Efficient Static Analysis of Dynamical Properties using the Process Hitting

10 janvier 2012

Loïc Paulevé

École Polytechnique / LIX (équipe AMIB)
pauleve@lix.polytechnique.fr
http://loicpauleve.name

Work with Morgan Magnin and Olivier (F.) Roux (PhD thesis)
Overview

Context

- Computer science for systems biology.
- Abstract (discrete) modelling.

Problematics

- Scalable formal verification.
- Properties of interest:
  - Fixpoint enumeration.
  - Reachability properties.
  - Control (drug targets).
Overview

Context
- Computer science for systems biology.
- Abstract (discrete) modelling.

Problematics
- Scalable formal verification.
- Properties of interest:
  - Fixpoint enumeration.
  - Reachability properties.
  - Control (drug targets).

The Process Hitting [Paulevé, Magnin, Roux in TCSB 2011]
- Subclass of Communicating Finite State Machines / Petri Nets / etc.
- Suitable to model Biological Regulatory Networks.
- Efficient static analysis of dynamical properties.
  [Paulevé, Magnin, Roux in MSCS 2012]
Outline

1. The Process Hitting

2. Static Analysis
   - Fixpoint Enumeration
   - Abstract Interpretation of Successive Reachability Properties
   - Towards Control of Reachability Properties

3. Experiments and Comparisons

4. Conclusion and Outlook
1. The Process Hitting

2. Static Analysis
   - Fixpoint Enumeration
   - Abstract Interpretation of Successive Reachability Properties
   - Towards Control of Reachability Properties

3. Experiments and Comparisons

4. Conclusion and Outlook
Static Analysis of Process Hitting Dynamics: The Process Hitting

Biological Regulatory Networks

\[ f^a(x) = 0 \]
\[ f^b(x) = x[a] \land \neg x[b] \]
\[ f^c(x) = \neg x[b] \land (x[a] \lor x[c]) \]

[René Thomas in Journal of Theoretical Biology, 1973]
The Process Hitting Framework

[Paulevé, Magnin, Roux in TCSB 2011]

- **Sorts**: \(a, b, z\); **Processes**: \(a_0, a_1, b_0, b_1, z_0, z_1, z_2\);
- **Actions**: \(a_0\) hits \(b_1\) to make it bounce to \(b_0\), ...;
- **States**: \(\langle a_1, b_1, z_1 \rangle\), \(\langle a_0, b_1, z_1 \rangle\), \(\langle a_0, b_0, z_1 \rangle\), ...;
- **Restriction of Communicating Finite-State Machines (CFSM).**
The Process Hitting Framework

[Paulevé, Magnin, Roux in TCSB 2011]

- Sorts: a, b, z; Processes: a₀, a₁, b₀, b₁, z₀, z₁, z₂;
- Actions: a₀ hits b₁ to make it bounce to b₀, . . . ;
- States: ⟨a₁, b₁, z₁⟩, ⟨a₀, b₁, z₁⟩, ⟨a₀, b₀, z₁⟩, . . . ;
- Restriction of Communicating Finite-State Machines (CFSM).
The Process Hitting Framework

[Paulevé, Magnin, Roux in TCSB 2011]

- **Sorts**: a, b, z; **Processes**: \(a_0, a_1, b_0, b_1, z_0, z_1, z_2\);
- **Actions**: \(a_0\) hits \(b_1\) to make it **bounce** to \(b_0\), ...;
- **States**: \(\langle a_1, b_1, z_1 \rangle\), \(\langle a_0, b_1, z_1 \rangle\), \(\langle a_0, b_0, z_1 \rangle\), ...;
- **Restriction of Communicating Finite-State Machines (CFSM).**
Sorts: a, b, z; Processes: a₀, a₁, b₀, b₁, z₀, z₁, z₂;

Actions: a₀ hits b₁ to make it bounce to b₀, ...;

States: ⟨a₁, b₁, z₁⟩, ⟨a₀, b₁, z₁⟩, ⟨a₀, b₀, z₁⟩, ...;

Restriction of Communicating Finite-State Machines (CFSM).
The Process Hitting Framework

