Non-interference and opacity enforcement

From observability to privacy and security in discrete event systems

Prof. Gianmaria De Tommasi
Email: detommas@unina.it

December 2020

DIETI
UNIVERSITA' DEGLI STUDI DI NAPOLI FEDERICO II
DIPARTIMENTO DI INGEGNERIA ELETTRICA E DELLE TECNOLOGIE DELL'INFORMAZIONE

Course syllabus

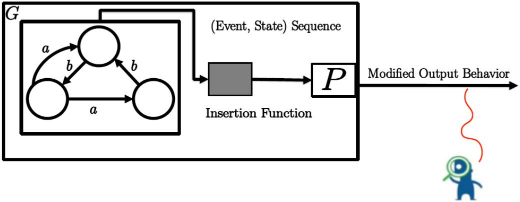
1. Discrete Event Systems (DES), Languages and Automata
2. Petri nets (PNs) and their twofold representation to model DES
3. MILP and ILP formulations: logical conditions, binary variables “do everything", and variable connecting
4. Adding uncertainty: unobservable events and observers for finite state automata and PNs
5. Augmenting the observers: diagnosability of prefix-closed languages, diagnosers and the fault detection for finite state automata
6. Diagnosability and fault detection in PNs - Part I: graph-based approaches
7. Diagnosability and fault detection in PNs - Part II: algebraic approaches for bounded systems
8. Security issues in DES: non-interference and opacity
9. **Non-interference and opacity enforcement**
10. **Open issues**

Outline

- **Enforcement by Supervisory Control** $\rightarrow$ to restrict the system's behaviour in order to preserve the security/privacy property

- **Enforcement by Supervisory Control** $\rightarrow$ to restrict the system's behaviour in order to preserve the security/privacy property
- **Enforcement by insertion/obfuscation** $\rightarrow$ to input or mask observable events of the systems so to output (possibly) modified information to the malicious observers

Enforcement by insertion: an example



Taken from

C. Keroglou and S. Lafortune
Embedded Insertion Functions for Opacity Enforcement,
IEEE Transactions on Automatic Control, 2020

Main assumptions

- The net system $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$ is bounded

SNNI definition for P/T nets

Main assumptions

- The net system $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$ is bounded
- The net system is assumed to be unlabeled
- The P/T net: $N = (P, L, H, \mathbf{Pre}, \mathbf{Post})$, with $L \cap H = \emptyset$
  - L low-level transitions
  - H high-level transitions
  - $T = L \cup H$

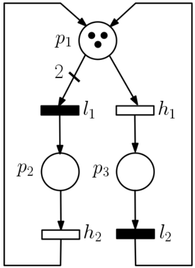
SNNI definition for P/T nets

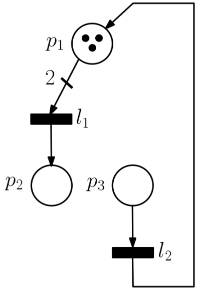
Main assumptions

- The net system $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$ is bounded
- The net system is assumed to be unlabeled
- The P/T net: $N = (P, L, H, \mathbf{Pre}, \mathbf{Post})$, with $L \cap H = \emptyset$
  - L low-level transitions
  - H high-level transitions
  - $T = L \cup H$

Objective: to exploit the twofold representation of PN systems to find algebraic conditions to assess SNNI

- The *low-level system* is the system *induced* by the low-level transitions
- $L = \{l_1, l_2\}$ and $H = \{h_1, h_2\}$





- Let $\mathcal{S} = \langle N\,, \boldsymbol{m}_0 \rangle$ be a net system and $\mathcal{S}_L = \langle N_L\,, \boldsymbol{m}_0 \rangle$ the correspondent low-level system

SNNI net system

- Let $\mathcal{S} = \langle N \, , \boldsymbol{m}_0 \rangle$ be a net system and $\mathcal{S}_L = \langle N_L \, , \boldsymbol{m}_0 \rangle$ the correspondent low-level system
- $\mathcal{S}$ is SNNI *if and only if*

$$Pr_L \left(\mathcal{L}(N \, , \boldsymbol{m}_0) \right) \; = \; \mathcal{L}(N_L \, , \boldsymbol{m}_0)$$

- The system is assumed **bounded**

Exploiting the boundedness assumption

- The system is assumed **bounded**
- Therefore it is possible to describe the state space by means of a **set of linear constraints** (as we did for diagnosability)

- The system is assumed **bounded**
- Therefore it is possible to describe the state space by means of a **set of linear constraints** (as we did for diagnosability)
- There exists a set of ρ integer vectors $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_\rho$ with $\rho \leq |\sigma|$ such that the following linear constraints are fulfilled

$$\begin{cases} \boldsymbol{m} \geq \mathbf{Pre} \cdot \boldsymbol{s}_1 \\ \boldsymbol{m} + \boldsymbol{C} \cdot \boldsymbol{s}_1 \geq \mathbf{Pre} \cdot \boldsymbol{s}_2 \\ \ldots \\ \boldsymbol{m} + \boldsymbol{C} \cdot \sum_{i=1}^{\rho-1} \boldsymbol{s}_i \geq \mathbf{Pre} \cdot \boldsymbol{s}_\rho \\ \sum_{i=1}^{\rho} \boldsymbol{s}_i = \pi(\sigma) \end{cases} \tag{1}$$

iff there exists at least one sequence σ, which is enabled under the marking $\boldsymbol{m}$ and such that $\pi(\sigma) = \boldsymbol{\sigma}$

