Supervisory Control for Opacity and other analysis problematics (Drafty version)

Hervé Marchand, Philippe Darondeau,
Jérémy Dubreil, Laurie Ricker

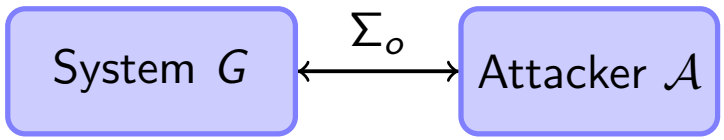
December 2013

Security of Information Systems

- Emergence of new services
  - (E-)voting system
  - Medical card (medical information storage)
  - E-passport (biometric & person. information)
  - Web-services : E-Bancking, on-line payment, travel agency, etc
- Have to deal with critical information that should not flow, be erased or corrupted
- Security Aspects for information systems are classified into 3 categories :
  1. *Integrity* : something illegal cannot be performed by a user
  2. *Confidentiality* : some secret information cannot be inferred by a user
  3. *Availability* : a user can always perform the actions that are allowed by the other security policies

Information flow

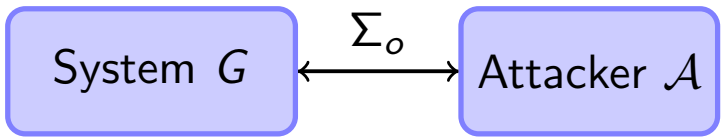
- Context : Given a system $\mathcal{G}$ and an attacker $\mathcal{A}$ who
  - knows the model of $\mathcal{G}$ and
  - observes $\mathcal{G}$ and interacts with $\mathcal{G}$ through Σ_o



- Confidentiality information : secret properties
  - State configurations (values of secret variables)
  - Occurrence of an event (e.g. password file is unlocked), Trajectories
- Information flow : an attacker is able to infer confidential information based on observations and the knowledge of the system.

Information flow

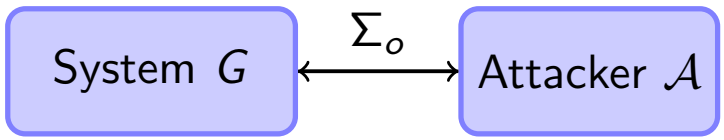
- Context : Given a system $\mathcal{G}$ and an attacker $\mathcal{A}$ who
  - knows the model of $\mathcal{G}$ and
  - observes $\mathcal{G}$ and interacts with $\mathcal{G}$ through Σ_o



- Confidentiality information : secret properties
  - State configurations (values of secret variables)
  - Occurrence of an event (e.g. password file is unlocked), Trajectories
- Information flow : an attacker is able to infer confidential information based on observations and the knowledge of the system.

Information flow

- Context : Given a system $\mathcal{G}$ and an attacker $\mathcal{A}$ who
  - knows the model of $\mathcal{G}$ and
  - observes $\mathcal{G}$ and interacts with $\mathcal{G}$ through Σ_o



- Confidentiality information : secret properties
  - State configurations (values of secret variables)
  - Occurrence of an event (e.g. password file is unlocked), Trajectories
- Information flow : an attacker is able to infer confidential information based on observations and the knowledge of the system.

Outline

Outline

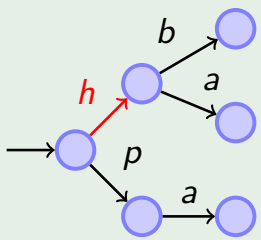
State-Based (language-Based) Opacity

Definition (State-Based opacity)

On $\mathcal{G}$, the secret S is opaque under the projection P_{Σ_o} or $(\mathcal{G}, P_{\Sigma_o})$-opaque if

$$\forall t \in \mathcal{L}_S(\mathcal{G}), \exists s \in \mathcal{L}(\mathcal{G}) : s \approx_{\Sigma_o} t \wedge s \notin \mathcal{L}_S(\mathcal{G}).$$

