Integrating prior knowledge in Automatic Network Reconstruction

M. Favre W. Marwan **A. Wagler**

Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes (LIMOS)
UMR CNRS 6158 – Université Blaise Pascal,Clermont-Ferrand, France

Magdeburg Center for Systems Biology (MaCS), Otto-von-Guericke-Universität, Magdeburg, Germany

BioPPN 2014

TUNIS

June 23, 2014

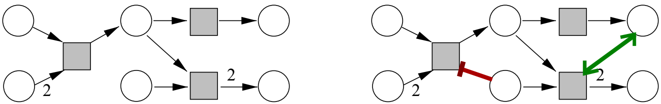
Outline

Outline

Petri nets: The networks

Standard networks $\mathcal{P} = (P, T, A, w)$

- P set of involved **components** ("places" $\bigcirc$), (molecules, receptors, genes)
- T set of involved **reactions** ("transitions" $\square$), (reactions, activations,...)
- $A \subseteq (P \times T) \cup (T \times P)$ set of directed **links** ("arcs" $\rightarrow$),
- arc weights w as stoichiometric coefficients.



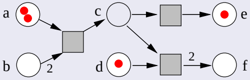
Standard network   Extended network

Extended networks $\mathcal{P} = (P, T, (A \cup A_R \cup A_I), w)$

- $A_R \subset P \times T$ **read-arcs**
- $A_I \subset P \times T$ **inhibitor-arcs**

States

- Each place $p \in P$ can be marked with an integral number $\mathbf{x}_p$ of tokens.
- A **state** can be represented as a vector $\mathbf{x} \in \mathbb{N}^{|P|}$ with entries $\mathbf{x}_p$ for all $p \in P$.



$$x = \begin{pmatrix} 2 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Switching transitions

- In a **standard** network, $t \in T$ is **enabled** at a state $\mathbf{x}$ if

  $\mathbf{x}_p \geq w(p,t)$ for all p with $(p,t) \in A$, and

  switching t yiels a new state $\mathbf{x}'$ with $x'_p = x_p + w(p,t)\ \forall (p,t),(t,p) \in A$.
- In an **extended** network, $t \in T$ is **enabled** at a state $\mathbf{x}$ if in addition

  $\mathbf{x}_p \geq w(p,t)$ for all p with $(p,t) \in A_R$, and,
  $\mathbf{x}_p < w(p,t)$ for all p with $(p,t) \in A_I$.

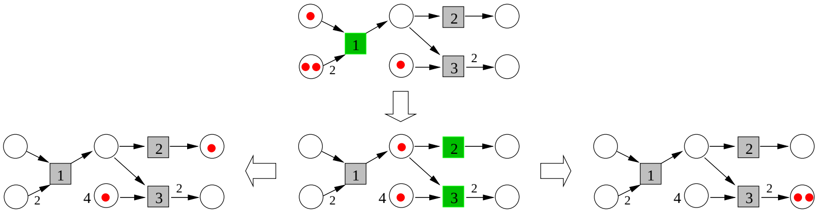
Dynamic processes and reachability

Dynamic processes

Dynamic processes correspond to sequences of system states, obtained by sequences of transition switches.

Starting from an initial state $\mathbf{x}^0$, explore the **dynamic behavior** of the system by

- consecutively switching enabled transitions,
- analyzing the reachability of certain system states.



Can we control transition switches and, thus, the dynamic behavior?

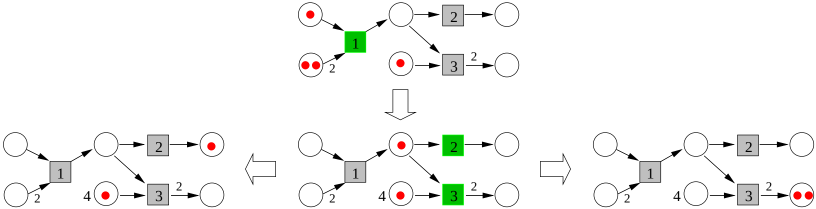
Dynamic processes and reachability

Dynamic processes

Dynamic processes correspond to sequences of system states, obtained by sequences of transition switches.

Starting from an initial state $\mathbf{x}^0$, explore the **dynamic behavior** of the system by

- consecutively switching enabled transitions,
- analyzing the reachability of certain system states.



Can we control transition switches and, thus, the dynamic behavior?

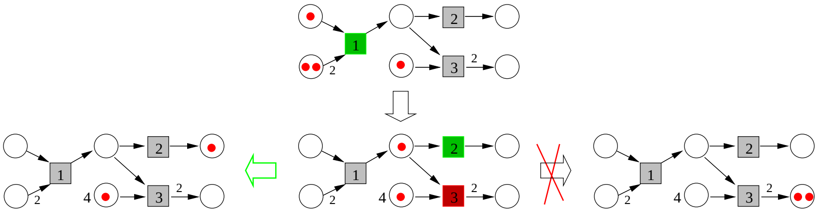
Prediction of the dynamic behavior

Priority relations for transitions (Marwan, W. & Weismantel 2008)

Let G be a network and $\mathcal{O}$ a **priority relation** on its transitions.

- If there are two or more transitions enabled at a state, this transition with **highest priority** will be switched.
- This allows to **predict the dynamic behavior** of the system (instead of listing all potentially possible switching sequences).

Example. Consider the network G with $\mathcal{O} = \{t_2 > t_3\}$.



Outline

The Input

The input $(P, \mathcal{I}_P, \mathcal{X}')$ consists of

- a set P of studied components
- the set $\mathcal{I}_P$ of all known P-invariants
- experimental time-series data $\mathcal{X}' = \{\mathcal{X}'(\mathbf{x}^1, \mathbf{x}^k) : \mathbf{x}^1 \in \mathcal{X}'_{ini}, \mathbf{x}^k \in \mathcal{X}'_{term}\}$ with
  - initial states $\mathcal{X}'_{ini} \subseteq \mathcal{X}'$, terminal states $\mathcal{X}'_{term} \subseteq \mathcal{X}'$
  - time series $\mathcal{X}'(\mathbf{x}^1, \mathbf{x}^k) = (\mathbf{x}^0; \mathbf{x}^1, \ldots, \mathbf{x}^j, \ldots, \mathbf{x}^k)$