- The system is assumed **bounded**
- Therefore it is possible to describe the state space by means of a **set of linear constraints** (as we did for diagnosability)
- There exists a set of ρ integer vectors $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_\rho$ with $\rho \leq |\sigma|$ such that the following linear constraints are fulfilled

$$\begin{cases} \boldsymbol{m} \geq \mathbf{Pre} \cdot \boldsymbol{s}_1 \\ \boldsymbol{m} + \boldsymbol{C} \cdot \boldsymbol{s}_1 \geq \mathbf{Pre} \cdot \boldsymbol{s}_2 \\ \ldots \\ \boldsymbol{m} + \boldsymbol{C} \cdot \sum_{i=1}^{\rho-1} \boldsymbol{s}_i \geq \mathbf{Pre} \cdot \boldsymbol{s}_\rho \\ \sum_{i=1}^{\rho} \boldsymbol{s}_i = \pi(\sigma) \end{cases} \tag{1}$$

  iff there exists at least one sequence σ, which is enabled under the marking $\boldsymbol{m}$ and such that $\pi(\sigma) = \boldsymbol{\sigma}$
- **For a bounded net, given a sufficiently large number of inequality constraints (1), it is possible to describe the $R(N, \boldsymbol{m}_0)$ set**

- The system is assumed **bounded**
- Therefore it is possible to describe the state space by means of a **set of linear constraints** (as we did for diagnosability)
- There exists a set of ρ integer vectors $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_\rho$ with $\rho \leq |\sigma|$ such that the following linear constraints are fulfilled

$$\begin{cases} \boldsymbol{m} \geq \mathbf{Pre} \cdot \boldsymbol{s}_1 \\ \boldsymbol{m} + \boldsymbol{C} \cdot \boldsymbol{s}_1 \geq \mathbf{Pre} \cdot \boldsymbol{s}_2 \\ \ldots \\ \boldsymbol{m} + \boldsymbol{C} \cdot \sum_{i=1}^{\rho-1} \boldsymbol{s}_i \geq \mathbf{Pre} \cdot \boldsymbol{s}_\rho \\ \sum_{i=1}^{\rho} \boldsymbol{s}_i = \pi(\sigma) \end{cases} \tag{1}$$

  iff there exists at least one sequence σ, which is enabled under the marking $\boldsymbol{m}$ and such that $\pi(\sigma) = \boldsymbol{\sigma}$
- **For a bounded net, given a sufficiently large number of inequality constraints** (1)**, it is possible to describe the** $R(N, \boldsymbol{m}_0)$ **set** $\rightarrow$ **let assume that** J **inequalities are sufficient to this purpose**

The key idea exploited in the algebraic approach

- For bounded net, given J there exists a maximum number of time a transition can fire given the constraints (1)

The key idea exploited in the algebraic approach

- For bounded net, given J there exists a maximum number of time a transition can fire given the constraints (1)
- Let us denote as φ_t the maximum number of firings of a **low-level transition** t in the low-level system $\mathcal{S}_L$

The key idea exploited in the algebraic approach

- For bounded net, given J there exists a maximum number of time a transition can fire given the constraints (1)
- Let us denote as φ_t the maximum number of firings of a **low-level transition** t in the low-level system $\mathcal{S}_L$
- If it is possible to have at least one additional firing of t in the original net system, this implies interference

The key idea exploited in the algebraic approach

- For bounded net, given J there exists a maximum number of time a transition can fire given the constraints (1)
- Let us denote as φ_t the maximum number of firings of a **low-level transition** t in the low-level system $\mathcal{S}_L$
- If it is possible to have at least one additional firing of t in the original net system, this implies interference
- The other source of interference is the possibility of using high-level transitions to enable the firing of t

Maximum number of firing of a low-level transition in $\mathcal{S}_L$

Given J constraints in (1), the maximum number of firings for $t \in L$ in $\mathcal{S}_L$ can be computed as the solution of the ILP

$$\varphi_t = \max \sum_{i=1}^{J} \boldsymbol{\sigma}_i(t)$$

subject to

$$\begin{cases} \boldsymbol{m}_0 \geq \mathbf{Pre}_L \cdot \boldsymbol{\sigma}_1 \\ \boldsymbol{m}_0 + \boldsymbol{C}_L \cdot \boldsymbol{\sigma}_1 \geq \mathbf{Pre}_L \cdot \boldsymbol{\sigma}_2 \\ \dots \\ \boldsymbol{m}_0 + \boldsymbol{C}_L \cdot \sum_{i=1}^{J-1} \boldsymbol{\sigma}_i \geq \mathbf{Pre}_L \cdot \boldsymbol{\sigma}_J \\ \boldsymbol{m}_0 + \boldsymbol{C}_L \cdot \sum_{i=1}^{J} \boldsymbol{\sigma}_i \geq \mathbf{0} \\ \boldsymbol{\sigma}_i \in \mathbb{N}^n, \quad i = 1, 2, \dots, J \end{cases}$$

