Towards the use of Process Hitting to tackle biological observations inconsistent with background knowledge

Joint work: Yoshitaka Yamamoto (1), Morgan Magnin (2,3,4), Maxime Folschette (2), Katsumi Inoue (3)

(1) University of Yamanashi, Yamanashi, Japan
(2) Ecole Centrale de Nantes, IRCCyN, Nantes, France
(3) National Institute of Informatics, Tôkyô, Japan
(4) JSPS Fellow

NII-Yamanashi-LRI Workshop
2014/10/07
Motivations

• Given an existing network (background knowledge) and a (new) observation that is inconsistent with this network, how can we automatically design the minimal set of missing actions that can mimic the observation?

• Process Hitting is an efficient framework to cope with large networks (~ 100 components)
Motivations

- Our proposition: design a method taking advantage of the Process Hitting methods to address the completion of networks with inconsistent observations

- Restrictions w.r.t. current work:
  - Consider only addition of actions, not removal of actions
  - Modeling of the evolution of a gene expression in case of ko w.r.t. wild type, under steady state assumption
Overview

• Motivating example
• Reminder about the Process Hitting framework
• 4-level based logics and associated truth tables
• Translating 4-level based models into Process Hitting
• Further discussions
Motivating example

- Background theory $B$: Boolean network consisting of the three Boolean functions
  - $\text{Mig}1p = \text{not GRR}1$
  - $\text{Rgt}1p = \text{not (Mig}1p \& \text{RGT}1)$
  - $\text{YGL}157w = \text{not Rgt}1p$

- Observation $O$:

  When $\text{GRR}1$ is ko, then the gene expression of $\text{YGL}157w$ decreases, i.e.:
  When the gene expression of $\text{GRR}1$ decreases, the gene expression of $\text{YGL}157w$ also decreases.

  (we write it by $\text{promoted(}ygl157w, grr1)$)
Inconsistency between B and O

Given the following initial state, we meet the fact that the gene expression of YGL157w decreases
\( \langle GRR1 = -1, \text{Mig}1p = 0, \text{Rgt}1p = 0, \text{RGT}1 = 0, YGL157w = 0 \rangle \)
⇒ This is inconsistent with the observation...

Observation
\( \text{promoted}(ygl157w, \text{grr}1) \)
Overview

• Motivating example
• Reminder about the Process Hitting framework
• 4-level based logics and associated truth tables
• Translating 4-level based models into Process Hitting
• Further discussions
The Process Hitting modelling

[Paulevé et al., *Transactions on Computational Systems Biology*, 2011]

**Sorts**: components  \( a, b, z \)
The Process Hitting modelling

[Paulevé et al., Transactions on Computational Systems Biology, 2011]

**Sorts:** components \( a, b, z \)

**Processes:** local states / levels of expression \( z_0, z_1, z_2 \)
The Process Hitting modelling

[Paulevé et al., Transactions on Computational Systems Biology, 2011]

**Sorts:** components  \( a, b, z \)

**Processes:** local states / levels of expression  \( z_0, z_1, z_2 \)

**States:** sets of active processes  \( \langle a_0, b_1, z_0 \rangle \)
The Process Hitting modelling

[Paulevé et al., Transactions on Computational Systems Biology, 2011]

**Sorts:** components $a$, $b$, $z$

**Processes:** local states / levels of expression $z_0$, $z_1$, $z_2$

**States:** sets of active processes $\langle a_0, b_1, z_0 \rangle$

**Actions:** dynamics $b_1 \rightarrow z_0 \uparrow z_1$, $a_0 \rightarrow a_0 \uparrow a_1$, $a_1 \rightarrow z_1 \uparrow z_2$
The Process Hitting modelling

[Paulevé et al., Transactions on Computational Systems Biology, 2011]

**Sorts:** components  $a, b, z$

**Processes:** local states / levels of expression  $z_0, z_1, z_2$

**States:** sets of active processes  $\langle a_0, b_1, z_0 \rangle$

**Actions:** dynamics  $b_1 \rightarrow z_0 \rightarrow z_1$, $a_0 \rightarrow a_0 \rightarrow a_1$, $a_1 \rightarrow z_1 \rightarrow z_2$
The Process Hitting modelling

[Paulevé et al., Transactions on Computational Systems Biology, 2011]

**Sorts:** components $a, b, z$

**Processes:** local states / levels of expression $z_0, z_1, z_2$

**States:** sets of active processes $\langle a_0, b_1, z_1 \rangle$

**Actions:** dynamics $b_1 \rightarrow z_0 \uparrow z_1, a_0 \rightarrow a_0 \uparrow a_1, a_1 \rightarrow z_1 \uparrow z_2$
The Process Hitting modelling