Example



$$\Sigma = \{h, p, a, b\},\ \Sigma_o = \{a, b\}$$

$$L_\varphi = \Sigma^*.h.\Sigma^*$$

Verification of Opacity

Theorem

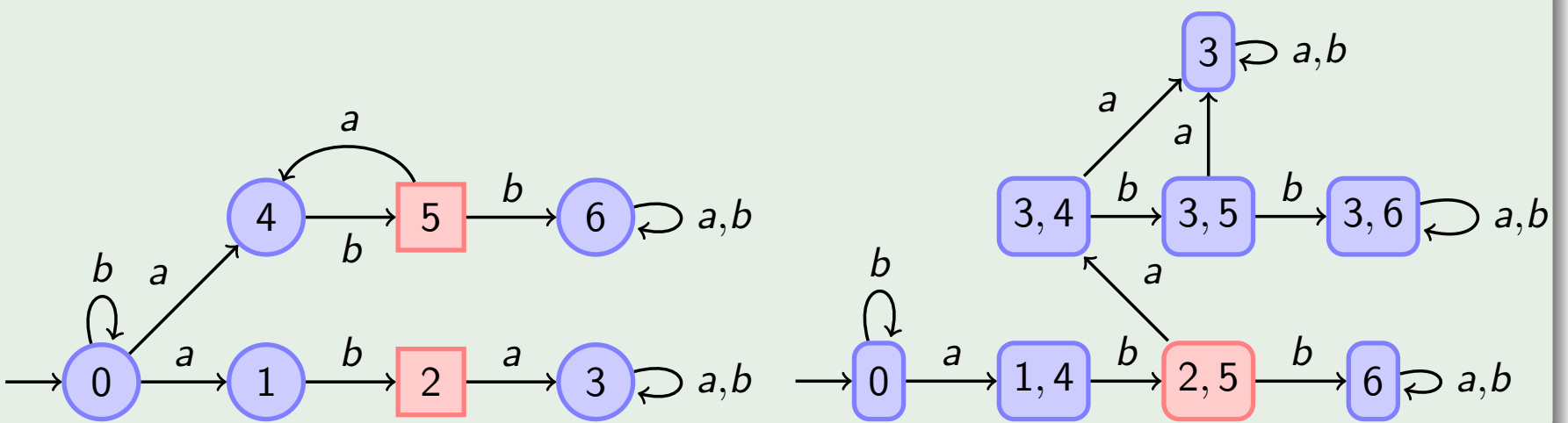
Checking opacity is PSPACE complete

$\Rightarrow$ Reduction of universality problem

Algorithm to Check Opacity

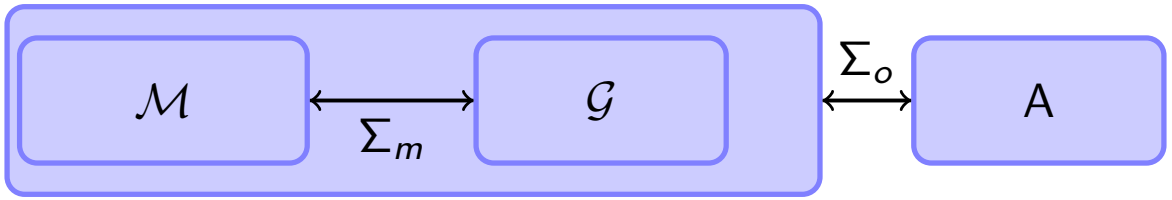
1. Subset construction of $\mathcal{G}$
2. check whether a macro-state $F \subseteq S$ is reachable

Example



Monitoring an information flow

Compute a monitor in charge of detecting an information flow



- Set of observations for which there is an information flow

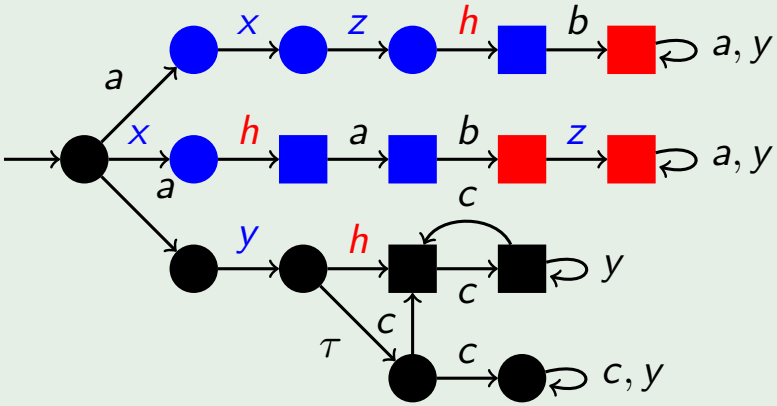
$$\mathcal{F} = \mathcal{L}_{2^S}(Det_{\Sigma_o}(\mathcal{G}))$$

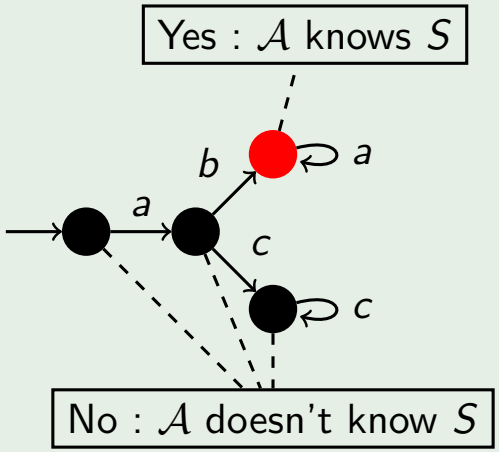
- $\mathcal{M}$ M has to diagnose/predict the occurrence of the trajectories of $\mathcal{I} = \mathcal{L}(\mathcal{G}) \cap P_{\Sigma_o}^{-1}(\mathcal{F}).\Sigma^*$ by observing Σ_m.

- Some information flows might not be detected by the monitor

Monitoring an information flow

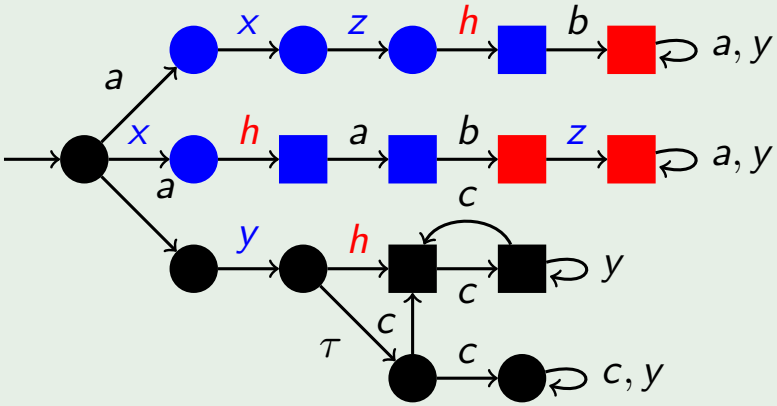
Example ($\Sigma_o = \{a, b, c\}$, $\Sigma_m = \{x, y\}$)