Required properties of $\mathcal{X}'$

- **reproducibility:** for each $\mathbf{x}^j \in \mathcal{X}'$ there is a unique observed successor state $\mathrm{succ}_{\mathcal{X}'}(\mathbf{x}^j) = \mathbf{x}^{j+1} \in \mathcal{X}'$
  (can be ensured by pre-processing (W., Wegener 2013))
- **monotonicity:** for each pair $(\mathbf{x}^j, \mathbf{x}^{j+1}) \in \mathcal{X}'$, the values of the elements do not oscillate in the possible intermediate states between $\mathbf{x}^j$ and $\mathbf{x}^{j+1}$
  (depends on the chosen time-points (Durzinsky, W., Weismantel 2008))

The Input

The input $(P, \mathcal{I}_P, \mathcal{X}')$ consists of

- a set P of studied components
- the set $\mathcal{I}_P$ of all known P-invariants
- experimental time-series data $\mathcal{X}' = \{\mathcal{X}'(\mathbf{x}^1, \mathbf{x}^k) : \mathbf{x}^1 \in \mathcal{X}'_{ini}, \mathbf{x}^k \in \mathcal{X}'_{term}\}$ with
  - initial states $\mathcal{X}'_{ini} \subseteq \mathcal{X}'$, terminal states $\mathcal{X}'_{term} \subseteq \mathcal{X}'$
  - time series $\mathcal{X}'(\mathbf{x}^1, \mathbf{x}^k) = (\mathbf{x}^0; \mathbf{x}^1, \ldots, \mathbf{x}^j, \ldots, \mathbf{x}^k)$

Required properties of $\mathcal{X}'$

- **reproducibility:** for each $\mathbf{x}^j \in \mathcal{X}'$ there is a unique observed successor state $\text{succ}_{\mathcal{X}'}(\mathbf{x}^j) = \mathbf{x}^{j+1} \in \mathcal{X}'$
  (can be ensured by pre-processing (W., Wegener 2013))
- **monotonicity:** for each pair $(\mathbf{x}^j, \mathbf{x}^{j+1}) \in \mathcal{X}'$, the values of the elements do not oscillate in the possible intermediate states between $\mathbf{x}^j$ and $\mathbf{x}^{j+1}$
  (depends on the chosen time-points (Durzinsky, W., Weismantel 2008))

Running example: Input data

Example: Light-induced sporulation of *Physarum polycephalum*

Experimental biological data from Starostzik and Marwan (1995)

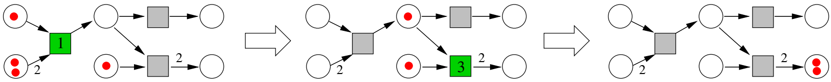
$$
\begin{array}{lll}
P = \{FR, R, P_{FR}, P_R, G, Spo\}, & \mathcal{X}'(\mathbf{x}^1, \mathbf{x}^4) = (\mathbf{x}^0; \mathbf{x}^1, \mathbf{x}^2, \mathbf{x}^3, \mathbf{x}^4), & \mathcal{X}'_{ini} = \{\mathbf{x}^1, \mathbf{x}^5, \mathbf{x}^6\}, \\
\mathcal{I}_P = \{P_{FR}, P_R\}, & \mathcal{X}'(\mathbf{x}^5, \mathbf{x}^0) = (\mathbf{x}^2; \mathbf{x}^5, \mathbf{x}^0), & \mathcal{X}'_{term} = \{\mathbf{x}^4, \mathbf{x}^0, \mathbf{x}^8\} \\
 & \mathcal{X}'(\mathbf{x}^6, \mathbf{x}^8) = (\mathbf{x}^3; \mathbf{x}^6, \mathbf{x}^7, \mathbf{x}^8), &
\end{array}
$$

serve as input for the algorithm, we present all observed states schematically:

$$
\begin{tikzcd}
\overset{\boldsymbol{x}}{\begin{pmatrix} \boldsymbol{x}_{FR} \\ \boldsymbol{x}_{R} \\ \boldsymbol{x}_{P_{FR}} \\ \boldsymbol{x}_{P_R} \\ \boldsymbol{x}_{G} \\ \boldsymbol{x}_{Spo} \end{pmatrix}}
& \overset{\boldsymbol{x}^0}{\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}} \arrow[r, dashed, "\mathrm{FR}"]
& \overset{\boldsymbol{x}^1}{\begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}} \arrow[r, "d^1"]
& \overset{\boldsymbol{x}^2}{\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}} \arrow[r, "d^2"] \arrow[d, dashed, "\mathrm{R}"]
& \overset{\boldsymbol{x}^3}{\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}} \arrow[rr, "d^3"] \arrow[d, dashed, "\mathrm{R}"]
& & \overset{\boldsymbol{x}^4}{\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}} \\
& & & \overset{\boldsymbol{x}^5}{\begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}} \arrow[ull, bend left, "d^4"]
& \overset{\boldsymbol{x}^6}{\begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}} \arrow[r, "d^5"]
& \overset{\boldsymbol{x}^7}{\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}} \arrow[r, "d^6"]
& \overset{\boldsymbol{x}^8}{\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{pmatrix}}
\end{tikzcd}
$$

Output: $\mathcal{X}'$-deterministic extended Petri nets with priorities

- Time series $(\mathbf{x}^0, \mathbf{x}^1, \dots, \mathbf{x}^k)$ are sequences of measured system states.
- A network **fits** the given data $\mathcal{X}'$ if it can **reproduce** the observations:



- For given $\mathcal{X}'$, a network $\mathcal{P} = (P, T, (A \cup A_R \cup A_I), w)$ with priorities $\mathcal{O}$ is $\mathcal{X}'$-**deterministic** if it contains and correctly selects transitions to reach, from every observed state $\mathbf{x}^j \in \mathcal{X}'$, its observed successor $\mathbf{x}^{j+1} \in \mathcal{X}'$.

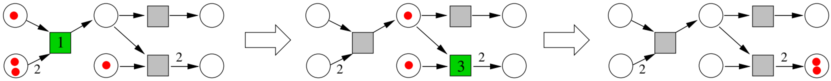
Output

all $\mathcal{X}'$-deterministic extended Petri nets with priorities

Remark: Finding all networks fitting the data means to create all **minimal** ones (that do not reproduce the data anymore after removing any element).

Output: $\mathcal{X}'$-deterministic extended Petri nets with priorities

- Time series $(\mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^k)$ are sequences of measured system states.
- A network **fits** the given data $\mathcal{X}'$ if it can **reproduce** the observations:



- For given $\mathcal{X}'$, a network $\mathcal{P} = (P, T, (A \cup A_R \cup A_I), w)$ with priorities $\mathcal{O}$ is $\mathcal{X}'$-**deterministic** if it contains and correctly selects transitions to reach, from every observed state $\mathbf{x}^j \in \mathcal{X}'$, its observed successor $\mathbf{x}^{j+1} \in \mathcal{X}'$.