[Paulevé, Magnin, Roux in TCSB 2011]

- **Sorts**: a, b, z; **Processes**: \(a_0, a_1, b_0, b_1, z_0, z_1, z_2\);
- **Actions**: \(a_0\) hits \(b_1\) to make it **bounce** to \(b_0\), ...;
- **States**: \(\langle a_1, b_1, z_1 \rangle\), \(\langle a_0, b_1, z_1 \rangle\), \(\langle a_0, b_0, z_1 \rangle\), ...;
- **Restriction of Communicating Finite-State Machines (CFSM).**
Static Analysis of Process Hitting Dynamics: The Process Hitting Subclass of Petri Nets

Subclass of Petri Nets
Static Analysis of Process Hitting Dynamics: Static Analysis

Outline

1. The Process Hitting

2. Static Analysis
   - Fixpoint Enumeration
   - Abstract Interpretation of Successive Reachability Properties
   - Towards Control of Reachability Properties

3. Experiments and Comparisons

4. Conclusion and Outlook
Static Analysis of Process Hittings

Intuition

- Simplicity of the Process Hitting $\Rightarrow$ models with simple structures.
- Efficient static derivation of dynamical properties.
Static Analysis of Process Hittings

Intuition

- Simplicity of the Process Hitting $\Rightarrow$ models with simple structures.
- Efficient static derivation of dynamical properties.

Fixed Points

- Reduction to the $n$-cliques of an $n$-partite graph.
Intuition

- Simplicity of the Process Hitting ⇒ models with simple structures.
- Efficient static derivation of dynamical properties.

Fixed Points

- Reduction to the $n$-cliques of an $n$-partite graph.

Successive reachability properties $\mathsf{EF} \ a_i \land (\mathsf{EF} \ b_j \land \ldots)$

- Limited complexity but may be inconclusive (Yes/No/Inconc).
- Abstract interpretation techniques.
- Extraction of key processes (towards control).
1 The Process Hitting

2 Static Analysis
   Fixpoint Enumeration
   Abstract Interpretation of Successive Reachability Properties
   Towards Control of Reachability Properties

3 Experiments and Comparisons

4 Conclusion and Outlook
Fixed Points

[Paulevé, Magnin, Roux in TCSB 2011]

Process Hitting

Hitless graph

$n$-cliques are fixed points
Fixed Points

[Paulevé, Magnin, Roux in TCSB 2011]

Process Hitting

Hitless graph

$n$-cliques are fixed points
**Fixed Points**

[Paulevé, Magnin, Roux in TCSB 2011]

Process Hitting

Hitless graph

$n$-cliques are fixed points
Outline

1. The Process Hitting

2. Static Analysis
   - Fixpoint Enumeration
   - Abstract Interpretation of Successive Reachability Properties
   - Towards Control of Reachability Properties

3. Experiments and Comparisons

4. Conclusion and Outlook
Static Analysis of Process Hitting Dynamics: Static Analysis

**Scenarios**

\[ a_0 \rightarrow c_0 \xrightarrow{} c_1 \rightarrow b_0 \xrightarrow{\top} d_0 \xrightarrow{\top} d_1 \rightarrow c_1 \rightarrow b_0 \xrightarrow{\top} b_1 \rightarrow d_1 \xrightarrow{\top} d_2 \]
a_0 \rightarrow c_0 \xrightarrow{\cdot} c_1 :: b_0 \rightarrow d_0 \xrightarrow{\cdot} d_1 :: c_1 \rightarrow b_0 \xrightarrow{\cdot} b_1 :: b_1 \rightarrow d_1 \xrightarrow{\cdot} d_2
\[a_0 \rightarrow c_0 \Rightarrow c_1 \Rightarrow b_0 \Rightarrow d_0 \Rightarrow d_1 \Rightarrow c_1 \Rightarrow b_0 \Rightarrow b_1 \Rightarrow d_1 \Rightarrow d_2\]
a₀ → c₀ ↦ c₁ :: b₀ → d₀ ↦ d₁ :: c₁ → b₀ ↦ b₁ :: b₁ → d₁ ↦ d₂