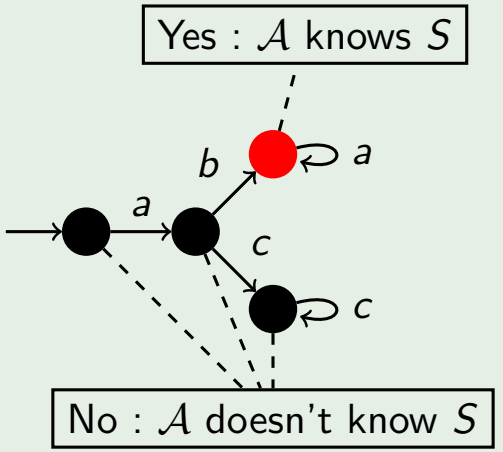


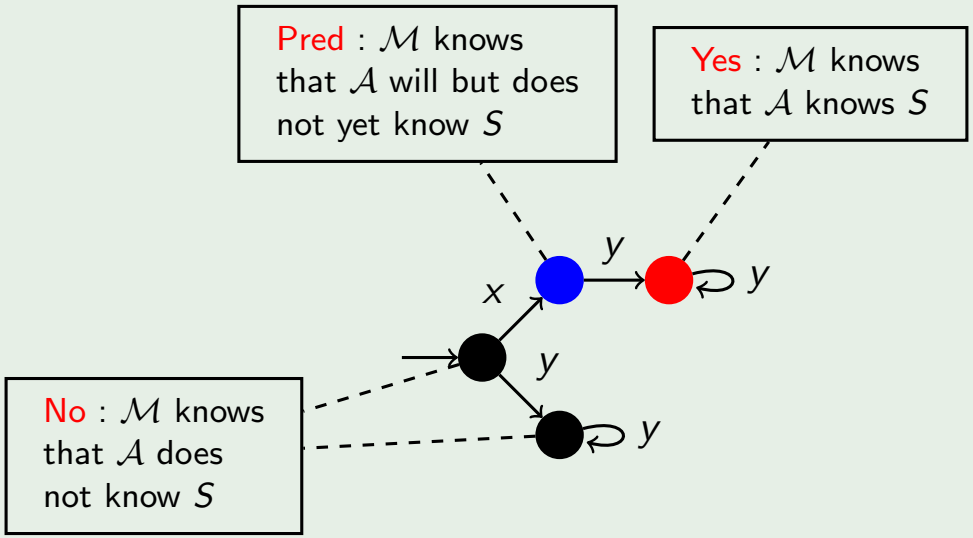


Monitoring an information flow

Example ($\Sigma_o = \{a, b, c\}$, $\Sigma_m = \{x, y\}$)

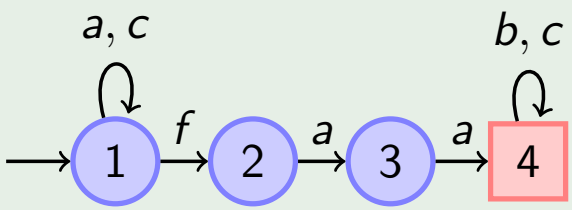




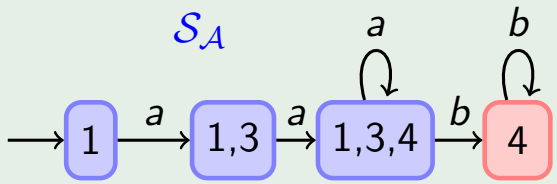


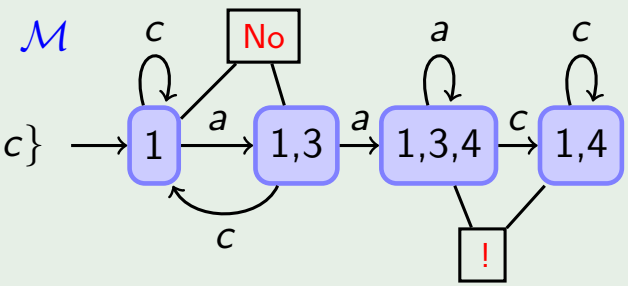
How to be sure that $\mathcal{A}$ knows the secret

Example



$\Sigma_o = \{a, b\}$





$\Sigma_m = \{a, c\}$

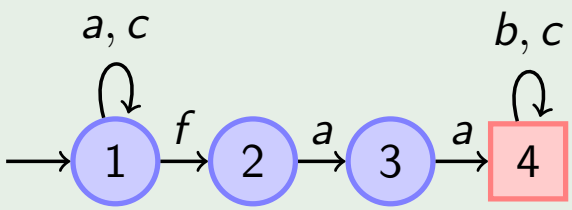
$s = aabc \Rightarrow \begin{cases} P_{\Sigma_o}(s) = aab \\ P_{\Sigma_m}(s) = aac \end{cases}$