Output

all $\mathcal{X}'$-deterministic extended Petri nets with priorities

Remark: Finding all networks fitting the data means to create all **minimal** ones (that do not reproduce the data anymore after removing any element).

Step 1: Representation of observed responses

- Time series $(\mathbf{x}^0; \mathbf{x}^1 \ldots, \mathbf{x}^k)$ may not consist of consecutive system states, as between $(\mathbf{x}^j, \mathbf{x}^{j+1}) \in \mathcal{X}'$, there can be **non-observed intermediate states.**
- To reach $\mathbf{x}^{j+1}$ from $\mathbf{x}^j$, we then need a **sequence** of transitions $t_i, \ldots, t_k$ whose update vectors satisfy $\mathbf{x}^{j+1} - \mathbf{x}^j = \mathbf{r}^{t_i} + \ldots + \mathbf{r}^{t_k}$.

Theorem (Durzinsky, W., Weismantel 2008)

To represent the difference $\mathbf{d} = \mathbf{x}^{j+1} - \mathbf{x}^j$, use only sign-compatibel vectors from

$$\text{Box}(\mathbf{d}) = \left\{ \mathbf{r} \in \mathbb{Z}^{|P|} : \begin{array}{rl} 0 \leq r_p \leq d_p & \text{if } d_p > 0 \\ d_p \leq r_p \leq 0 & \text{if } d_p < 0 \\ r_p = 0 & \text{if } d_p = 0 \\ \sum_{p \in P'} r_p = 0 & \forall P' \in \mathcal{I}_P \end{array} \right\} \setminus \{0\}.$$

For each $\mathbf{d} = \mathbf{x}^{j+1} - \mathbf{x}^j$, determine all solutions $\boldsymbol{\lambda} \in \Lambda(\mathbf{d})$ with

$$\mathbf{d} = \sum_{\mathbf{r}^t \in Box(\mathbf{d})} \lambda_t \, \mathbf{r}^t, \ \lambda_t \in \mathbb{Z}_+$$

and let $\mathcal{R}(\mathbf{d}, \boldsymbol{\lambda}) = \{\mathbf{r}^t \in \text{Box}(\mathbf{d}) : \lambda_t \neq 0\}$ be the set of used update vectors.

Step 1: Representation of observed responses

- Time series $(\mathbf{x}^0; \mathbf{x}^1 \ldots, \mathbf{x}^k)$ may not consist of consecutive system states, as between $(\mathbf{x}^j, \mathbf{x}^{j+1}) \in \mathcal{X}'$, there can be **non-observed intermediate states.**
- To reach $\mathbf{x}^{j+1}$ from $\mathbf{x}^j$, we then need a **sequence** of transitions $t_i, \ldots, t_k$ whose update vectors satisfy $\mathbf{x}^{j+1} - \mathbf{x}^j = \mathbf{r}^{t_i} + \ldots + \mathbf{r}^{t_k}$.

Theorem (Durzinsky, W., Weismantel 2008)

To represent the difference $\mathbf{d} = \mathbf{x}^{j+1} - \mathbf{x}^j$, use only sign-compatibel vectors from

$$\mathrm{Box}(\mathbf{d}) = \left\{ \mathbf{r} \in \mathbb{Z}^{|P|} : \begin{array}{rl} 0 \leq r_p \leq d_p & \text{if } d_p > 0 \\ d_p \leq r_p \leq 0 & \text{if } d_p < 0 \\ r_p = 0 & \text{if } d_p = 0 \\ \sum_{p \in P'} r_p = 0 & \forall P' \in \mathcal{I}_P \end{array} \right\} \setminus \{0\}.$$

For each $\mathbf{d} = \mathbf{x}^{j+1} - \mathbf{x}^j$, determine all solutions $\boldsymbol{\lambda} \in \Lambda(\mathbf{d})$ with

$$\mathbf{d} = \sum_{\mathbf{r}^t \in Box(\mathbf{d})} \lambda_t \, \mathbf{r}^t, \ \lambda_t \in \mathbb{Z}_+$$

and let $\mathcal{R}(\mathbf{d}, \boldsymbol{\lambda}) = \{\mathbf{r}^t \in \mathrm{Box}(\mathbf{d}) : \lambda_t \neq 0\}$ be the set of used update vectors.

Step 1: Representation of observed responses

- Time series $(\mathbf{x}^0; \mathbf{x}^1 \ldots, \mathbf{x}^k)$ may not consist of consecutive system states, as between $(\mathbf{x}^j, \mathbf{x}^{j+1}) \in \mathcal{X}'$, there can be **non-observed intermediate states.**
- To reach $\mathbf{x}^{j+1}$ from $\mathbf{x}^j$, we then need a **sequence** of transitions $t_i, \ldots, t_k$ whose update vectors satisfy $\mathbf{x}^{j+1} - \mathbf{x}^j = \mathbf{r}^{t_i} + \ldots + \mathbf{r}^{t_k}$.

Theorem (Durzinsky, W., Weismantel 2008)

To represent the difference $\mathbf{d} = \mathbf{x}^{j+1} - \mathbf{x}^j$, use only sign-compatibel vectors from

$$\mathrm{Box}(\mathbf{d}) = \left\{ \mathbf{r} \in \mathbb{Z}^{|P|} : \begin{array}{rl} 0 \leq r_p \leq d_p & \text{if } d_p > 0 \\ d_p \leq r_p \leq 0 & \text{if } d_p < 0 \\ r_p = 0 & \text{if } d_p = 0 \\ \sum_{p \in P'} r_p = 0 & \forall P' \in \mathcal{I}_P \end{array} \right\} \setminus \{0\}.$$

For each $\mathbf{d} = \mathbf{x}^{j+1} - \mathbf{x}^j$, determine all solutions $\boldsymbol{\lambda} \in \Lambda(\mathbf{d})$ with

$$\mathbf{d} = \sum_{\mathbf{r}^t \in Box(\mathbf{d})} \lambda_t \, \mathbf{r}^t, \ \lambda_t \in \mathbb{Z}_+$$

and let $\mathcal{R}(\mathbf{d}, \boldsymbol{\lambda}) = \{\mathbf{r}^t \in \mathrm{Box}(\mathbf{d}) : \lambda_t \neq 0\}$ be the set of used update vectors.

Step 1: Representation of observed responses (cont.)