Scenarios
Static Analysis of Process Hitting Dynamics: Static Analysis

**Scenarios**

\[ a_0 \rightarrow c_0 \xrightarrow{\mathsf{r}} c_1 \quad \vdash b_0 \xrightarrow{d_0} \xrightarrow{\mathsf{r}} d_1 \xrightarrow{c_1} b_0 \xrightarrow{b_1} d_1 \xrightarrow{d_1} d_2 \]
Successive Reachability $\mathcal{R}$

- Given a Process Hitting $\mathcal{PH}$ with an initial state,
- is it possible to reach the process $a_i$? . . .
- then the process $b_j$? . . . etc.
Static Analysis of Successive Reachability Properties

[Paulevé, Magnin, Roux in MSCS 2012]

**Successive Reachability** $\mathcal{R}$

- Given a Process Hitting $\mathcal{PH}$ with an initial state,
- is it possible to reach the process $a_i$? ...
- then the process $b_j$? ... etc.

**Difficulties:** combinatorial explosion of dynamics to explore.
Static Analysis of Successive Reachability Properties

[Paulevé, Magnin, Roux in MSCS 2012]

Successive Reachability $\mathcal{R}$

- Given a Process Hitting $\mathcal{P}\mathcal{H}$ with an initial state,
- is it possible to reach the process $a_i$? ... 
- then the process $b_j$? ... etc.

Difficulties: combinatorial explosion of dynamics to explore.

Chosen approach

Over-approximations

$\mathcal{P}\mathcal{H}$ does not satisfy $\mathcal{P} \implies \mathcal{R}$ is impossible.

Under-approximations

$\mathcal{P}\mathcal{H}$ satisfies $\mathcal{Q} \implies \mathcal{R}$ is possible.

Requirement: checking $\mathcal{P}$ ($\mathcal{Q}$) is fast.
Abstract Interpretation of Scenarios

Scenarios – Successively playable actions.

- E.g. $\delta = a_0 \rightarrow c_0 \uparrow c_1 :: b_0 \rightarrow d_0 \uparrow d_1 :: c_1 \rightarrow b_0 \uparrow b_1 :: b_1 \rightarrow d_1 \uparrow d_2$.

Context — For each sort, subset of initial processes.

- E.g. $\varsigma = \langle a_0, \{b_0, b_2\}, c_0, d_0 \rangle$.
Abstract Interpretation of Scenarios

Scenarios – Successively playable actions.
- E.g. \( \delta = a_0 \rightarrow c_0 \uparrow c_1 :: b_0 \rightarrow d_0 \uparrow d_1 :: c_1 \rightarrow b_0 \uparrow b_1 :: b_1 \rightarrow d_1 \uparrow d_2 \).

Context — For each sort, subset of initial processes.
- E.g. \( \varsigma = \langle a_0, \{ b_0, b_2 \}, c_0, d_0 \rangle \).

Overall approach
- 2 orthogonal abstractions;
- Bounce Sequences \( \text{BS} \);
- Objective Sequences \( \text{OS} \);
- Concretization: \( \gamma_\varsigma : \text{OS} \mapsto \wp(\text{Sce}) \);
- Refinements: \( \rho : \text{OS} \mapsto \wp(\text{OS}) \);
- \( \gamma_\varsigma(\omega) = \gamma_\varsigma(\rho(\omega)) \).
Two Orthogonal Abstractions

\[ a_0 \rightarrow c_0 \xrightarrow{\star} c_1 :: b_0 \rightarrow d_0 \xrightarrow{\star} d_1 :: c_1 \rightarrow b_0 \xrightarrow{\star} b_1 :: b_1 \rightarrow d_1 \xrightarrow{\star} d_2 \]