Given a K-bounded system $\mathcal{S}$, let consider the two ILP problems

$$\min \sum_{i=1}^{J} \sum_{t_h \in H} \boldsymbol{x}_i(t_h) \tag{2}$$

subject to

$$\mathcal{X}(\boldsymbol{m}_0, \varphi_t): \begin{cases} \boldsymbol{m}_0 \geq \mathbf{Pre} \cdot \boldsymbol{x}_1 \\ \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{x}_1 \geq \mathbf{Pre} \cdot \boldsymbol{x}_2 \\ \cdots & \text{(3a)} \\ \boldsymbol{m}_0 + \boldsymbol{C} \cdot \sum_{i=1}^{J-1} \boldsymbol{x}_i \geq \mathbf{Pre} \cdot \boldsymbol{x}_J \\ \boldsymbol{m}_0 + \boldsymbol{C} \cdot \sum_{i=1}^{J} \boldsymbol{x}_i \geq \mathbf{0} \\ \sum_{i=1}^{J} \boldsymbol{x}_i(t) \geq \varphi_t + 1 & \text{(3b)} \\ \boldsymbol{x}_i \in \mathbb{N}^n, \quad i = 1, 2, \ldots, J & \text{(3c)} \end{cases}$$

$$\min \left[\sum_{i=1}^{J} \sum_{t_l \in L} \boldsymbol{y}_i(t_l) + \epsilon \sum_{i=1}^{J} \sum_{t_h \in H} \boldsymbol{y}_i(t_h) \right] \tag{4}$$

subject to

$$\mathcal{Y}(\boldsymbol{m}_0, \varphi_t): \begin{cases} \boldsymbol{m}_0 \geq \mathbf{Pre} \cdot \boldsymbol{y}_1 \\ \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{y}_1 \geq \mathbf{Pre} \cdot \boldsymbol{y}_2 \\ \cdots & \text{(5a)} \\ \boldsymbol{m}_0 + \boldsymbol{C} \cdot \sum_{i=1}^{J-1} \boldsymbol{y}_i \geq \mathbf{Pre} \cdot \boldsymbol{y}_J \\ \boldsymbol{m}_0 + \boldsymbol{C} \cdot \sum_{i=1}^{J} \boldsymbol{y}_i \geq \mathbf{0} \\ \sum_{i=1}^{J} \boldsymbol{y}_i(t) = \varphi_t & \text{(5b)} \\ \boldsymbol{y}_i \in \mathbb{N}^n, \quad i = 1, 2, \ldots, J & \text{(5c)} \end{cases}$$

with $\epsilon < (K \cdot \mathrm{card}(H) \cdot J)^{-1}$

SNNI assessment in DES modeled as Petri nets (II)

System $\mathcal{S}$ is SNNI **iff** the following two conditions hold for each $t \in L$

1) the ILP problem (2)-(3) does not admit a solution

2) the solution of the ILP problem (4)-(5) $\tilde{\boldsymbol{y}}_1, \ldots, \tilde{\boldsymbol{y}}_J \in \mathbb{N}^n$ is such that $\sum_{i=1}^{J} \sum_{t_h \in H} \tilde{\boldsymbol{y}}_j(t_h) = 0$

The static SNNI enforcement problem

Given a bounded and not SNNI net system $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$, and given a set of *potentially selectable* transitions $\mathcal{PS} \subseteq H$, find a subset $\mathcal{D} \subseteq \mathcal{PS}$ such that if $\mathcal{E} = (L \cup H) \setminus \mathcal{D}$, then

i) the system $\mathcal{S}_{\mathcal{E}} = \langle N_{\mathcal{E}}, \boldsymbol{m}_0 \rangle$ is SNNI

ii) for all $\mathcal{D}' \subseteq \mathcal{PS}$ such that $\mathcal{D}' \subset \mathcal{D}$, if $\mathcal{E}' = (L \cup H) \setminus \mathcal{D}'$, then the system $\mathcal{S}_{\mathcal{E}'} = \langle N_{\mathcal{E}'}, \boldsymbol{m}_0 \rangle$ is not SNNI

The static SNNI enforcement problem

Given a bounded and not SNNI net system $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$, and given a set of *potentially selectable* transitions $\mathcal{PS} \subseteq H$, find a subset $\mathcal{D} \subseteq \mathcal{PS}$ such that if $\mathcal{E} = (L \cup H) \setminus \mathcal{D}$, then

i) the system $\mathcal{S}_{\mathcal{E}} = \langle N_{\mathcal{E}}, \boldsymbol{m}_0 \rangle$ is SNNI

ii) for all $\mathcal{D}' \subseteq \mathcal{PS}$ such that $\mathcal{D}' \subset \mathcal{D}$, if $\mathcal{E}' = (L \cup H) \setminus \mathcal{D}'$, then the system $\mathcal{S}_{\mathcal{E}'} = \langle N_{\mathcal{E}'}, \boldsymbol{m}_0 \rangle$ is not SNNI

- The solution to the static enforcement problem **is not necessarily unique**

Static non-interference enforcement

The static SNNI enforcement problem

Given a bounded and not SNNI net system $\mathcal{S} = \langle N\,, \boldsymbol{m}_0 \rangle$, and given a set of *potentially selectable* transitions $\mathcal{PS} \subseteq H$, find a subset $\mathcal{D} \subseteq \mathcal{PS}$ such that if $\mathcal{E} = (L \cup H) \setminus \mathcal{D}$, then

i) the system $\mathcal{S}_{\mathcal{E}} = \langle N_{\mathcal{E}}\,, \boldsymbol{m}_0 \rangle$ is SNNI

ii) for all $\mathcal{D}' \subseteq \mathcal{PS}$ such that $\mathcal{D}' \subset \mathcal{D}$, if $\mathcal{E}' = (L \cup H) \setminus \mathcal{D}'$, then the system $\mathcal{S}_{\mathcal{E}'} = \langle N_{\mathcal{E}'}\,, \boldsymbol{m}_0 \rangle$ is not SNNI

- The solution to the static enforcement problem **is not necessarily unique**
- There may exist several subsets $\mathcal{D}_1\,, \ldots\,, \mathcal{D}_k \subset \mathcal{PS}$ that satisfy conditions **i)** and **ii)**

Does the static SNNI problem admit a solution?