Proposition

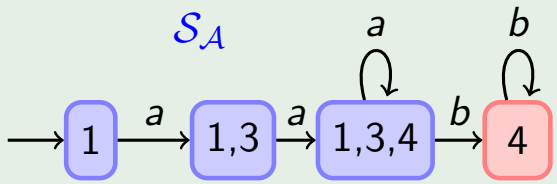
- *Every information flow is detectable iff $\mathcal{G}$ is $\mathcal{I}$-diagnosable w.r.t. Σ_m.*
- *Every information flow is detectable strictly before its occurrence iff $\mathcal{G}$ is $\mathcal{I}$-predictable w.r.t. Σ_m.*

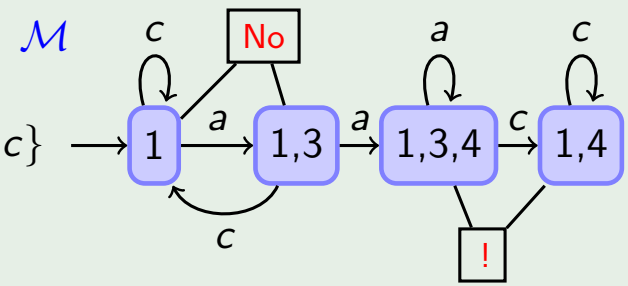
How to be sure that $\mathcal{A}$ knows the secret

Example



$\Sigma_o = \{a, b\}$





$\Sigma_m = \{a, c\}$

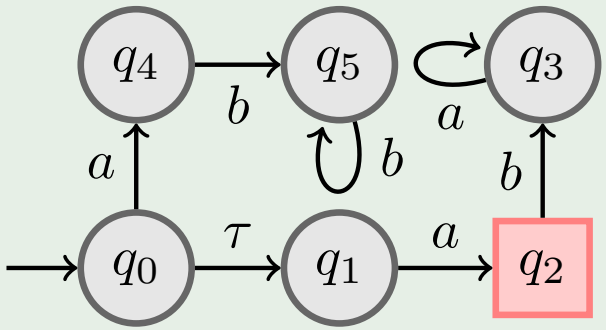
$s = aabc \Rightarrow \begin{cases} P_{\Sigma_o}(s) = aab \\ P_{\Sigma_m}(s) = aac \end{cases}$

Proposition

- *Every information flow is detectable iff $\mathcal{G}$ is $\mathcal{I}$-diagnosable w.r.t. Σ_m.*
- *Every information flow is detectable strictly before its occurrence iff $\mathcal{G}$ is $\mathcal{I}$-predictable w.r.t. Σ_m.*

K-step Weak Opacity (introduced in (Sabori, 2009))

Example



$\Sigma = \{\tau, a, b\}$, $\Sigma_o = \{a, b\}$
observation $a.b.a : \mathcal{G}$ was in the secret two observations ago

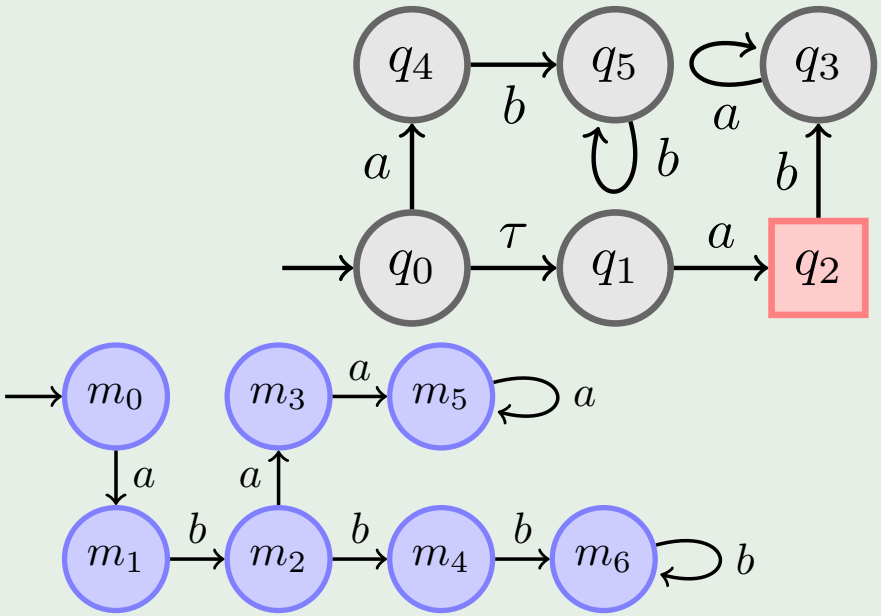
Definition (K-step weak opacity)

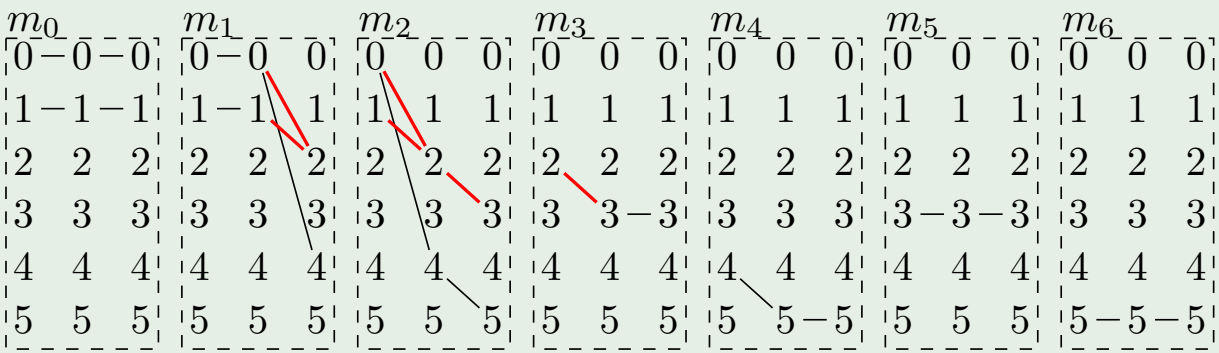
For $K \in \mathbb{N}$, the secret S is K-step weakly opaque on $\mathcal{G}$ under the projection P_{Σ_o} or $(\mathcal{G}, P_{\Sigma_o}, K)$-weakly opaque if