Abstraction by Objective Sequences

- \( c_0 \xrightarrow{\star \star} c_1 :: d_0 \xrightarrow{\star \star} d_1 :: b_0 \xrightarrow{\star \star} b_1 :: d_1 \xrightarrow{\star \star} d_2 \)
Two Orthogonal Abstractions

\[ a_0 \rightarrow c_0 \overset{\cdot}{\rightarrow} c_1 :: b_0 \rightarrow d_0 \overset{\cdot}{\rightarrow} d_1 :: c_1 \rightarrow b_0 \overset{\cdot}{\rightarrow} b_1 :: b_1 \rightarrow d_1 \overset{\cdot}{\rightarrow} d_2 \]

Abstraction by Objective Sequences

- \( c_0 \overset{\star}{\rightarrow} c_1 :: d_0 \overset{\star}{\rightarrow} d_1 :: b_0 \overset{\star}{\rightarrow} b_1 :: d_1 \overset{\star}{\rightarrow} d_2 \)
- \( b_0 \overset{\star}{\rightarrow} b_1 :: d_0 \overset{\star}{\rightarrow} d_2 \)
Two Orthogonal Abstractions

\[ a_0 \rightarrow c_0 \uparrow c_1 :: b_0 \rightarrow d_0 \uparrow d_1 :: c_1 \rightarrow b_0 \uparrow b_1 :: b_1 \rightarrow d_1 \uparrow d_2 \]

Abstraction by Objective Sequences

- \( c_0 \uparrow^* c_1 :: d_0 \uparrow^* d_1 :: b_0 \uparrow^* b_1 :: d_1 \uparrow^* d_2 \)
- \( b_0 \uparrow^* b_1 :: d_0 \uparrow^* d_2 \)
- \( d_0 \uparrow^* d_2, ... \)
Two Orthogonal Abstractions

\[ a_0 \rightarrow c_0 \mathrel{\xrightarrow{\dagger}} c_1 :: b_0 \rightarrow d_0 \mathrel{\xrightarrow{\dagger}} d_1 :: c_1 \rightarrow b_0 \mathrel{\xrightarrow{\dagger}} b_1 :: b_1 \rightarrow d_1 \mathrel{\xrightarrow{\dagger}} d_2 \]

Abstraction by Objective Sequences

- \( c_0 \mathrel{\xrightarrow{\dagger}^*} c_1 :: d_0 \mathrel{\xrightarrow{\dagger}^*} d_1 :: b_0 \mathrel{\xrightarrow{\dagger}^*} b_1 :: d_0 \mathrel{\xrightarrow{\dagger}^*} d_2 \)
- \( b_0 \mathrel{\xrightarrow{\dagger}^*} b_1 :: d_0 \mathrel{\xrightarrow{\dagger}^*} d_2 \)
- \( d_0 \mathrel{\xrightarrow{\dagger}^*} d_2, \ldots \)

Abstraction by Bounce Sequences

E.g.: \( b_0 \rightarrow d_0 \mathrel{\xrightarrow{\dagger}} d_1 :: b_1 \rightarrow d_1 \mathrel{\xrightarrow{\dagger}} d_2 (d_0 \mathrel{\xrightarrow{\dagger}^*} d_2) \)
Two Orthogonal Abstractions

\[\begin{align*}
a_0 &\rightarrow c_0 \uparrow c_1 :: b_0 \rightarrow d_0 \uparrow d_1 :: c_1 \rightarrow b_0 \uparrow b_1 :: b_1 \rightarrow d_1 \uparrow d_2
\end{align*}\]

Abstraction by Objective Sequences

- \( c_0 \uparrow^* c_1 :: d_0 \uparrow^* d_1 :: b_0 \uparrow^* b_1 :: d_1 \uparrow^* d_2 \)
- \( b_0 \uparrow^* b_1 :: d_0 \uparrow^* d_2 \)
- \( d_0 \uparrow^* d_2, \ldots \)

Abstraction by Bounce Sequences

E.g.: \( b_0 \rightarrow d_0 \uparrow d_1 :: b_1 \rightarrow d_1 \uparrow d_2 (d_0 \uparrow^* d_2) \)

\[\Rightarrow \text{can be computed off-line:}\]