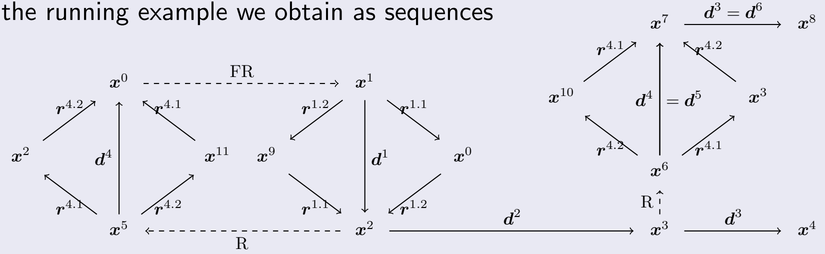
- Every permutation $\pi = (\mathbf{r}^{t_1}, \ldots, \mathbf{r}^{t_m})$ of the elements of a set $\mathcal{R}(\mathbf{d}^j, \boldsymbol{\lambda})$ yields a sequence of intermediate states $\mathbf{x}^j = \mathbf{y}^1, \mathbf{y}^2, ..., \mathbf{y}^m, \mathbf{y}^{m+1} = \mathbf{x}^{j+1}$ with

$$\sigma_{\pi,\boldsymbol{\lambda}}(\mathbf{x}^j, \mathbf{d}^j) = \left((\mathbf{y}^1, \mathbf{r}^{t_1}), (\mathbf{y}^2, \mathbf{r}^{t_2}), \ldots, (\mathbf{y}^m, \mathbf{r}^{t_m})\right).$$

- Every sequence σ respects monotonicity and induces a priority relation $\mathcal{O}_\sigma$.

Example

For the running example we obtain as sequences



with $\mathbf{x}^9 = (1,0,0,1,0,0)^T$, $\mathbf{x}^{10} = (0,1,1,0,1,0)^T$ and $\mathbf{x}^{11} = (0,1,1,0,0,0)^T$.

Step 2: Detecting priority conflicts between sequences

Sequences and their conflicts

Sequences σ and σ' are in **priority conflict** if there are update vectors $\mathbf{r}^t \neq \mathbf{r}^{t'}$ and intermediate states $\mathbf{y}, \mathbf{y}'$ such that both t, t' are enabled at $\mathbf{y}, \mathbf{y}'$ but $(\mathbf{y}, \mathbf{r}^t) \in \sigma$ and $(\mathbf{y}', \mathbf{r}^{t'}) \in \sigma'$ (since this implies $t > t'$ in $\mathcal{O}_\sigma$ but $t' > t$ in $\mathcal{O}_{\sigma'}$).

weak priority conflict (WPC)

if $\mathbf{y} \neq \mathbf{y}'$, the priority conflict **can** be resolved by adding appropriate control-arcs

strong priority conflict (SPC)

if $\mathbf{y} = \mathbf{y}'$, the priority conflict **cannot** be resolved by adding appropriate control-arcs

Note: we have a SPC between the trivial sequence $\sigma(\mathbf{x}^k, \mathbf{0})$ for any terminal state $\mathbf{x}^k$ and any sequence σ containing $\mathbf{x}^k$ as intermediate state.

Example: $\sigma_{\pi_1,\lambda^2}(\mathbf{x}^1, \mathbf{d}^1) = ((\mathbf{x}^1, \mathbf{r}^{1.1}), (\mathbf{x}^0, \mathbf{r}^{1.2}))$ and $\sigma(\mathbf{x}^0, \mathbf{0})$ are in SPC.

Task: select one sequence per $\mathbf{d}^j$ s.t. no SPCs occur between them.

Step 2: Detecting priority conflicts between sequences

Sequences and their conflicts

Sequences σ and σ' are in **priority conflict** if there are update vectors $\mathbf{r}^t \neq \mathbf{r}^{t'}$ and intermediate states $\mathbf{y}, \mathbf{y}'$ such that both t, t' are enabled at $\mathbf{y}, \mathbf{y}'$ but $(\mathbf{y}, \mathbf{r}^t) \in \sigma$ and $(\mathbf{y}', \mathbf{r}^{t'}) \in \sigma'$ (since this implies $t > t'$ in $\mathcal{O}_\sigma$ but $t' > t$ in $\mathcal{O}_{\sigma'}$).

weak priority conflict (WPC)

if $\mathbf{y} \neq \mathbf{y}'$, the priority conflict **can** be resolved by adding appropriate control-arcs

strong priority conflict (SPC)

if $\mathbf{y} = \mathbf{y}'$, the priority conflict **cannot** be resolved by adding appropriate control-arcs

Note: we have a SPC between the trivial sequence $\sigma(\mathbf{x}^k, \mathbf{0})$ for any terminal state $\mathbf{x}^k$ and any sequence σ containing $\mathbf{x}^k$ as intermediate state.

Example: $\sigma_{\pi_1, \lambda^2}(\mathbf{x}^1, \mathbf{d}^1) = ((\mathbf{x}^1, \mathbf{r}^{1.1}), (\mathbf{x}^0, \mathbf{r}^{1.2}))$ and $\sigma(\mathbf{x}^0, \mathbf{0})$ are in SPC.

Task: select one sequence per $\mathbf{d}^j$ s.t. no SPCs occur between them.

Step 2: Detecting priority conflicts between sequences

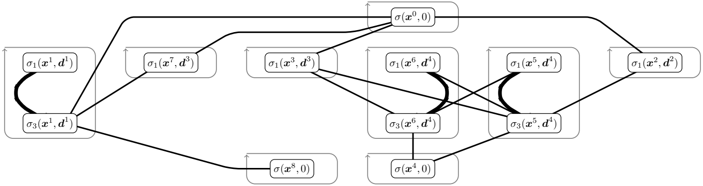
Priority conflict graph $\mathcal{G} = (V_D \cup V_{term}, E_D \cup E_W \cup E_S)$

the nodes correspond to sequences:

- V_D: the sequence $\sigma_{\pi,\boldsymbol{\lambda}}(\mathbf{x}^j, \mathbf{d}^j)$ for all permutations π of $\mathcal{R}(\mathbf{d}^j, \boldsymbol{\lambda})$
- V_{term}: the trivial sequence $\sigma(\mathbf{x}^k, \mathbf{0})$ for all terminal states $\mathbf{x}^k$

the edges correspond to their priority conflicts:

- E_D: all SPCs between two sequences σ, σ' from the same difference vector
- E_S: all SPCs between sequences σ, σ' from distinct difference vectors
- E_W: all WPCs between sequences σ, σ' from distinct difference vectors



In $\mathcal{G}$, there are 8 possible selections S avoiding SPCs.