The static SNNI enforcement problem admits a solution **iff** the algorithm returns `true`

Input: $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle, \mathcal{PS} \subseteq H,$ $J, \mathcal{T},$ and $\varphi_{\bar{t}}$ for all $\bar{t} \in \mathcal{T}$

Output: return `true` if Problem 1 admits a solution, `false` otherwise

1 $\mathcal{E} := (L \cup H) \setminus \mathcal{PS}$; `/* assume that all the transitions in` $\mathcal{PS}$ `are disabled */`

2 **foreach** $\bar{t}_l \in \mathcal{T}$ **do**

3 **solve** the ILP problem $\min \left[\sum_{i=1}^{J} \sum_{t \in L} \boldsymbol{y}_i(t) + \varepsilon \cdot \sum_{i=1}^{J} \sum_{t \in H \setminus \mathcal{PS}} \boldsymbol{y}_i(t) \right]$ subject to the set of constraints $\mathcal{Y}_{\mathcal{E}}(\boldsymbol{m}_0, \varphi_{\bar{t}})$;

4 **let** $\tilde{\boldsymbol{y}}_i$, with $i = 1, \ldots, J$, be the solution of the ILP problem solved at Step 3;

5 **if** $\sum_{i=1}^{J} \sum_{t \in H \setminus \mathcal{PS}} \tilde{\boldsymbol{y}}_i(t) > 0$; `/* interference is due to non selectable high-level transitions */`

6 **then**

7 **return** `false`

8 **end**

9 **solve** the ILP problem $\min \sum_{i=1}^{J} \sum_{t \in H \setminus \mathcal{PS}} \boldsymbol{x}_i(t)$ subject to the set of constraints $\mathcal{X}_{\mathcal{E}}(\boldsymbol{m}_0, \varphi_{\bar{t}})$;

10 **let** $\tilde{\boldsymbol{x}}_i$, with $i = 1, \ldots, J$, be the solution of the ILP problem solved at Step 9;

11 **if** $\sum_{i=1}^{J} \sum_{t \in H \setminus \mathcal{PS}} \tilde{\boldsymbol{x}}_i(t) > 0$; `/* interference is due to non selectable high-level transitions */`

12 **then**

13 **return** `false`

14 **end**

15 **end**

16 **return** `true`

Compute a solution to the static enforcement problem

Input: $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$, $\mathcal{PS} \subseteq H$, J, $\mathcal{T}$, and $\varphi_{\bar{t}}$ for all $\bar{t} \in \mathcal{T}$
Output: a set $\mathcal{D} \subseteq \mathcal{PS}$ that solves Problem 1
Precondition: Algorithm 1 returns `true`

1 $\mathcal{NS} := (L \cup H) \setminus \mathcal{PS}$; `/* set of the transitions that are always enabled */`
2 $\mathcal{CS} := \mathcal{PS}$; `/* initialize the set of selectable transitions */`
3 $\mathcal{D} := \emptyset$; `/* initialize` $\mathcal{D}$ `*/`
4 $\mathcal{E} := \mathcal{NS} \cup \mathcal{CS}$; `/* initialize the set of enabled transitions */`
5 **foreach** $\bar{t}_l \in \mathcal{T}$ **do**
6   **solve** the ILP problem $\min \left[\sum_{i=1}^{J} \sum_{t \in L} \boldsymbol{y}_i(t) + \epsilon \sum_{i=1}^{J} \sum_{t \in \mathcal{CS}} \boldsymbol{y}_i(t) \right]$ subject to the set of constraints $\mathcal{Y}_{\mathcal{E}}(\boldsymbol{m}_0, \varphi_{\bar{t}})$;
7   **let** $\tilde{\boldsymbol{y}}_i$, with $i = 1, \ldots, J$, be the solution of the ILP problem solved at Step 6;
8   **if** $\sum_{i=1}^{J} \sum_{t \in \mathcal{CS}} \tilde{\boldsymbol{y}}_j(t) > 0$ **then**
9   **let** $\tilde{t} \in \mathcal{CS}$ be such that $\sum_{i=1}^{J} \tilde{\boldsymbol{y}}_i(\tilde{t}) > 0$;
        `/* choose one` $\tilde{t} \in \mathcal{CS}$ `that causes interference and update the various sets */`
10         $\mathcal{D} := \mathcal{D} \cup \{\tilde{t}\}$;
11         $\mathcal{CS} := \mathcal{CS} \setminus \{\tilde{t}\}$;
12         $\mathcal{E} := \mathcal{NS} \cup \mathcal{CS}$;
13   **if** $\mathcal{CS} \neq \emptyset$ **then**
14   **go to** Step 6
15   **else**
16   Problem 1 admits the solution $\mathcal{D} = \mathcal{PS}$ and the algorithm **terminates**
17   **end**
18   **end**
19   **solve** the ILP problem $\min \sum_{i=1}^{J} \sum_{t \in \mathcal{CS}} \boldsymbol{x}_i(t)$ subject to the set of constraints $\mathcal{X}_{\mathcal{E}}(\boldsymbol{m}_0, \varphi_{\bar{t}})$;
20   **let** $\tilde{\boldsymbol{x}}_i$, with $i = 1, \ldots, J$, be the solution of the ILP problem solved at Step 19;
21   **if** $\sum_{i=1}^{J} \sum_{t \in \mathcal{CS}} \tilde{\boldsymbol{x}}_i(t) > 0$ **then**
22   **let** $\tilde{t} \in \mathcal{CS}$ be such that $\sum_{i=1}^{J} \tilde{\boldsymbol{x}}_i(\tilde{t}) > 0$;
        `/* choose one` $\tilde{t} \in \mathcal{CS}$ `that causes interference and update the various sets */`
23         $\mathcal{D} := \mathcal{D} \cup \{\tilde{t}\}$;
24         $\mathcal{CS} := \mathcal{CS} \setminus \{\tilde{t}\}$;
25         $\mathcal{E} := \mathcal{NS} \cup \mathcal{CS}$;
26   **if** $\mathcal{CS} \neq \emptyset$ **then**
27   **go to** Step 19
28   **else**
29   Problem 1 admits the solution $\mathcal{D} = \mathcal{PS}$ and the algorithm **terminates**
30   **end**
31   **end**
32 **end**