- \( \text{BS}(d_0 \uparrow^* d_2) = \{ b_0 \rightarrow d_0 \uparrow d_1 :: b_1 \rightarrow d_1 \uparrow d_2, \]
  \(b_2 \rightarrow d_0 \uparrow d_2 \}\};
- \( \text{BS}^\wedge(d_0 \uparrow^* d_2) = \{ \{b_0, b_1\}, \{b_2\}\}.\]
Two Orthogonal Abstractions

\[ a_0 \rightarrow c_0 \xrightarrow{\dagger} c_1 :: b_0 \rightarrow d_0 \xrightarrow{\dagger} d_1 :: c_1 \rightarrow b_0 \xrightarrow{\dagger} b_1 :: b_1 \rightarrow d_1 \xrightarrow{\dagger} d_2 \]

Abstraction by Objective Sequences

- \( c_0 \xrightarrow{\dagger*} c_1 :: d_0 \xrightarrow{\dagger*} d_1 :: b_0 \xrightarrow{\dagger*} b_1 :: d_1 \xrightarrow{\dagger*} d_2 \);
- \( b_0 \xrightarrow{\dagger*} b_1 :: d_0 \xrightarrow{\dagger*} d_2 \);
- \( d_0 \xrightarrow{\dagger*} d_2, \ldots \)

Abstraction by Bounce Sequences

E.g.: \( b_0 \rightarrow d_0 \xrightarrow{\dagger} d_1 :: b_1 \rightarrow d_1 \xrightarrow{\dagger} d_2 \) (\( d_0 \xrightarrow{\dagger*} d_2 \))

\( \Rightarrow \) can be computed off-line:

- \( \text{BS}(d_0 \xrightarrow{\dagger*} d_2) = \{ b_0 \rightarrow d_0 \xrightarrow{\dagger} d_1 :: b_1 \rightarrow d_1 \xrightarrow{\dagger} d_2, b_2 \rightarrow d_0 \xrightarrow{\dagger} d_2 \} \);
- \( \text{BS}^\wedge(d_0 \xrightarrow{\dagger*} d_2) = \{ \{ b_0, b_1 \}, \{ b_2 \} \} \).
- \( \text{BS}(d_1 \xrightarrow{\dagger*} d_2) = \{ b_1 \rightarrow d_1 \xrightarrow{\dagger} d_2, c_1 \rightarrow d_1 \xrightarrow{\dagger} d_0 :: b_2 \rightarrow d_0 \xrightarrow{\dagger} d_2 \} \);
- \( \text{BS}^\wedge(d_1 \xrightarrow{\dagger*} d_2) = \{ \{ b_1 \}, \{ b_2, c_1 \} \} \).
Objective Sequence Refinements

\[ \gamma_s(\omega) = \{ \delta \in \text{Sce} \mid \omega \text{ abstracts } \delta \land \text{support}(\delta) \subseteq s \} \]

Idea: the more details we know, the better \( \gamma_s(\omega) \) should be understood.
Objective Sequence Refinements

\[ \gamma_\varsigma(\omega) = \{ \delta \in \text{Sce} \mid \omega \text{ abstracts} \delta \land \text{support}(\delta) \subseteq \varsigma \}. \]

Idea: the more details we know, the better \( \gamma_\varsigma(\omega) \) should be understood.

Objective Refinement by BS\(^{\wedge}\): \( \rho^{\wedge} \)

<table>
<thead>
<tr>
<th>( \text{Obj} \times \wp(\text{BS}^{\wedge}) )</th>
<th>( \wp(\text{OS}) )</th>
</tr>
</thead>
</table>
| \( d_0 \triangleleft^* d_2 \) \{\{b_0, b_1\}, \{b_2\}\} | \( \star \triangleleft^* b_0 :: b_0 \triangleleft^* b_1 :: d_0 \triangleleft^* d_2, \)  
| | \( \star \triangleleft^* b_1 :: b_1 \triangleleft^* b_0 :: d_0 \triangleleft^* d_2, \)  
| | \( \star \triangleleft^* b_2 :: d_0 \triangleleft^* d_2 \)  
| \( \gamma_\varsigma(d_0 \triangleleft^* d_2) \) | \( = \gamma_\varsigma(\rho^{\wedge}(d_0 \triangleleft^* d_2, \text{BS}^{\wedge}(d_0 \triangleleft^* d_2))) \)  

(Loïc Paulevé)
Objective Sequence Refinements

\[ \gamma_s(\omega) = \{ \delta \in \text{Sce} \mid \omega \text{ abstracts } \delta \land \text{support}(\delta) \subseteq s \}. \]

Idea: the more details we know, the better \( \gamma_s(\omega) \) should be understood.