$$\forall t \in \mathcal{L}(\mathcal{G}), \forall t' \preceq t : |t - t'|_{\Sigma_o} \leq K \wedge t' \in \mathcal{L}_S(\mathcal{G})$$
$$\Rightarrow \exists s \in \mathcal{L}(\mathcal{G}), \exists s' \preceq s : s \approx_{\Sigma_o} t \wedge s' \approx_{\Sigma_o} t' \wedge s' \notin \mathcal{L}_S(\mathcal{G}).$$

Verification of K-step weak opacity

Example





Other notions of Opacity

- K-step strong opacity
  $\Rightarrow$ the system traversed the secret less that K observations ago

- infinite step opacity
  $\Rightarrow$ the system traversed the secret in the past

- Initial opacity
  $\Rightarrow$ similar to infinite opacity but the secret is a subset of the initial states.

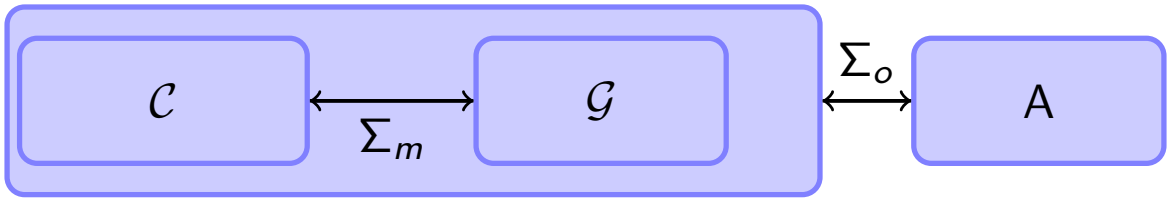
- Initial & Final State-Based opacity

- etc.

Outline

Supervisory Control for opacity

Compute an Access Control using Supervisory Control Theory



Control problem

Hyp : $\mathcal{C}$ observes $\Sigma_m \subseteq \Sigma$ and controls $\Sigma_c \subseteq \Sigma$,
Pb : Compute a maximally permissive access control C observing Σ_m and controlling $\Sigma_c \subseteq \Sigma_m$ s.t. φ is opaque w.r.t. $\mathcal{G} \times \mathcal{C}$ and Σ_o.

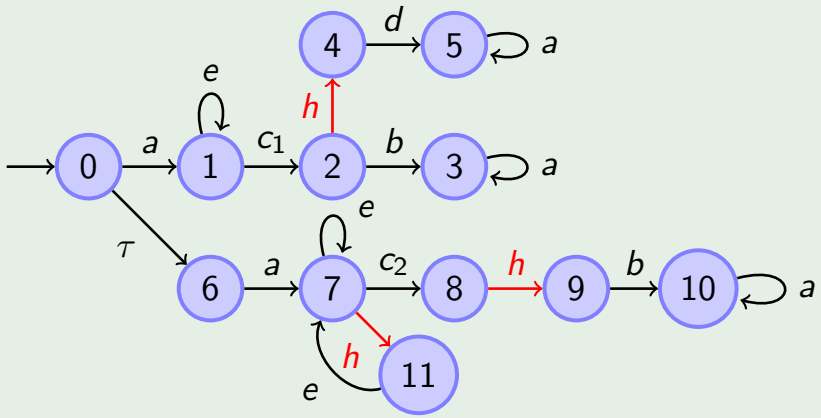
Maximality : $\forall \mathcal{C}'$, $L(\mathcal{G} \times \mathcal{C}') \models \psi \implies L(\mathcal{G} \times \mathcal{C}') \subseteq L(\mathcal{G} \times \mathcal{C})$

Supervisory Control for Opacity

Problem

Compute a maximally permissive access control $\mathcal{C}$ observing Σ_m and controlling $\Sigma_c \subseteq \Sigma_m$ s.t. φ is opaque w.r.t. $\mathcal{G} \times \mathcal{C}$ and Σ_o.