Compute all the solutions to the static enforcement problem

Algorithm to compute all the solutions to the static SNNI enforcement problem

Input: $\mathcal{S} = \langle N, m_0 \rangle, \mathcal{PS} \subseteq H$, $J, \mathcal{T}$, and $\varphi_{\bar{t}}$ for all $\bar{t} \in \mathcal{T}$

Output: the set $\mathcal{SOL}$ of all the solutions to Problem 1

Precondition: Algorithm 1 returns `true`

$\mathcal{SOL} := \emptyset$; `/* initialize the output */`

run Algorithm 2 and let $\bar{\mathcal{D}}$ be the output;

add $\bar{\mathcal{D}}$ to $\mathcal{SOL}$ and flag it as "`new`";

foreach $\mathcal{D} \in \mathcal{SOL}$ *flagged as* "`new`" **do**

    flag $\mathcal{D}$ as "`old`";

    foreach $t \in \mathcal{D}$ **do**

        if *Algorithm 1 returns* `true` *when executed using* $\mathcal{PS} \setminus \{t\}$ *as set of potentially selectable transitions* **then**

            run Algorithm 2 using $\mathcal{PS} \setminus \{t\}$ as set of potentially selectable transitions and let $\mathcal{D}'$ the solution;

            add $\mathcal{D}'$ to $\mathcal{SOL}$ and flag it as "`new`"

        end

    end

end

Set $\mathcal{SOL}$ of the solutions to the static enforcement problem

Given a bounded and not SNNI net system $\mathcal{S} = \langle N, \boldsymbol{m}_0 \rangle$ and a set of potentially selectable transitions $\mathcal{PS} \subseteq H$, let $\mathcal{SOL}$ be the family of solutions to the static SNNI enforcement problem, i.e.

$$\mathcal{SOL} = \{\mathcal{D} \mid \mathcal{D} \subseteq \mathcal{PS} \text{ and } \mathcal{D} \text{ solves the static SNNI enforcement problem}\}$$

and let

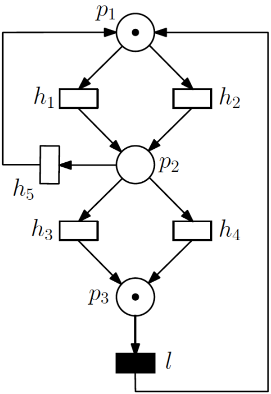
$$\overline{\mathcal{SOL}} = \{\mathcal{E} \subseteq L \cup H \mid \mathcal{E} = (L \cup H) \setminus \mathcal{D}, \text{with } \mathcal{D} \in \mathcal{SOL}\}$$

be the family of transitions in $\mathcal{PS}$ that can be enabled without violating SNNI. If $\mathcal{SOL}$ is not empty, then the following properties hold

a) $\overline{\mathcal{SOL}}$ is not empty

b) $\overline{\mathcal{SOL}}$ is not closed under union

Example



- The static SNNI enforcement problem admits multiple solutions when $\mathcal{PS} = H = \{h_1, h_2, h_3, h_4, h_5\}$
- $\mathcal{SOL} = \{\{h_1, h_2\}, \{h_3, h_4\}\}$
- $\overline{\mathcal{SOL}} = \{\{l, h_3, h_4, h_5\}, \{l, h_1, h_2, h_5\}\}$
- The union of sets in $\overline{\mathcal{SOL}}$ is equal to $L \cup H$ that does not belong to $\overline{\mathcal{SOL}}$

Dynamic SNNI enforcement

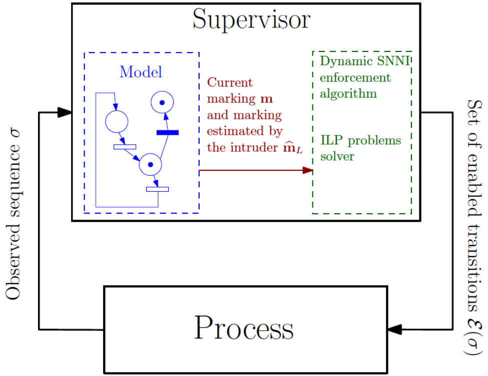
the dynamic SNNI enforcement problem

Given a bounded and not SNNI net system $\mathcal{S} = \langle N , \boldsymbol{m}_0 \rangle$, a set of potentially selectable transitions $\mathcal{PS} \subseteq H$, and a reachable marking $\boldsymbol{m} \in R(N, \boldsymbol{m}_0)$, find a subset $\mathcal{D}(\boldsymbol{m}) \subseteq \mathcal{PS}$ such that if $\mathcal{E}(\boldsymbol{m}) = (L \cup H) \setminus \mathcal{D}(\boldsymbol{m})$, then

i) The system $\mathcal{S}_{\mathcal{E}(\boldsymbol{m})} = \langle N_{\mathcal{E}(\boldsymbol{m})} , \boldsymbol{m} \rangle$ is SNNI

ii) for all $\mathcal{D}' \subseteq \mathcal{PS}$ such that $\mathcal{D}' \subset \mathcal{D}(\boldsymbol{m})$, if $\mathcal{E}' = (L \cup H) \setminus \mathcal{D}'$, then the system $\mathcal{S}_{\mathcal{E}'} = \langle N_{\mathcal{E}'} , \boldsymbol{m} \rangle$ is not SNNI