Objective Refinement by \( BS^\wedge \): \( \rho^\wedge \)

<table>
<thead>
<tr>
<th>( \text{Obj} \times \wp(\text{BS}^\wedge) )</th>
<th>( \wp(\text{OS}) )</th>
</tr>
</thead>
</table>
| \( d_0^{\uparrow\star}d_2 \), \( \{b_0, b_1\}, \{b_2\} \) | \( \ast \downarrow \uparrow b_0 :: b_0^{\uparrow\star}b_1 :: d_0^{\uparrow\star}d_2, \)
|                      | \( \ast \downarrow \uparrow b_1 :: b_1^{\uparrow\star}b_0 :: d_0^{\uparrow\star}d_2, \)
|                      | \( \ast \downarrow \uparrow b_2 :: d_0^{\uparrow\star}d_2 \) |

\[ \gamma_s(d_0^{\uparrow\star}d_2) = \gamma_s(\rho^\wedge(d_0^{\uparrow\star}d_2, BS^\wedge(d_0^{\uparrow\star}d_2))) \]

Generalization to \( OS \) refinements: \( \bar{\rho} \)

<table>
<thead>
<tr>
<th>\text{OS} \times \wp(\text{BS}^\wedge)</th>
<th>\wp(\text{OS})</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \omega, BS^\wedge )</td>
<td>\text{interleave } \left( \begin{array}{c} \omega' \ \omega_1..n-1 \end{array} \right) :: \omega_n..</td>
</tr>
<tr>
<td></td>
<td>\text{where } n \in \mathbb{I}^\omega</td>
</tr>
<tr>
<td></td>
<td>\text{and } \omega' :: \omega_n \in \rho^\wedge(\omega_n, BS^\wedge(\omega_n))</td>
</tr>
</tbody>
</table>

\[ \gamma_s(\omega) = \gamma_s(\bar{\rho}(\omega, BS^\wedge)) \]
Abstract Structure of Process Hitting

\[ \mathcal{A}_\omega, \omega = d_0 \overset{\ast}{\rightarrow} d_2, \varsigma = \langle a_1, \{b_1, b_2\}, c_1, d_0 \rangle \]

Legend

- **Requirement**
  - \( a_j \rightarrow a_i \overset{\ast}{\rightarrow} a_j \)

- **Solution**
  - \( \{b_i, c_j\} \in BS^\wedge(a_i \overset{\ast}{\rightarrow} a_j) \)

- **Continuity**
  - \( a_i \overset{\ast}{\rightarrow} a_j \rightarrow a_k \overset{\ast}{\rightarrow} a_j \)

- **Trivial solution**
  - \( a_i \overset{\ast}{\rightarrow} a_j \rightarrow \)

- **No solution**
  - \( a_i \overset{\ast}{\rightarrow} a_j \downarrow \)
Approximations of Successive Reachability

Over-approximations
- Un-ordered approximation.
- Ordered approximation.
- Ordered Approximation with occurrences order constraints.

Under-approximations
- Un-ordered approximation.
- Ordered approximation.

No / Inconc

Yes / Inconc
Approximations of Successive Reachability

Over-approximations

- Un-ordered approximation.
- Ordered approximation.
- Ordered Approximation with occurrences order constraints.

Under-approximations

- Un-ordered approximation.
- Ordered approximation.