[Paulevé et al., Transactions on Computational Systems Biology, 2011]

**Sorts:** components $a, b, z$

**Processes:** local states / levels of expression $z_0, z_1, z_2$

**States:** sets of active processes $\langle a_1, b_1, z_1 \rangle$

**Actions:** dynamics $b_1 \rightarrow z_0 \uparrow z_1, a_0 \rightarrow a_0 \uparrow a_1, a_1 \rightarrow z_1 \uparrow z_2$
The Process Hitting modelling

[Paulevé et al., Transactions on Computational Systems Biology, 2011]

**Sorts:** components $a, b, z$

**Processes:** local states / levels of expression $z_0, z_1, z_2$

**States:** sets of active processes $\langle a_1, b_1, z_2 \rangle$

**Actions:** dynamics $b_1 \rightarrow z_0 \uparrow z_1, a_0 \rightarrow a_0 \uparrow a_1, a_1 \rightarrow z_1 \uparrow z_2$
Static analysis: successive reachability

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

• Initial state

\[ \langle a_1, b_0, c_0, d_0 \rangle \]
Static analysis: successive reachability

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

- Initial state
  \( \langle a_1, b_0, c_0, d_0 \rangle \)

- Objectives
  \[ \left\langle \vdash d_1 \ :: \ \vdash d_2 \right\rangle \]
Static analysis: successive reachability

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

- Initial state
  \[ \langle a_1, b_0, c_0, d_0 \rangle \]
- Objectives
  \[
  \begin{align*}
  & [ \overset{\rightarrow}{d_1} :: \overset{\rightarrow}{d_2} ] \\
  & [ \overset{\rightarrow}{d_1} :: \overset{\rightarrow}{b_1} :: \overset{\rightarrow}{d_2} ]
  \end{align*}
  \]
Static analysis: successive reachability

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

- Initial state
  \( \langle a_1, b_0, c_0, d_0 \rangle \)

- Objectives
  
  \[
  \begin{align*}
  [ \xrightarrow{d_1} &:: \xrightarrow{d_2} ] \\
  [ \xrightarrow{d_1} &:: \xrightarrow{b_1} :: \xrightarrow{d_2} ] \\
  [ \xrightarrow{d_2} ]
  \end{align*}
  \]
Static analysis: successive reachability

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

- Initial state
  \[ \langle a_1, b_0, c_0, d_0 \rangle \]

- Objectives
  \[ \left[ \xrightarrow{d_1} \vdash d_2 \right] \]
  \[ \left[ \xrightarrow{d_1} \vdash b_1 \vdash d_2 \right] \]
  \[ \left[ \xrightarrow{d_2} \right] \]

\[ \rightarrow \text{Concretization of the objective } = \text{scenario} \]
\[ a_0 \rightarrow c_0 \xrightarrow{c_1} b_0 \rightarrow d_0 \xrightarrow{d_1} c_1 \rightarrow b_0 \xrightarrow{b_1} b_1 \rightarrow d_1 \xrightarrow{d_2} \]
Static analysis: successive reachability

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

- Initial state
  \( \langle a_1, b_0, c_0, d_0 \rangle \)
- Objectives
  \[
  \begin{align*}
  & \langle \uparrow d_1 :: \uparrow d_2 \rangle \\
  & \langle \uparrow d_1 :: \uparrow b_1 :: \uparrow d_2 \rangle \\
  & \langle \uparrow d_2 \rangle 
  \end{align*}
  \]

→ Concretization of the objective = scenario

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

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

- Initial state
  \[ \langle a_1, b_0, c_0, d_0 \rangle \]

- Objectives
  \[
  \begin{align*}
  & \{ \overset{\uparrow}{d}_1 :: \overset{\uparrow}{d}_2 \} \\
  & \{ \overset{\uparrow}{d}_1 :: \overset{\uparrow}{b}_1 :: \overset{\uparrow}{d}_2 \} \\
  & \{ \overset{\uparrow}{d}_2 \}
  \end{align*}
  \]

→ Concretization of the objective = scenario

\[ a_0 \rightarrow c_0 \overset{\uparrow}{c}_1 :: b_0 \rightarrow d_0 \overset{\uparrow}{d}_1 :: c_1 \rightarrow b_0 \overset{\uparrow}{b}_1 :: b_1 \rightarrow d_1 \overset{\uparrow}{d}_2 \]
Static analysis: successive reachability