Control architecture for dynamic SNNI enforcement



When the firing of the k-th transition t is observed, then the supervisor updates the two state vectors estimates

$$\boldsymbol{m}(k) = \boldsymbol{m}(k-1) + \boldsymbol{C}(\cdot, t)\,,$$

$$\begin{cases} \widehat{\boldsymbol{m}}_L(k) = \widehat{\boldsymbol{m}}_L(k-1) + \boldsymbol{C}(\cdot, t)\,, & \text{if } t \in L\,, \\ \widehat{\boldsymbol{m}}_L(k) = \widehat{\boldsymbol{m}}_L(k-1)\,, & \text{otherwise} \end{cases}$$

System marking estimates

When the firing of the k-th transition t is observed, then the supervisor updates the two state vectors estimates

$$\boldsymbol{m}(k) = \boldsymbol{m}(k-1) + \boldsymbol{C}(\cdot, t)\,,$$

$$\begin{cases} \widehat{\boldsymbol{m}}_L(k) = \widehat{\boldsymbol{m}}_L(k-1) + \boldsymbol{C}(\cdot, t)\,, & \text{if } t \in L\,, \\ \widehat{\boldsymbol{m}}_L(k) = \widehat{\boldsymbol{m}}_L(k-1)\,, & \text{otherwise} \end{cases}$$

Initial values

- The two marking estimates are initially set equal to the initial marking, i.e. $\boldsymbol{m}(0) = \widehat{\boldsymbol{m}}(0) = \boldsymbol{m}_0$

Initial values

- The two marking estimates are initially set equal to the initial marking, i.e. $\boldsymbol{m}(0) = \widehat{\boldsymbol{m}}(0) = \boldsymbol{m}_0$
- When the observed sequence is equal to ε, the supervisor disables all the high-level transitions that belong to at least one of the solutions to the static enforcement problem, i.e. the initial guess for the set of high-level transitions to be disabled is

$$\mathcal{D}(\boldsymbol{m}_{-1}) = \bigcup_{\bar{\mathcal{D}} \in \mathcal{SOL}} \bar{\mathcal{D}}\,,$$

  hence, the initial guess for the enabled transitions is $\mathcal{E}(\boldsymbol{m}_{-1}) = (L \cup H) \setminus \mathcal{D}(\boldsymbol{m}_{-1})$

Initial values

- The two marking estimates are initially set equal to the initial marking, i.e. $\boldsymbol{m}(0) = \widehat{\boldsymbol{m}}(0) = \boldsymbol{m}_0$
- When the observed sequence is equal to ε, the supervisor disables all the high-level transitions that belong to at least one of the solutions to the static enforcement problem, i.e. the initial guess for the set of high-level transitions to be disabled is

$$\mathcal{D}(\boldsymbol{m}_{-1}) = \bigcup_{\bar{\mathcal{D}} \in \mathcal{SOL}} \bar{\mathcal{D}},$$

  hence, the initial guess for the enabled transitions is $\mathcal{E}(\boldsymbol{m}_{-1}) = (L \cup H) \setminus \mathcal{D}(\boldsymbol{m}_{-1})$
- $\mathcal{D}(\boldsymbol{m}_{-1})$ and $\mathcal{E}(\boldsymbol{m}_{-1})$ are used to compute $\mathcal{D}(\boldsymbol{m}_0)$ and $\mathcal{E}(\boldsymbol{m}_0) = (L \cup H) \setminus \mathcal{D}(\boldsymbol{m}_0)$

Algorithm for dynamic SNNI enforcement

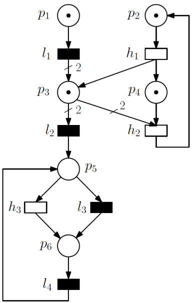
Input: $N\,, \boldsymbol{m}(k)\,, \widehat{\boldsymbol{m}}_L(k)\,, J\,, \mathcal{T}$
Input/Output: $\mathcal{D}(\cdot)\,, \mathcal{E}(\cdot)$
Precondition: Execute Algorithm 3

foreach $t_l \in \mathcal{T}$ **do**
  let $\bar{s}_{t_l} = \max \sum_{i=1}^{J} \boldsymbol{s}_i(t_l)$
  subject to

$$\begin{cases} \widehat{\boldsymbol{m}}_L(k) \geq \mathbf{Pre}_{\mathcal{E}(\boldsymbol{m}(k-1))} \cdot \boldsymbol{s}_1 \\ \widehat{\boldsymbol{m}}_L(k) + \boldsymbol{C}_{\mathcal{E}(\boldsymbol{m}(k-1))} \cdot \boldsymbol{s}_1 \geq \mathbf{Pre}_{\mathcal{E}(\boldsymbol{m}(k-1))} \cdot \boldsymbol{s}_2 \\ \dots \\ \widehat{\boldsymbol{m}}_L(k) + \boldsymbol{C}_{\mathcal{E}(\boldsymbol{m}(k-1))} \cdot \sum_{i=1}^{J-1} \boldsymbol{s}_i \geq \mathbf{Pre}_{\mathcal{E}(\boldsymbol{m}(k-1))} \\ \widehat{\boldsymbol{m}}_L(k) + \boldsymbol{C}_{\mathcal{E}(\boldsymbol{m}(k-1))} \cdot \sum_{i=1}^{J} \boldsymbol{s}_i \geq \mathbf{0} \\ \boldsymbol{s}_i \in \mathbb{N}^{\mathrm{card}(\mathcal{E})}\,, \quad i = 1\,, 2\,, \dots\,, J \end{cases}$$