Example



$\Sigma_o = \{a, b, d, e\}$
$\Sigma_m = \{a, c_1, c_2, b, d, e\}$
$\Sigma_c = \{b, c_1, c_2, e\}$

Secret : $L_\varphi = \Sigma^*.h.\Sigma^*$

Theorem

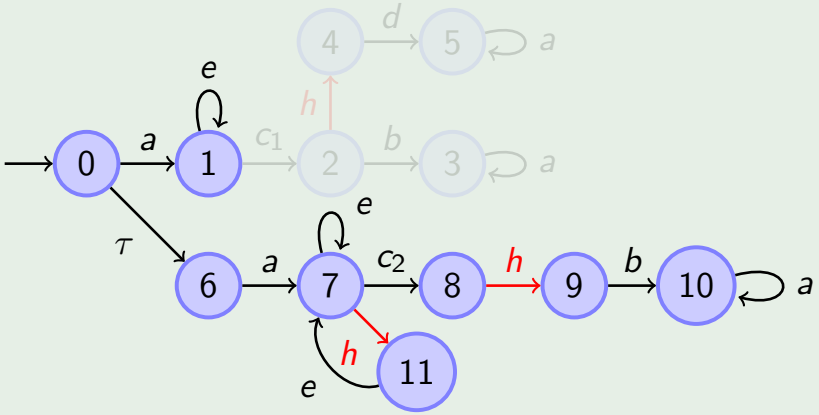
If φ is opaque w.r.t. $\mathcal{L}(\mathcal{G}) \cap \Sigma^*_{uc}$ and Σ_o, and $\Sigma_m \subseteq \Sigma_o$, or $\Sigma_o \subseteq \Sigma_m$, there exists a maximal access control $\mathcal{C}$ s.t. φ is opaque w.r.t. $\mathcal{G} \times \mathcal{C}$ and Σ_o

Supervisory Control for Opacity

Problem

Compute a maximally permissive access control $\mathcal{C}$ observing Σ_m and controlling $\Sigma_c \subseteq \Sigma_m$ s.t. φ is opaque w.r.t. $\mathcal{G} \times \mathcal{C}$ and Σ_o.

Example



$$\Sigma_o = \{a, b, d, e\}$$
$$\Sigma_m = \{a, c_1, c_2, b, d, e\}$$
$$\Sigma_c = \{b, c_1, c_2, e\}$$

Secret : $L_\varphi = \Sigma^*.h.\Sigma^*$

Theorem

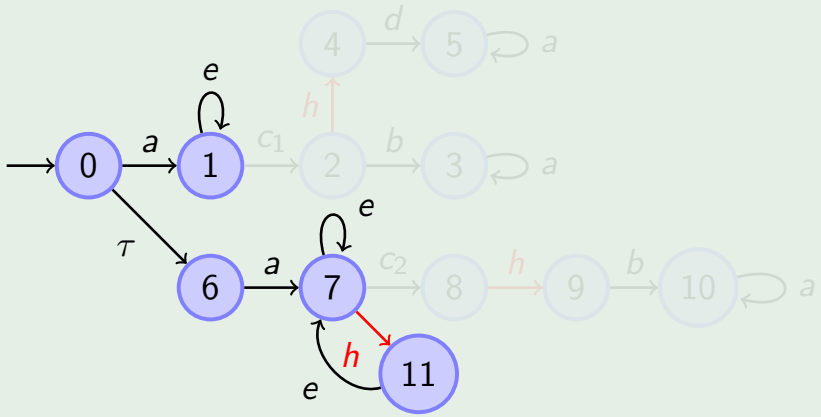
If φ is opaque w.r.t. $\mathcal{L}(\mathcal{G}) \cap \Sigma^*_{uc}$ and Σ_o, and $\Sigma_m \subseteq \Sigma_o$, or $\Sigma_o \subseteq \Sigma_m$, there exists a maximal access control $\mathcal{C}$ s.t. φ is opaque w.r.t. $\mathcal{G} \times \mathcal{C}$ and Σ_o

Supervisory Control for Opacity

Problem

Compute a maximally permissive access control $\mathcal{C}$ observing Σ_m and controlling $\Sigma_c \subseteq \Sigma_m$ s.t. φ is opaque w.r.t. $\mathcal{G} \times \mathcal{C}$ and Σ_o.

Example



$$\Sigma_o = \{a, b, d, e\}$$
$$\Sigma_m = \{a, c_1, c_2, b, d, e\}$$
$$\Sigma_c = \{b, c_1, c_2, e\}$$

Secret : $L_\varphi = \Sigma^*.h.\Sigma^*$

Theorem

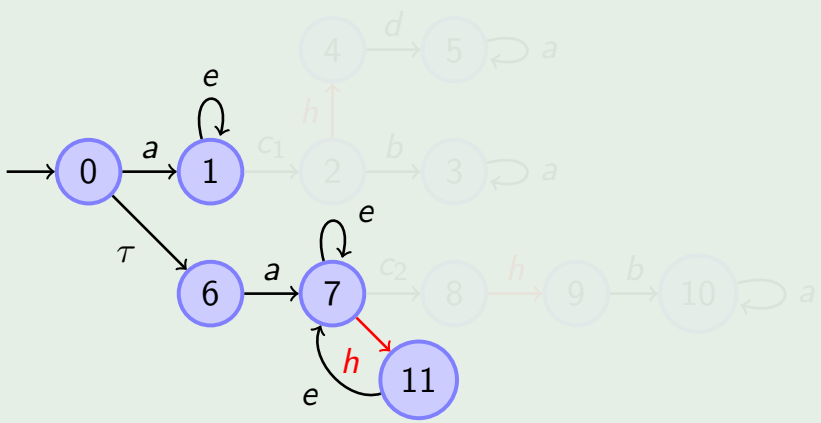
If φ is opaque w.r.t. $\mathcal{L}(\mathcal{G}) \cap \Sigma_{uc}^*$ and Σ_o, and $\Sigma_m \subseteq \Sigma_o$, or $\Sigma_o \subseteq \Sigma_m$, there exists a maximal access control $\mathcal{C}$ s.t. φ is opaque w.r.t. $\mathcal{G} \times \mathcal{C}$ and Σ_o

Supervisory Control for Opacity

Problem

Compute a maximally permissive access control $\mathcal{C}$ observing Σ_m and controlling $\Sigma_c \subseteq \Sigma_m$ s.t. φ is opaque w.r.t. $\mathcal{G} \times \mathcal{C}$ and Σ_o.