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

- Initial state
  \[ \langle a_1, b_0, c_0, d_0 \rangle \]
- Objectives
  \[
  \begin{align*}
  & \left[ \overset{\rightarrow}{d}_1 :: \overset{\rightarrow}{d}_2 \right] \\
  & \left[ \overset{\rightarrow}{d}_1 :: \overset{\rightarrow}{b}_1 :: \overset{\rightarrow}{d}_2 \right] \\
  & \left[ \overset{\rightarrow}{d}_2 \right]
  \end{align*}
  \]

\[ \text{→ Concretization of the objective = scenario} \]
\[ a_0 \rightarrow c_0 \overset{\rightarrow}{c}_1 :: b_0 \rightarrow d_0 \overset{\rightarrow}{d}_1 :: c_1 \rightarrow b_0 \overset{\rightarrow}{b}_1 :: b_1 \rightarrow d_1 \overset{\rightarrow}{d}_2 \]
Static analysis: successive reachability

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

• Initial state

\[ \langle a_1, b_0, c_0, d_0 \rangle \]

• Objectives

\[ [ \uparrow d_1 :: \uparrow d_2 ] \]
\[ [ \uparrow d_1 :: \uparrow b_1 :: \uparrow d_2 ] \]
\[ [ \uparrow d_2 ] \]

→ Concretization of the objective = scenario

\[ a_0 \rightarrow c_0 \uparrow c_1 :: b_0 \rightarrow d_0 \uparrow d_1 :: c_1 \rightarrow b_0 \uparrow b_1 :: b_1 \rightarrow d_1 \uparrow d_2 \]
Over- and Under-approximations

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

→ Directly checking $R$ is hard (exponential)
→ Rather check approximations $P$ and $Q$ so that: $P \Rightarrow R \Rightarrow Q$
Over- and Under-approximations

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

→ Directly checking $R$ is hard (exponential)
→ Rather check approximations $P$ and $Q$ so that: $P \Rightarrow R \Rightarrow Q$

Over-Approximation

Exact solutions

$R$

$\neg Q$
Over- and Under-approximations

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

→ Directly checking $R$ is hard (exponential)
→ Rather check approximations $P$ and $Q$ so that: $P \Rightarrow R \Rightarrow Q$

Over-Approximation

Exact solutions

$R$

$\neg Q$
Over- and Under-approximations

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

→ Directly checking $R$ is hard (exponential)
→ Rather check approximations $P$ and $Q$ so that: $P \Rightarrow R \Rightarrow Q$

Over-Approximation

Under-Approximation

Exact solutions
Over- and Under-approximations

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

→ Directly checking $R$ is hard \textbf{(exponential)}
→ Rather check \textbf{approximations} $P$ and $Q$ so that: $P \Rightarrow R \Rightarrow Q$
Over- and Under-approximations

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

→ Directly checking $R$ is hard (exponential)
→ Rather check approximations $P$ and $Q$ so that: $P \Rightarrow R \Rightarrow Q$

![Diagram showing Over-Approximation and Under-Approximation with Exact solutions](image-url)
Over- and Under-approximations

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

→ Directly checking $R$ is hard (exponential)
→ Rather check approximations $P$ and $Q$ so that: $P \Rightarrow R \Rightarrow Q$

![Diagram showing over- and under-approximations with sets $P$, $R$, and $Q$.]
Over- and Under-approximations

[Paulevé et al., Mathematical Structures in Computer Science, 2012]

→ Directly checking $R$ is hard (exponential)
→ Rather check approximations $P$ and $Q$ so that: $P \Rightarrow R \Rightarrow Q$

Computing $P$ or $Q$ is much simpler (roughly polynomial)
→ Efficient for big models → Hundredths of seconds
Several additions to improve the expressiveness of the Process Hitting:

- Priorities
  - Groups of actions with similar temporal/probabilistic parameters
- Neutralizing edges
  - Atomistic delay relations between actions
- Synchronous actions
  - Multiple reactants and products → Biochemical reactions

All these formalisms can be translated to a canonical form

A new static analysis has been developed to check reachability properties
→ Efficient dynamic analysis on large models
Priorities

[Folschette et al., CS2Bio’13, 2013]

- Each action is linked to a class of priority
- An action is playable only if no action with a higher priority is playable

1  2  3  ...  n

highest priority

lowest priority
• Each action is linked to a class of priority
• An action is playable only if no action with a higher priority is playable

→ \( b_1 \) can never be reached
Each action is linked to a class of priority.
An action is playable only if no action with a higher priority is playable.