Step 2: Detecting priority conflicts between sequences

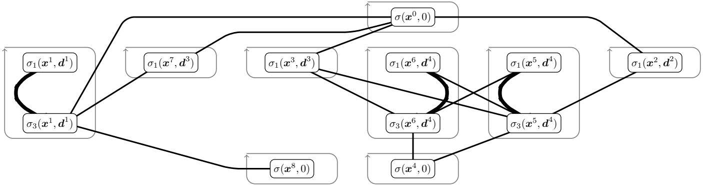
Priority conflict graph $\mathcal{G} = (V_D \cup V_{term}, E_D \cup E_W \cup E_S)$

the nodes correspond to sequences:

- V_D: the sequence $\sigma_{\pi,\boldsymbol{\lambda}}(\mathbf{x}^j, \mathbf{d}^j)$ for all permutations π of $\mathcal{R}(\mathbf{d}^j, \boldsymbol{\lambda})$
- V_{term}: the trivial sequence $\sigma(\mathbf{x}^k, \mathbf{0})$ for all terminal states $\mathbf{x}^k$

the edges correspond to their priority conflicts:

- E_D: all SPCs between two sequences σ, σ' from the same difference vector
- E_S: all SPCs between sequences σ, σ' from distinct difference vectors
- E_W: all WPCs between sequences σ, σ' from distinct difference vectors



In $\mathcal{G}$, there are 8 possible selections S avoiding SPCs.

Step 3: Constructing $\mathcal{X}'$-deterministic Petri nets

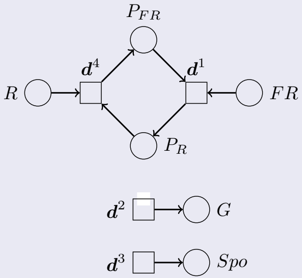
Composition of standard networks

For every selected set S, we obtain a standard network $\mathcal{P}_S$ by deriving transitions from the update vectors in the sets $\mathcal{R}(\mathbf{d}^j, \boldsymbol{\lambda})$ of the selected sequences $\sigma \in S$.

Example

In solution S_1 selecting all canonical sequences $\sigma_1(\mathbf{x}^j, \mathbf{d}^j)$, there are three WPCs:

- between $\sigma_1(\mathbf{x}^2, \mathbf{d}^2)$ and $\sigma(\mathbf{x}^0, \mathbf{0})$ since $\mathbf{d}^2, \mathbf{0}$ are enabled at $\mathbf{x}^2, \mathbf{x}^0$
- between $\sigma_1(\mathbf{x}^3, \mathbf{d}^3)$ and $\sigma(\mathbf{x}^0, \mathbf{0})$ since $\mathbf{d}^3, \mathbf{0}$ are enabled at $\mathbf{x}^3, \mathbf{x}^0$
- between $\sigma_1(\mathbf{x}^7, \mathbf{d}^3)$ and $\sigma(\mathbf{x}^0, \mathbf{0})$ since $\mathbf{d}^3, \mathbf{0}$ are enabled at $\mathbf{x}^7, \mathbf{x}^0$



Step 3: Constructing $\mathcal{X}'$-deterministic Petri nets (cont.)

A WPC(σ, σ') due to $(\mathbf{y}, \mathbf{r}^t) \in \sigma$ and $(\mathbf{y}', \mathbf{r}^{t'}) \in \sigma'$ can be **resolved** by

- either disabling t at state $\mathbf{y}'$ (and adding priority $t > t'$),
- or disabling t' at state $\mathbf{y}$ (and adding priority $t < t'$).

This can be done by inserting control-arcs from CA(σ, σ') which partitions into a subset $\text{CA}_{t>t'}(\sigma, \sigma')$ containing

- a read-arc (p, t) with weight $w(p, t) > y'_p$ for all p with $y_p > y'_p$,
- an inhibitor-arc (p, t) with $w(p, t) < y_p$ for all p with $y_p < y'_p$,

and a subset $\text{CA}_{t<t'}(\sigma, \sigma')$ defined analogously.

Remark: If one of $\mathbf{y}, \mathbf{y}'$ is a terminal state, say $\mathbf{y}'$, one of the alternatives is not possible, then t has to be disabled at $\mathbf{y}'$ and $t > t' = \mathbf{0}$ holds automatically.

- Inserting one control-arc from each CA(σ, σ') resolves all WPCs.
- The obtained extended Petri nets can be made $\mathcal{X}'$-deterministic by adding the requested priorities $t > t'$ or $t < t'$.

Step 3: Constructing $\mathcal{X}'$-deterministic Petri nets (cont.)

A WPC(σ, σ') due to $(\mathbf{y}, \mathbf{r}^t) \in \sigma$ and $(\mathbf{y}', \mathbf{r}^{t'}) \in \sigma'$ can be **resolved** by

- either disabling t at state $\mathbf{y}'$ (and adding priority $t > t'$),
- or disabling t' at state $\mathbf{y}$ (and adding priority $t < t'$).

This can be done by inserting control-arcs from CA(σ, σ') which partitions into a subset CA$_{t>t'}(\sigma, \sigma')$ containing

- a read-arc (p, t) with weight $w(p, t) > y'_p$ for all p with $y_p > y'_p$,
- an inhibitor-arc (p, t) with $w(p, t) < y_p$ for all p with $y_p < y'_p$,

and a subset CA$_{t<t'}(\sigma, \sigma')$ defined analogously.

Remark: If one of $\mathbf{y}, \mathbf{y}'$ is a terminal state, say $\mathbf{y}'$, one of the alternatives is not possible, then t has to be disabled at $\mathbf{y}'$ and $t > t' = \mathbf{0}$ holds automatically.

- Inserting one control-arc from each CA(σ, σ') resolves all WPCs.
- The obtained extended Petri nets can be made $\mathcal{X}'$-deterministic by adding the requested priorities $t > t'$ or $t < t'$.

Step 3: Constructing $\mathcal{X}'$-deterministic Petri nets (cont.)

A WPC(σ, σ') due to $(\mathbf{y}, \mathbf{r}^t) \in \sigma$ and $(\mathbf{y}', \mathbf{r}^{t'}) \in \sigma'$ can be **resolved** by

- either disabling t at state $\mathbf{y}'$ (and adding priority $t > t'$),
- or disabling t' at state $\mathbf{y}$ (and adding priority $t < t'$).