Example



$$\Sigma_o = \{a, b, d, e\}$$
$$\Sigma_m = \{a, c_1, c_2, b, d, e\}$$
$$\Sigma_c = \{b, c_1, c_2, e\}$$

Secret : $L_\varphi = \Sigma^*.h.\Sigma^*$

Theorem

If φ is opaque w.r.t. $\mathcal{L}(\mathcal{G}) \cap \Sigma_{uc}^*$ and Σ_o, and $\Sigma_m \subseteq \Sigma_o$, or $\Sigma_o \subseteq \Sigma_m$, there exists a maximal access control $\mathcal{C}$ s.t. φ is opaque w.r.t. $\mathcal{G} \times \mathcal{C}$ and Σ_o

Enforcing Opacity on Modal Transition Systems

Classical Opacity control Problem

1. The controller knows exactly the model of the system
2. The model is exact w.r.t. the implementation

In reality

- The model $\equiv$ an implementation with different versions.
- The implementation should fulfill some requirements with a certain flexibility.

$\Rightarrow$ The model determines the events that the implementation may or must allow after a given run.

- The model : Modal transition systems (MTS) can be used to model this uncertainty with regards to the implementation(s)
- Problem : How extending the Opacity Control problem results to MTS ?

Enforcing Opacity on Modal Transition Systems

Classical Opacity control Problem

1. The controller knows exactly the model of the system
2. The model is exact w.r.t. the implementation

In reality

- The model $\equiv$ an implementation with different versions.
- The implementation should fulfill some requirements with a certain flexibility.

$\Rightarrow$ The model determines the events that the implementation may or must allow after a given run.

- The model : Modal transition systems (MTS) can be used to model this uncertainty with regards to the implementation(s)
- Problem : How extending the Opacity Control problem results to MTS ?

Enforcing Opacity on Modal Transition Systems

Classical Opacity control Problem

1. The controller knows exactly the model of the system
2. The model is exact w.r.t. the implementation

In reality

- The model $\equiv$ an implementation with different versions.
- The implementation should fulfill some requirements with a certain flexibility.

$\Rightarrow$ The model determines the events that the implementation may or must allow after a given run.

- The model : Modal transition systems (MTS) can be used to model this uncertainty with regards to the implementation(s)
- Problem : How extending the Opacity Control problem results to MTS ?

Larsen's modal transition systems

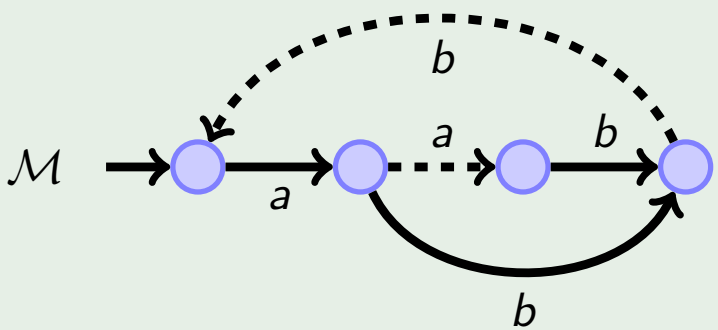
An MTS $\mathcal{M}$ is used to model a non-empty and possibly infinite set of LTS models $(\mathcal{G} \models \mathcal{M})$, each of them being seen as a possible implementation.

Definition

$\mathcal{M} = (S, \Sigma, \delta^{\Box}, \delta^{\Diamond}, s_0)$ where $\delta^{\Box} \subseteq \delta^{\Diamond} \subseteq S \times \Sigma \times S$

- $\delta^{\Diamond}$: May transition (could be present in $\mathcal{G} \models \mathcal{M}$)
- $\delta^{\Box}$: Must transition (must be present in all $\mathcal{G} \models \mathcal{M}$)