\[ \begin{align*}
1 & \rightarrow b_1 \text{ can never be reached} \\
\end{align*} \]
Priorities
[Folschette et al., CS2Bio’13, 2013]

• Each action is linked to a class of priority
• An action is playable only if no action with a higher priority is playable

1 2 3 ... n

highest priority

lowest priority

• Allows to model classes of actions with similar temporal/stochastic parameters

A B C ... N

instantaneous very fast very slow
Neutralizing edges

- Allows to integrate temporal data about relative reaction delays
- Atomistic preemptions

\[ c_0 \rightarrow d_0 \xrightarrow{\epsilon} d_1 \] cannot be played while \[ a_0 \rightarrow b_0 \xrightarrow{\epsilon} b_1 \] is playable

\[ \rightarrow d_1 \] is always reached after \( b_1 \)
Neutralizing edges

- Allows to integrate temporal data about relative reaction delays
- Atomistic preemptions

\[c_0 \rightarrow d_0 \rightarrow d_1\] cannot be played \textbf{while} \[a_0 \rightarrow b_0 \rightarrow b_1\] is playable

\[\rightarrow d_1\text{ is always reached after } b_1\]
Neutralizing edges

- Allows to integrate temporal data about relative reaction delays
- Atomistic preemptions

$c_0 \rightarrow d_0 \triangleright d_1$ cannot be played while $a_0 \rightarrow b_0 \triangleright b_1$ is playable

$\rightarrow d_1$ is always reached after $b_1$
Synchronous actions

- Synchronization between actions:
  - Presence of catalysts
  - Consumption of reactants
  - Creation of products

- Convenient for biochemical equations: \( X \rightarrow Y \rightarrow Z \) in the following form:
  \[
  \{X_1, Y_1, Z_0\} \rightarrow \{X_0, Z_1\}
  \]

\[
\begin{align*}
  h_1 &= \{c_1\} \rightarrow \{c_0\} \\
  h_2 &= \{a_0, b_1, c_0, d_0\} \rightarrow \{c_1, d_1\}
\end{align*}
\]

All processes of \( A \) must be present to play \( A \rightarrow B \)

After playing \( A \rightarrow B \), all processes of \( B \) are active
Synchronous actions

- Synchronization between actions:
  - Presence of catalysts
  - Consumption of reactants
  - Creation of products

- Convenient for biochemical equations: \(X \rightarrow Y \rightarrow Z\) in the following form:
  \[
  \{X_1, Y_1, Z_0\} \rightarrow \{X_0, Z_1\}
  \]

\[
\begin{align*}
h_1 &= \{c_1\} \rightarrow \{c_0\} \\
h_2 &= \{a_0, b_1, c_0, d_0\} \rightarrow \{c_1, d_1\}
\end{align*}
\]

All processes of \(A\) must be present to play \(A \rightarrow B\)

After playing \(A \rightarrow B\), all processes of \(B\) are active
Synchronous actions

- Synchronization between actions:
  - Presence of catalysts
  - Consumption of reactants
  - Creation of products

- Convenient for biochemical equations:
  \[
  X \stackrel{Y}{\longrightarrow} Z
  \]
  in the following form:
  \[
  \{X_1, Y_1, Z_0\} \mapsto \{X_0, Z_1\}
  \]

\[
\begin{align*}
h_1 &= \{c_1\} \mapsto \{c_0\} \\
h_2 &= \{a_0, b_1, c_0, d_0\} \mapsto \{c_1, d_1\}
\end{align*}
\]

All processes of \(A\) must be present to play \(A \mapsto B\).

After playing \(A \mapsto B\),
all processes of \(B\) are active.
Overview

• Motivating example
• Reminder about the Process Hitting framework
• 4-level based logics and associated truth tables
• Translating 4-level based models into Process Hitting
• Further discussions
Modeling ideas

• 4 cases to consider:
  • The concentration of a component c in ko of a given gene g is **higher** than its concentration in Wild Type (which will be denoted ↑)
  • The concentration of a component c in case of ko of a given gene g is **lower** than its concentration in Wild Type (which then will be denoted ↓)
  • The concentration of a component c in case of ko of a given gene g is **stable** compared to Wild Type (which then will be denoted -)
  • When the evolution of the concentration of a component c between ko and wild type is **unknown**: add a fourth level « unknown" in the logical framework, but not necessary in the Process Hitting final representation.
Our stoichiometric modeling

- **A and B**: denoting the effect by *the complex* of A and B
  - \( \Rightarrow \) Strength: depending on the amount of the complex