This can be done by inserting control-arcs from CA(σ, σ') which partitions into a subset CA$_{t>t'}(\sigma, \sigma')$ containing

- a read-arc (p, t) with weight $w(p, t) > y'_p$ for all p with $y_p > y'_p$,
- an inhibitor-arc (p, t) with $w(p, t) < y_p$ for all p with $y_p < y'_p$,

and a subset CA$_{t<t'}(\sigma, \sigma')$ defined analogously.

Remark: If one of $\mathbf{y}, \mathbf{y}'$ is a terminal state, say $\mathbf{y}'$, one of the alternatives is not possible, then t has to be disabled at $\mathbf{y}'$ and $t > t' = \mathbf{0}$ holds automatically.

- Inserting one control-arc from each CA(σ, σ') resolves all WPCs.
- The obtained extended Petri nets can be made $\mathcal{X}'$-deterministic by adding the requested priorities $t > t'$ or $t < t'$.

Outline

Inserting minimal sets of control-arcs

In a standard network $\mathcal{P}_S$, we have to add control-arcs that resolve all WPCs:

- Hereby, inserting one control-arc from CA(σ, σ') resolves WPC(σ, σ'), but possibly also another WPC in $\mathcal{P}_S$.
- Let $\mathcal{A}_S$ be the union of all the control-arcs from the sets CA(σ, σ') for all WPC(σ, σ') in $\mathcal{P}_S$.
- To resolve the WPCs in $\mathcal{P}_S$, we have to find subsets A' of $\mathcal{A}_S$ **hitting** each CA(σ, σ').
- Non-minimal hitting sets yield extended Peri nets with unnecessary control-arcs and, thus, being not minimal.

Theorem

All **minimal** extended Petri nets based on $\mathcal{P}_S$ can be obtained by computing all **minimal** hitting sets in $\mathcal{A}_S$.

Inserting minimal sets of control-arcs

In a standard network $\mathcal{P}_S$, we have to add control-arcs that resolve all WPCs:

- Hereby, inserting one control-arc from $\mathrm{CA}(\sigma, \sigma')$ resolves $\mathrm{WPC}(\sigma, \sigma')$, but possibly also another WPC in $\mathcal{P}_S$.
- Let $\mathcal{A}_S$ be the union of all the control-arcs from the sets $\mathrm{CA}(\sigma, \sigma')$ for all $\mathrm{WPC}(\sigma, \sigma')$ in $\mathcal{P}_S$.
- To resolve the WPCs in $\mathcal{P}_S$, we have to find subsets A' of $\mathcal{A}_S$ **hitting** each $\mathrm{CA}(\sigma, \sigma')$.
- Non-minimal hitting sets yield extended Peri nets with unnecessary control-arcs and, thus, being not minimal.

Theorem

All **minimal** extended Petri nets based on $\mathcal{P}_S$ can be obtained by computing all **minimal** hitting sets in $\mathcal{A}_S$.

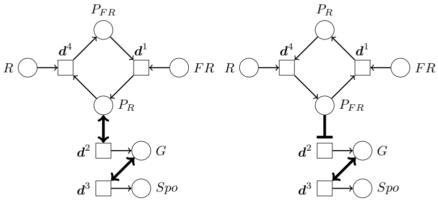
Inserting minimal sets of control-arcs: Example

Example

For the standard network $\mathcal{P}_{S_1}$ with $T_1 = \{\mathbf{d}^1, \mathbf{d}^2, \mathbf{d}^3 = \mathbf{d}^6, \mathbf{d}^4 = \mathbf{d}^5\}$, we have:

	$(P_{FR},\mathbf{d}^2) \in A_I$	$(P_R,\mathbf{d}^2) \in A_R$	$(P_{FR},\mathbf{d}^3) \in A_I$	$(P_R,\mathbf{d}^3) \in A_R$	$(G,\mathbf{d}^3) \in A_R$
WPC1	x	x			
WPC2			x	x	x
WPC3					x

Two minimal hitting sets result in two **minimal** extended Petri nets based on $\mathcal{P}_{S_1}$:



The total number of **minimal** $\mathcal{X}'$-deterministic extended Petri nets is 66.

Outline

Indecomposability of difference vectors

In Step 1 *Representing the observed responses*, the set $\mathrm{Box}(\mathbf{d}^j)$ of update vectors to refine the sequence between $(\mathbf{x}^j, \mathbf{x}^{j+1}) \in \mathcal{X}'$ can be restricted if

- $\mathbf{d}^j$ exactly corresponds to a well-known biochemical reaction (including the correct stoichiometry) or to a well-known mechanism (that a certain trigger is detected by a suitable receptor),
- experiments have shown that subsets of the input components of $\mathbf{d}^j$ do not lead to the observed response,
- $\mathbf{d}^j$ is treated as black box-like reaction where only input and output matter, but not the intermediate mechanism due to the chosen level of detail.

Indecomposability of difference vectors

Exclude the corresponding response $\mathbf{d}^j \in \mathcal{D}$ from decomposition, just define:

$$\mathrm{Box}(\mathbf{d}^j) := \{\mathbf{d}^j\}$$

Indecomposability of difference vectors

In Step 1 *Representing the observed responses*, the set $\text{Box}(\mathbf{d}^j)$ of update vectors to refine the sequence between $(\mathbf{x}^j, \mathbf{x}^{j+1}) \in \mathcal{X}'$ can be restricted if

- $\mathbf{d}^j$ exactly corresponds to a well-known biochemical reaction (including the correct stoichiometry) or to a well-known mechanism (that a certain trigger is detected by a suitable receptor),
- experiments have shown that subsets of the input components of $\mathbf{d}^j$ do not lead to the observed response,
- $\mathbf{d}^j$ is treated as black box-like reaction where only input and output matter, but not the intermediate mechanism due to the chosen level of detail.

Indecomposability of difference vectors

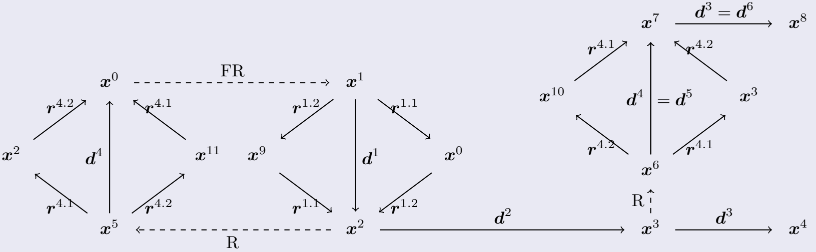
Exclude the corresponding response $\mathbf{d}^j \in \mathcal{D}$ from decomposition, just define:

$$\text{Box}(\mathbf{d}^j) := \{\mathbf{d}^j\}$$

Indecomposability of difference vectors: Example

Example

Since the light-dependent reactions of the photoreceptor are so much faster than the subsequent processes, the difference vectors $\mathbf{d}^1$, $\mathbf{d}^4$ and $\mathbf{d}^5$ describing the photoconversions will not be decomposed due to the chosen level of detail.