No / Inconc

Yes / Inconc

Successive Reachability
Un-ordered Over-approximation

Example

Necessary condition for $\gamma_\varsigma(\omega) \neq \emptyset$:
From each objective within $\omega$, there exists a traversal of $A_\varsigma^\omega$ such that:
- objective $\rightarrow$ follow at least one solution;
- process $\rightarrow$ follow all objectives;
- no cycle.

\[
\begin{align*}
\bullet \quad & b_0 \overset{\gamma^*}{\rightarrow} b_2 \\
\bullet \quad & c_0 \overset{\gamma^*}{\rightarrow} c_1 \\
\bullet \quad & a_0 \overset{\gamma^*}{\rightarrow} a_1
\end{align*}
\]
Un-ordered Over-approximation

Example

Necessary condition for $\gamma_s(\omega) \neq \emptyset$:
From each objective within $\omega$, there exists a traversal of $A^\omega_s$ such that:
- objective → follow at least one solution;
- process → follow all objectives;
- no cycle.

$\downarrow$

No
Un-ordered Over-approximation

Example

Necessary condition for \( \gamma_\varsigma(\omega) \neq \emptyset \):
From each objective within \( \omega \), there exists a traversal of \( \mathcal{A}_\varsigma^\omega \) such that:
- objective \( \rightarrow \) follow at least one solution;
- process \( \rightarrow \) follow all objectives;
- no cycle.

\[
\begin{align*}
\text{Inconc}
\end{align*}
\]
Approximations of Successive Reachability

Over-approximations
- Un-ordered approximation.
- Ordered approximation.
- Ordered Approximation with occurrences order constraints.

Under-approximations
- Un-ordered approximation.
- Ordered approximation.

Successive Reachability

No / Inconc

Yes / Inconc
Un-ordered Under-approximation

Example

Sufficient condition for $\gamma_\varsigma(\omega) \neq \emptyset$:

- $[B^\omega_\varsigma]$ has no cycle;
- each objective has at least one solution.

$[B^\omega_\varsigma]$: saturated $A^\omega_\varsigma$. 
Sufficient condition for $\gamma_\varsigma(\omega) \neq \emptyset$:

- $[B^\omega_\varsigma]$ has no cycle;
- each objective has at least one solution.

$[B^\omega_\varsigma]$: saturated $A^\omega_\varsigma$. 
Sufficient condition for $\gamma_s(\omega) \neq \emptyset$:

- $[B^\omega]$ has no cycle;
- each objective has at least one solution.

$[B^\omega]$ : saturated $A^\omega$.
Un-ordered Under-approximation

Example

Sufficient condition for $\gamma_\varsigma(\omega) \neq \emptyset$:

- $[\mathcal{B}_\varsigma^\omega]$ has no cycle;
- each objective has at least one solution.

$[\mathcal{B}_\varsigma^\omega]$: saturated $\mathcal{A}_\varsigma^\omega$. 
Un-ordered Under-approximation

Example

Sufficient condition for $\gamma_{s}(\omega) \neq \emptyset$:

- $[B_{s}^{\omega}]$ has no cycle;
- each objective has at least one solution.

$[B_{s}^{\omega}]$: saturated $A_{s}^{\omega}$.
Static Analysis of Successive Reachability

Over-approximations

- Un-ordered approximation.
- Ordered approximation.
- Ordered Approximation with occurrences order constraints.

Under-approximations

- Un-ordered approximation.
- Ordered approximation.

No / Inconc

Still inconclusive?
- Require new analyses of the abstract structure
  ⇒ drive refinements of \( \omega \).

Yes / Inconc
Static Analysis of Process Hitting Dynamics: Static Analysis

**Static Analysis of Successive Reachability**

- **Over-approximations**
  - Un-ordered approximation.
  - Ordered approximation.
  - Ordered Approximation with occurrences order constraints.

- **Under-approximations**
  - Un-ordered approximation.
  - Ordered approximation.

**Still inconclusive?**
- Require new analyses of the abstract structure
- ⇒ drive refinements of $\omega$. 

No / Inconc

Yes / Inconc
Abstract Structures $A_\varsigma^\omega$, $[B_\varsigma^\omega]$

- $BS^\wedge$ computation: exponential in the number of processes within a single sort.
- Size of $BS^\wedge$: combinations of $|\text{Proc}_a|$ processes ($|\text{Proc}_a|$).
- Size of $A_\varsigma^\omega$ (and $[B_\varsigma^\omega]$): polynomial in processes number $\times$ size of $BS^\wedge$.