end
$\mathcal{DD} := \mathcal{D}(\boldsymbol{m}(k-1))$;
$\mathcal{EE} := \mathcal{E}(\boldsymbol{m}(k-1))$;
foreach $t_h \in \mathcal{D}(\boldsymbol{m}(k-1))$ *such that* $\boldsymbol{m}(k) \geq \mathbf{Pre}(\cdot\,, t_h)$ **do**
  $\mathcal{E}' := \mathcal{EE} \cup \{t_h\}$;
  `keep_disbled:=false`;
  foreach $t_l \in \mathcal{T}$ **do**
    if *the ILP problem* $\min \sum_{i=1}^{J} \boldsymbol{x}_i(t_h)$ *subject to the set of constraints* $\mathcal{X}_{\mathcal{E}'}\left(\boldsymbol{m}(k)\,, \bar{s}_{t_l}\right)$ *admits a solution* **then**
      `keep_disabled:=true`;
      break
    end
    if *the ILP problem* $\min \left[\sum_{i=1}^{J} \sum_{t \in L} \boldsymbol{y}_i(t) + \epsilon \sum_{i=1}^{J} \boldsymbol{y}_i(t_h)\right]$ *subject to the set of constraints* $\mathcal{Y}_{\mathcal{E}'}\left(\widehat{\boldsymbol{m}}_L(k)\,, \bar{s}_{t_l}\right)$ *admits a solution* $\boldsymbol{y}'_i\,, i = 1\,, \dots\,, J$ *such that* $\sum_{i=1}^{J} \boldsymbol{y}'_i(t_h) > 0$
    and *the ILP problem* $\min \left[\sum_{i=1}^{J} \sum_{t \in L} \boldsymbol{y}_i(t) + \epsilon \sum_{i=1}^{J} \boldsymbol{y}_i(t_h)\right]$ *subject to the set of constraints* $\mathcal{Y}_{\mathcal{E}'}\left(\boldsymbol{m}(k)\,, \bar{s}_{t_l}\right)$ *admits a solution* $\boldsymbol{y}''_i\,, i = 1\,, \dots\,, J$ *such that* $\sum_{i=1}^{J} \boldsymbol{y}''_i(t_h) > 0$ **then**
      `keep_disabled:=true`;
      break
    end
  end
  if `keep_disabled=false` **then**
    $\mathcal{DD} := \mathcal{DD} \setminus \{t_h\}$;
    $\mathcal{EE} := \mathcal{EE} \cup \{t_h\}$
  end
end
$\mathcal{D}(\boldsymbol{m}(k)) = \mathcal{DD}$;
$\mathcal{E}(\boldsymbol{m}(k)) = \mathcal{EE}$;

Example



Observed sequence	$\boldsymbol{m}$	$\widehat{\boldsymbol{m}}_L$	$\bar{s}_{l_2}$	$\bar{s}_{l_3}$	$\bar{s}_{l_4}$	Comment	$\mathcal{D}(\cdot)$
ε	$(1\ 1\ 1\ 1\ 0\ 0)^T$	$(1\ 1\ 1\ 1\ 0\ 0)^T$	1	3	3	Under the initial marking h_1 is the only transition in $\mathcal{D}(\cdot)$ that could be potentially enabled. However, for h_1 the ILP problem (14) admits a solution when l_2 is considered. Hence, it must be kept in $\mathcal{D}(\cdot)$.	$\{h_1, h_3\}$
l_1	$(0\ 1\ 3\ 1\ 0\ 0)^T$	$(0\ 1\ 3\ 1\ 0\ 0)^T$	1	4	3	h_1 is still the only transition in $\mathcal{D}(\cdot)$ that could be enabled, but also in this case the ILP problem (14) admits a solution when l_2 is considered.	$\{h_1, h_3\}$
$l_1 h_2$	$(0\ 2\ 1\ 0\ 0\ 0)^T$	$(0\ 1\ 3\ 1\ 0\ 0)^T$	1	4	3	In this case h_1 can be moved to $\mathcal{E}(\cdot)$.	$\{h_3\}$
$l_1 h_2 h_1$	$(0\ 1\ 2\ 1\ 0\ 0)^T$	$(0\ 1\ 3\ 1\ 0\ 0)^T$	1	4	3	h_3 is the only transition in $\mathcal{D}(\cdot)$, but is not enabled under $\boldsymbol{m}$, therefore Algorithm 4 does not need to be executed.	$\{h_3\}$
$l_1 h_2 h_1 h_1$	$(0\ 0\ 3\ 2\ 0\ 0)^T$	$(0\ 1\ 3\ 1\ 0\ 0)^T$	1	4	3	h_3 is still not enabled under $\boldsymbol{m}$.	$\{h_3\}$
$l_1 h_2 h_1 h_1 l_2$	$(0\ 0\ 1\ 2\ 1\ 0)^T$	$(0\ 1\ 1\ 1\ 1\ 0)^T$	0	4	4	h_3 could now be potentially enabled, but when the low-level transition l_4 is considered, both ILP problems (15) and (16) admit a solution $\tilde{\boldsymbol{y}}_i$, $i = 1, \ldots, J$, such that $\sum_{i=1}^{J} \tilde{\boldsymbol{y}}_i(h_3) > 0$. Therefore, h_3 must be kept disabled.	$\{h_3\}$