Thus, only solutions based on $\mathcal{P}_{S_1}$ with $T_1 = \{\mathbf{d}^1, \mathbf{d}^2, \mathbf{d}^3 = \mathbf{d}^6, \mathbf{d}^4 = \mathbf{d}^5\}$ remain. Accordingly, the total number of solutions reduces from 66 to 2.

Treating equal difference vectors in the same way

If two difference vectors $\mathbf{d}^i, \mathbf{d}^j \in \mathcal{D}$ are equal, we have $\text{Box}(\mathbf{d}^i) = \text{Box}(\mathbf{d}^j)$ and $\Lambda(\mathbf{d}^i) = \Lambda(\mathbf{d}^j)$. In Step 2 *Detecting priority conflicts between sequences*, it is natural to require that $\mathbf{d}^i = \mathbf{d}^j$ are decomposed in the same way: using the same

- $\lambda \in \Lambda(\mathbf{d}^i) = \Lambda(\mathbf{d}^j)$ means that the same subset of molecules involved in a reaction will not interact according to different mechanisms,
- permutation π of the elements in $\mathcal{R}(\mathbf{d}^i, \lambda) = \mathcal{R}(\mathbf{d}^j, \lambda)$ means that the order in which the reactions are applied reflects the relative rates of the reactions in $\mathcal{R}(\mathbf{d}^i, \lambda) = \mathcal{R}(\mathbf{d}^j, \lambda)$, which leads to the same priorities within the resulting sequences.

We call $\sigma_{\pi,\boldsymbol{\lambda}}(\mathbf{x}^i, \mathbf{d}^i)$ and $\sigma'_{\pi,\boldsymbol{\lambda}}(\mathbf{x}^j, \mathbf{d}^j)$ **twin sequences** if $\mathbf{d}^i = \mathbf{d}^j$ and the same $\boldsymbol{\lambda}$, π has been used.

Treating equal difference vectors in the same way

Force that twin sequences are always selected together by adding strong priority conflicts in $\mathcal{G}$ between all other sequences stemming from a pair $\mathbf{d}^i = \mathbf{d}^j \in \mathcal{D}$.

Treating equal difference vectors in the same way

If two difference vectors $\mathbf{d}^i, \mathbf{d}^j \in \mathcal{D}$ are equal, we have $\text{Box}(\mathbf{d}^i) = \text{Box}(\mathbf{d}^j)$ and $\Lambda(\mathbf{d}^i) = \Lambda(\mathbf{d}^j)$. In Step 2 *Detecting priority conflicts between sequences*, it is natural to require that $\mathbf{d}^i = \mathbf{d}^j$ are decomposed in the same way: using the same

- $\lambda \in \Lambda(\mathbf{d}^i) = \Lambda(\mathbf{d}^j)$ means that the same subset of molecules involved in a reaction will not interact according to different mechanisms,
- permutation π of the elements in $\mathcal{R}(\mathbf{d}^i, \lambda) = \mathcal{R}(\mathbf{d}^j, \lambda)$ means that the order in which the reactions are applied reflects the relative rates of the reactions in $\mathcal{R}(\mathbf{d}^i, \lambda) = \mathcal{R}(\mathbf{d}^j, \lambda)$, which leads to the same priorities within the resulting sequences.

We call $\sigma_{\pi,\boldsymbol{\lambda}}(\mathbf{x}^i, \mathbf{d}^i)$ and $\sigma'_{\pi,\boldsymbol{\lambda}}(\mathbf{x}^j, \mathbf{d}^j)$ **twin sequences** if $\mathbf{d}^i = \mathbf{d}^j$ and the same $\boldsymbol{\lambda}$, π has been used.

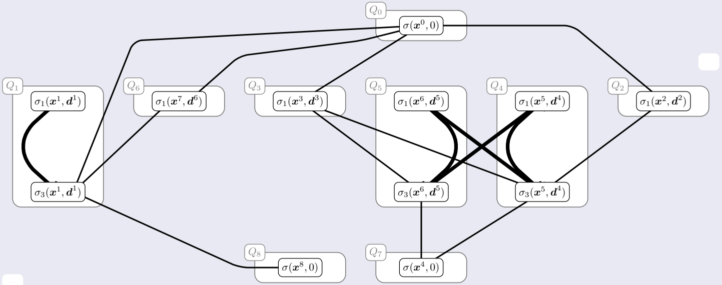
Treating equal difference vectors in the same way

Force that twin sequences are always selected together by adding strong priority conflicts in $\mathcal{G}$ between all other sequences stemming from a pair $\mathbf{d}^i = \mathbf{d}^j \in \mathcal{D}$.

Treating equal difference vectors in the same way: Example

Example

In our running example, we have $\mathbf{d}^3 = \mathbf{d}^6$ and $\mathbf{d}^4 = \mathbf{d}^5$. The latter vectors can be decomposed in different ways, among the resulting sequences we have $\sigma_1(\mathbf{x}^5, \mathbf{d}^4)$, $\sigma_1(\mathbf{x}^6, \mathbf{d}^5)$ and $\sigma_3(\mathbf{x}^5, \mathbf{d}^4)$, $\sigma_3(\mathbf{x}^6, \mathbf{d}^5)$ as pairs of twin sequences.



Forcing to select these pairs together rules out 4 of the previously possible 8 selections. Accordingly, the total number of solutions reduces from 66 to 18.

Integrating knowledge on relative reaction rates

In Step 3 *Constructing $\mathcal{X}'$-deterministic Petri nets*, to resolve a WPC(σ, σ') between σ, σ' involving update vectors $\mathbf{r}^t \neq \mathbf{r}^{t'}$ and intermediate states $\mathbf{y} \neq \mathbf{y}'$, either transition t has to be disabled at $\mathbf{y}'$ or transition t' at $\mathbf{y}$, while the decision between t and t' on the other state can be handled by a priority.

