	4
Distance betw. PerCry peaks	generic	512	12	9584	12	708	80	12373	104
	dedicated	4	4	40	8	32	28	80	48
Distance betw. succ. PerCry peaks	generic	532	12	10980	12	1188	36	23101	156
	dedicated	4	0	40	8	4	0	28	4
Regularity of PerCry peaks	generic	**1700**	**32**	**34818**	**32**	**3056**	**96**	**60776**	**108**
	dedicated	**0**	**0**	**36**	**0**	**4**	**0**	**52**	**20**
Phase betw. PerCry and MPF	generic	456	16	9332	16	496	32	9365	32
	dedicated	4	4	68	12	4	0	76	20

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.