- **A or B**: denoting the (individual) effects by A and B
  - \( \Rightarrow \) Strength: depending on the amount of both A and B
Truth table in 4 valued logic (1/2)

\[ \uparrow: \text{ increase. } \downarrow: \text{ decrease. } -: \text{ unchanged.} \]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A and B</th>
<th>A or B</th>
</tr>
</thead>
<tbody>
<tr>
<td>↑</td>
<td>↑</td>
<td>↑</td>
<td>↑</td>
</tr>
<tr>
<td>↑</td>
<td>↓</td>
<td>↓</td>
<td>unknown</td>
</tr>
<tr>
<td>↑</td>
<td>-</td>
<td>-</td>
<td>↑</td>
</tr>
<tr>
<td>↑</td>
<td>unknown</td>
<td>unknown</td>
<td>unknown</td>
</tr>
<tr>
<td>↓</td>
<td>↓</td>
<td>↓</td>
<td>↓</td>
</tr>
<tr>
<td>↓</td>
<td>-</td>
<td>↓</td>
<td>↓</td>
</tr>
<tr>
<td>↓</td>
<td>unknown</td>
<td>↓</td>
<td>unknown</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>unknown</td>
<td>unknown</td>
<td>unknown</td>
</tr>
<tr>
<td>unknown</td>
<td>unknown</td>
<td>unknown</td>
<td>unknown</td>
</tr>
</tbody>
</table>
Truth table in 4 values logic (2/2)

↑: increase. ↓: decrease. -: unchanged.

<table>
<thead>
<tr>
<th>A</th>
<th>¬A</th>
</tr>
</thead>
<tbody>
<tr>
<td>↑</td>
<td>↓</td>
</tr>
<tr>
<td>↓</td>
<td>↑</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>unknown</td>
<td>unknown</td>
</tr>
</tbody>
</table>
Overview

• Motivating example
• Reminder about the Process Hitting framework
• 4-level based logics and associated truth tables
• Translating 4-level based models into Process Hitting
• Further discussions
Principle of the translation of «4 valued logics» into PH

• When A has more than one regulator, use a cooperative sort to update A according to the state of regulators —> need to use priorities in PH

• «unknown» is modeled by modeling every potential underlying behavior
Translating 4 valued logics into Process Hitting: A=B

1. A = B
2. A = not B
3. A = B and C
4. A = B or C
Translating 4 valued logics into Process Hitting

1. \( A = B \)
2. \( A = \text{not } B \)
3. \( A = B \) and \( C \)
4. \( A = B \) or \( C \)
• Maybe add a slide with the translation of $A = B$ and $C$, but the resulting PH is quite complex?
Back to the example

GRR1  Mig1p  Rgt1p  YGL157w

RGT

initial state
Back to the example: one execution
Back to the example: one execution

This result is different from the observation

initial state

next state
And with synchronous semantics?

This result is different from the observation

initial state

next state
Our question

In case that the dynamics of the model does not encompass the observation into any playable scenario of actions... how to detect missing actions as few as possible that can lead the goal state?
Related discussions

• Asynchronous versus synchronous semantics, w.r.t. the addition of priorities

• Compare 4-valued logics with existing approaches with ODEs

• Interest for a cut-sets based approach
Cut-sets in Process Hitting

• Sets of necessary processes that, should they be disabled, would prevent the considered reachability

• Useful to refute a model: if a cut set computed from the model does not prevent the reachability in the concrete (modeled) system, then it is a proof that there exists concrete behaviors that are not reproducible by the model.

• See (Paulevé et al., 2014) and Loïc’s talk last year
Problem setting
(for abduction in process hitting)

- Finding actions for explaining the observation with the background theory (Boolean network)

![Diagram with nodes and arrows representing the boolean network and initial state.]

- Initial state
Problem setting (for abduction in process hitting)

- Finding actions for explaining the observation with the background theory (Boolean network)

Initial state

New actions
Problem setting (for abduction in process hitting)

- Finding actions for explaining the observation with the background theory (Boolean network)
Problem setting (for abduction in process hitting)

- Finding actions for explaining the observation with the background theory (Boolean network)

We can have observation
Overview

• Motivating example
• Reminder about the Process Hitting framework
• 4-level based logics and associated truth tables
• Translating 4-level based models into Process Hitting
• Further discussions
Research plan and future work

- **Formalize** an algorithmic approach to tackle this completion problem

- Study models with **feedback loops** and extend the principle of 4-valued logics

- Tackle models with **time series data** as input