Analyses

- Over-approximations: polynomial in the size of $A_\varsigma^\omega$.
- Different strategies of under-approximation:
  - global: polynomial in the size of $[B_\varsigma^\omega]$;
  - per solution: $\times$ exponential in the size of $BS^\wedge$.

$\implies$ efficient with a small number of processes per sort, while a very large number of sorts can be handled.
Outline

1. The Process Hitting

2. Static Analysis
   - Fixpoint Enumeration
   - Abstract Interpretation of Successive Reachability Properties
   - Towards Control of Reachability Properties

3. Experiments and Comparisons

4. Conclusion and Outlook
Extraction of Key Processes

Necessary condition for $\gamma_\varsigma(\omega) \neq \emptyset$: From each objective within $\omega$, there exists a traversal of $A^\omega_\varsigma$ such that:

- objective $\rightarrow$ follow at least one solution;
- process $\rightarrow$ follow all objectives;
- no cycle.

Inconc
Extraction of Key Processes

Necessary condition for $\gamma_\varsigma(\omega) \neq \emptyset$: From each objective within $\omega$, there exists a traversal of $A^\omega_\varsigma$ such that:

- objective $\rightarrow$ follow at least one solution;
- process $\rightarrow$ follow all objectives;
- no cycle.
Outline

1. The Process Hitting

2. Static Analysis
   - Fixpoint Enumeration
   - Abstract Interpretation of Successive Reachability Properties
   - Towards Control of Reachability Properties

3. Experiments and Comparisons

4. Conclusion and Outlook
Static Analysis of Process Hitting Dynamics: Experiments and Comparisons

EGFR/ErbB Signalling Network
(104 components)


Process Hitting
193 sorts, 748 processes, 2356 actions:
$\approx 2 \cdot 10^{96}$ states.

Loïc Paulevé
Execution times

- Real biological models.
- Wide-range of biological/arbitrary reachability analysis.
- Always conclusive.

<table>
<thead>
<tr>
<th>Model</th>
<th>sorts</th>
<th>procs</th>
<th>actions</th>
<th>states</th>
<th>Biocham$^1$</th>
<th>libddd</th>
<th>PINT$^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>egfr20</td>
<td>35</td>
<td>196</td>
<td>670</td>
<td>$2^{64}$</td>
<td>[3s-KO]</td>
<td>[1s-150s]</td>
<td>0.007s</td>
</tr>
<tr>
<td>tcrsig40</td>
<td>54</td>
<td>156</td>
<td>301</td>
<td>$2^{73}$</td>
<td>[1s-KO]</td>
<td>[0.6s-KO]</td>
<td>0.004s</td>
</tr>
<tr>
<td>tcrsig94</td>
<td>133</td>
<td>448</td>
<td>1124</td>
<td>$2^{194}$</td>
<td>KO</td>
<td>KO</td>
<td>0.030s</td>
</tr>
<tr>
<td>egfr104</td>
<td>193</td>
<td>748</td>
<td>2356</td>
<td>$2^{320}$</td>
<td>KO</td>
<td>KO</td>
<td>0.050s</td>
</tr>
</tbody>
</table>

$^1$ [Inria Paris-Rocquencourt/Contraintes] using NuSMV2

$^2$ http://process.hitting.free.fr
Outline

1. The Process Hitting

2. Static Analysis
   - Fixpoint Enumeration
   - Abstract Interpretation of Successive Reachability Properties
   - Towards Control of Reachability Properties

3. Experiments and Comparisons

4. Conclusion and Outlook
Conclusion

Static Analysis of Process Hittings

- Static listing of fixed points.
- Very efficient approximations of successive reachability properties.
- Key processes uncovering (necessary to a given reachability) (towards control).
- Make tractable the formal analysis of large Biological Regulatory Networks.

Future work

- Formal link with event structures (such as Petri Nets unfoldings);
- Improve the analysis with libdd?
- Extension to the Process Hitting with Priorities (allows the weak bisimulation of CFSM).
Thank you for your attention.