Prior knowledge about the **relative reaction rates** of t and t' can help to decide whether $t > t'$ or $t < t'$ better reflects the reality:

Integrating knowledge on relative reaction rates

- If $\mathbf{y}'$ is a **terminal state** and $\sigma' = \sigma(\mathbf{y}', \mathbf{0})$ its trivial sequence, then $t > \mathbf{0}$ holds automatically at $\mathbf{y}$, but t has to be disabled at $\mathbf{y}'$ using control-arcs from CA$_{t>\mathbf{0}}(\sigma, \sigma')$ whereas CA$_{t<\mathbf{0}}(\sigma, \sigma')$ is empty.
- For a WPC(σ, σ') where the **time-scale** of the corresponding experimental observations clearly differs, deduce the correct priority $t > t'$ or $t < t'$ and reduce CA(σ, σ') accordingly either to CA$_{t>t'}(\sigma, \sigma')$ or to CA$_{t'>t}(\sigma, \sigma')$.

Integrating knowledge on relative reaction rates

In Step 3 *Constructing $\mathcal{X}'$-deterministic Petri nets*, to resolve a WPC(σ, σ') between σ, σ' involving update vectors $\mathbf{r}^t \neq \mathbf{r}^{t'}$ and intermediate states $\mathbf{y} \neq \mathbf{y}'$, either transition t has to be disabled at $\mathbf{y}'$ or transition t' at $\mathbf{y}$, while the decision between t and t' on the other state can be handled by a priority.

Prior knowledge about the **relative reaction rates** of t and t' can help to decide whether $t > t'$ or $t < t'$ better reflects the reality:

Integrating knowledge on relative reaction rates

- If $\mathbf{y}'$ is a **terminal state** and $\sigma' = \sigma(\mathbf{y}', \mathbf{0})$ its trivial sequence, then $t > \mathbf{0}$ holds automatically at $\mathbf{y}$, but t has to be disabled at $\mathbf{y}'$ using control-arcs from $\mathrm{CA}_{t>\mathbf{0}}(\sigma, \sigma')$ whereas $\mathrm{CA}_{t<\mathbf{0}}(\sigma, \sigma')$ is empty.
- For a WPC(σ, σ') where the **time-scale** of the corresponding experimental observations clearly differs, deduce the correct priority $t > t'$ or $t < t'$ and reduce $\mathrm{CA}(\sigma, \sigma')$ accordingly either to $\mathrm{CA}_{t>t'}(\sigma, \sigma')$ or to $\mathrm{CA}_{t'>t}(\sigma, \sigma')$.

Integrating knowledge on relative reaction rates: Example

Example

In our running example, we have different time-scales for the experimental observations as

- $\mathbf{d}^1$ and $\mathbf{d}^4 = \mathbf{d}^5$ need only milliseconds to occur,
- $\mathbf{d}^2$ needs about 1 hour to occur, and
- $\mathbf{d}^3 = \mathbf{d}^6$ need at least 10 hours

which implies $\mathbf{d}^1, \mathbf{d}^4, \mathbf{d}^5 > \mathbf{d}^2 > \mathbf{d}^3, \mathbf{d}^6$.

Consequently, we can reduce the sets $\mathsf{CA}(\sigma, \sigma')$ of four WPCs as follows:

- for WPC4 between $\sigma_3(\mathbf{x}^1, \mathbf{d}^1)$, $\sigma(\mathbf{x}^7, \mathbf{d}^3)$ to $\mathsf{CA}_{\mathbf{r}^{1.2} > \mathbf{d}^3}(\mathsf{WPC4})$;
- for WPC7 between $\sigma_3(\mathbf{x}^5, \mathbf{d}^4)$, $\sigma(\mathbf{x}^2, \mathbf{d}^2)$ to $\mathsf{CA}_{\mathbf{r}^{4.2} > \mathbf{d}^2}(\mathsf{WPC7})$;
- for WPC8 between $\sigma_3(\mathbf{x}^5, \mathbf{d}^4)$, $\sigma(\mathbf{x}^3, \mathbf{d}^3)$ to $\mathsf{CA}_{\mathbf{r}^{4.2} > \mathbf{d}^3}(\mathsf{WPC8})$;
- for WPC9 between $\sigma_3(\mathbf{x}^6, \mathbf{d}^4)$, $\sigma(\mathbf{x}^3, \mathbf{d}^3)$ to $\mathsf{CA}_{\mathbf{r}^{4.2} > \mathbf{d}^3}(\mathsf{WPC9})$.

Accordingly, the number of solutions decreases from 66 to 36.

Outline

Summary and Conclusions

- ANR aims at reconstructing **all** $\mathcal{X}'$-deterministic extended Petri nets that fit given experimental data $\mathcal{X}'$ which typically results in large solution sets.
- To keep this solution set reasonably small while still guaranteeing its completeness, we firstly generate only **minimal** solutions.
- To integrate **prior knowledge** in the reconstruction procedure to make the "right decisions" in some intermediate steps, we extend the input from $(P, I_P, \mathcal{X}')$ to $(P, I_P, \mathcal{X}', \mathcal{D}_{in}, \mathcal{D}_{eq}, \mathcal{O}_D)$ where
  - $\mathcal{D}_{in}$ contains all indecomposable difference vectors;
  - $\mathcal{D}_{eq}$ contains all pairs of equal difference vectors to be treated in the same way;
  - $\mathcal{O}_D$ contains relative reaction rates between difference vectors;

  which results in substantial reductions of solution alternatives.

Impact of ANR

Models computed by an exact reconstruction approach have predictive ability due to the **completeness** of the solution set guaranteed by mathematical **proofs**.

Summary and Conclusions

- ANR aims at reconstructing **all** $\mathcal{X}'$-deterministic extended Petri nets that fit given experimental data $\mathcal{X}'$ which typically results in large solution sets.
- To keep this solution set reasonably small while still guaranteeing its completeness, we firstly generate only **minimal** solutions.
- To integrate **prior knowledge** in the reconstruction procedure to make the "right decisions" in some intermediate steps, we extend the input from $(P, I_P, \mathcal{X}')$ to $(P, I_P, \mathcal{X}', \mathcal{D}_{in}, \mathcal{D}_{eq}, \mathcal{O}_D)$ where
  - $\mathcal{D}_{in}$ contains all indecomposable difference vectors;
  - $\mathcal{D}_{eq}$ contains all pairs of equal difference vectors to be treated in the same way;
  - $\mathcal{O}_D$ contains relative reaction rates between difference vectors;

  which results in substantial reductions of solution alternatives.

Impact of ANR

Models computed by an exact reconstruction approach have predictive ability due to the **completeness** of the solution set guaranteed by mathematical **proofs**.