- Fault prognosability in Labeled Petri nets

Some open issues

- Fault prognosability in Labeled Petri nets
- Tackle the multilevel (multi-domain) non-interference problem with PN systems exploiting the algebraic approach

Some open issues

- Fault prognosability in Labeled Petri nets
- Tackle the multilevel (multi-domain) non-interference problem with PN systems exploiting the algebraic approach
- Verification of state-based opacity for bounded Petri nets using an algebraic approach

Some open issues

- Fault prognosability in Labeled Petri nets
- Tackle the multilevel (multi-domain) non-interference problem with PN systems exploiting the algebraic approach
- Verification of state-based opacity for bounded Petri nets using an algebraic approach
  - for unbounded it may be only graph-based approaches are possible

Some open issues

- Fault prognosability in Labeled Petri nets
- Tackle the multilevel (multi-domain) non-interference problem with PN systems exploiting the algebraic approach
- Verification of state-based opacity for bounded Petri nets using an algebraic approach
  - for unbounded it may be only graph-based approaches are possible
- Extend security and privacy analysis frameworks to the stochastic case

Some open issues

- Fault prognosability in Labeled Petri nets
- Tackle the multilevel (multi-domain) non-interference problem with PN systems exploiting the algebraic approach
- Verification of state-based opacity for bounded Petri nets using an algebraic approach
  - for unbounded it may be only graph-based approaches are possible
- Extend security and privacy analysis frameworks to the stochastic case
- Supervisory control approach of networked system in the case of *man-in-the-middle* attacks

- Prognosability can be used to determine *a priori* if any fault occurrence in the system can be correctly predicted

- Prognosability can be used to determine *a priori* if any fault occurrence in the system can be correctly predicted
- It requires that any fault sequence must have a non-fault prefix for which we know for sure that a fault is guaranteed to occur within a finite number of steps

Fault prognosability

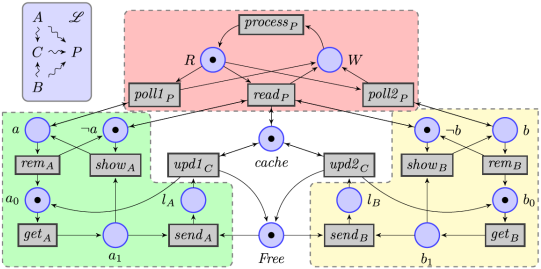
- Prognosability can be used to determine *a priori* if any fault occurrence in the system can be correctly predicted
- It requires that any fault sequence must have a non-fault prefix for which we know for sure that a fault is guaranteed to occur within a finite number of steps
- An alarm can then be issued before the fault occurs

Fault prognosability

- Prognosability can be used to determine *a priori* if any fault occurrence in the system can be correctly predicted
- It requires that any fault sequence must have a non-fault prefix for which we know for sure that a fault is guaranteed to occur within a finite number of steps
- An alarm can then be issued before the fault occurs
- See also

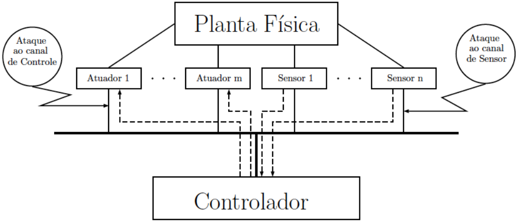
X. Yin
Verification of Prognosability for Labeled Petri Nets,
IEEE Transactions on Automatic Control, 2018

Multilevel non-inerference



Taken from

P. Baldan and A. Beggiato
Multilevel transitive and intransitive non-interference, causally,
Theoretical Computer Science, 2018



Taken from

P. L. Lima *et al.*
Security Against Network Attacks in Supervisory Control Systems,
20th IFAC World Congress, 2017

References

- Non-interference enforcement by means of supervisory control

F. Basile, G. De Tommasi and C. Sterle
Non-interference enforcement via supervisory control in bounded Petri nets,
IEEE Transactions on Automatic Control, to appear 2021

Subjects you should focus on

- Operations on automata (look at the complement automaton). . .
  - . . . operations on the correspondent generated languages
- Build the Observer Automaton for a nondeterministic automaton
- Draw the Reachability & Coverability Graph for simple petri Petri net systems
- Build the Observer Coverability Graph
- Check non-interference on DES modeled as automata

Subjects you should focus on

- Operations on automata (look at the complement automaton). . .
  - . . . operations on the correspondent generated languages
- Build the Observer Automaton for a nondeterministic automaton
- Draw the Reachability & Coverability Graph for simple petri Petri net systems
- Build the Observer Coverability Graph
- Check non-interference on DES modeled as automata

References

- Ch. 2 in Cassandras Lafortune textbook
- Giua Seatzu paper on IEEE TAC 2002
- First part of lecture #8

Subjects you should focus on

- Operations on automata (look at the complement automaton). . .
  - . . . operations on the correspondent generated languages
- Build the Observer Automaton for a nondeterministic automaton
- Draw the Reachability & Coverability Graph for simple petri Petri net systems
- Build the Observer Coverability Graph
- Check non-interference on DES modeled as automata

References

- Ch. 2 in Cassandras Lafortune textbook
- Giua Seatzu paper on IEEE TAC 2002
- First part of lecture #8

The assessment

- When you're ready I will send you 2/3 exercises
- You will have one week to send me back what you have done

Non-interference and opacity enforcement

From observability to privacy and security in discrete event systems

Prof. Gianmaria De Tommasi
Email: detommas@unina.it

December 2020

UNIVERSITA' DEGLI STUDI DI NAPOLI FEDERICO II
DIPARTIMENTO DI INGEGNERIA ELETTRICA E DELLE TECNOLOGIE DELL'INFORMAZIONE