Example



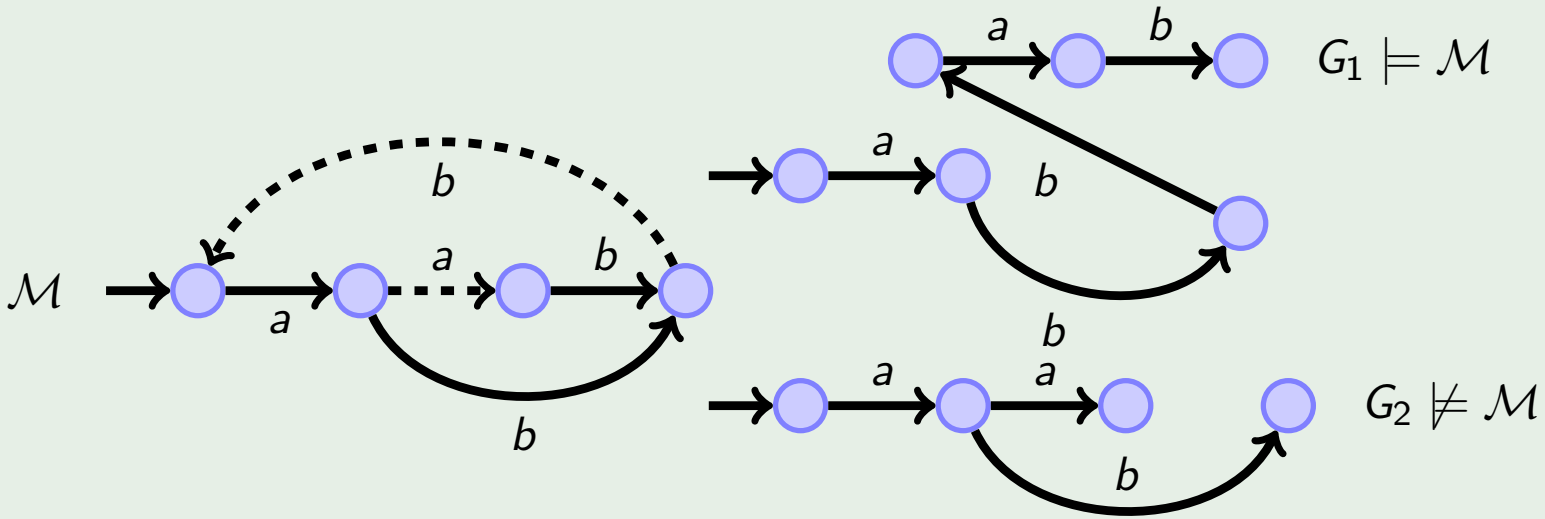
must ($\Box$) = plain edges,
may ($\Diamond$) = dotted edges

LTS associated to an MTS

Each MTS $\mathcal{M}$ determines a corresponding family of finite LTS that preserves the modality conditions and models $\mathcal{M}$ ($\mathcal{G} \models \mathcal{M}$)

$\Rightarrow$ $\models$ is based on an equivalence relation between states of the LTS and the ones of the MTS

Example



Note that G_i might be an unfolding of $\mathcal{M}$

The problem, formally

Given :

- $\mathcal{M} = (S, \Sigma, \delta^{\Box}, \delta^{\Diamond}, s_0)$, a modal transition system
- $Sec \subseteq \Sigma^*$, a regular secret
- Subsets of events :
  - $\Sigma = \Sigma_c \cup \Sigma_{uc}$
  - $\Sigma = \Sigma_m \cup \Sigma_{uo}$ (controller) and $\Sigma_o \subseteq \Sigma$ (attacker)

Construct (regular) controller K s.t.

- K is admissible w.r.t. Σ_c and Σ_m for every $G \models \mathcal{M}$,
- Sec is opaque w.r.t. Σ_o in K/G
- K is maximally permissive with these properties.

$\Rightarrow$ Need to ensure
- Controllability, Observability,
- Opacity

$\Big\}\ \forall G \models MTS$

Some Solutions

$\Sigma_c \subseteq \Sigma_m \subseteq \Sigma_o = \Sigma$

- NS Condition : $\mathcal{L}(K/G) \subseteq \Sigma^* \setminus Sec;\ \forall G \models \mathcal{M}$
- Solution : $K^\dagger$ is the max perm. solution of the SCP for $\overline{\mathcal{M}}$ and the prop. $\mathcal{L}(\overline{\mathcal{M}}) \setminus Sec$ with $\overline{\mathcal{M}} = (S, \Sigma, \delta^\Diamond, s_0)$

$\Sigma_o \subseteq \Sigma_c \subseteq \Sigma_m = \Sigma$

- $Sec = Sec.\Sigma^*$ and is encoded w.l.o.g in $\mathcal{M} = (S, \Sigma, \delta^\Box, \delta^\Diamond, s_0, S_F)$
- $K^\dagger$ computed from H s.t. $X_H \subseteq S \times \mathcal{P}(S)$, $\mathcal{L}(\mathcal{M}) = \mathcal{L}(H)$ and δ_H is build according to Σ_o and the modality of the transitions in $\mathcal{M}$.
  - NS Condition : Sec is opaque w.r.t. Σ_a in every $G \models \mathcal{M}$ if and only if $\forall w : \delta_H((s_0, E_0), w) = (s, E) \ \Rightarrow\ E \not\subseteq S^F$.
  - Solution : $K^\dagger$ is obtained by removing all access (w.r.t. controllabilty) paths to $(s, E),\ E \subseteq S^F$

Some Solutions

$\Sigma_c \subseteq \Sigma_m \subseteq \Sigma_o = \Sigma$

- NS Condition : $\mathcal{L}(K/G) \subseteq \Sigma^* \setminus Sec;\ \forall G \models \mathcal{M}$
- Solution : $K^\dagger$ is the max perm. solution of the SCP for $\overline{\mathcal{M}}$ and the prop. $\mathcal{L}(\overline{\mathcal{M}}) \setminus Sec$ with $\overline{\mathcal{M}} = (S, \Sigma, \delta^\Diamond, s_0)$

$\Sigma_o \subseteq \Sigma_c \subseteq \Sigma_m = \Sigma$

- $Sec = Sec.\Sigma^*$ and is encoded w.l.o.g in $\mathcal{M} = (S, \Sigma, \delta^\Box, \delta^\Diamond, s_0, S_F)$
- $K^\dagger$ computed from H s.t. $X_H \subseteq S \times \mathcal{P}(S)$, $\mathcal{L}(\mathcal{M}) = \mathcal{L}(H)$ and δ_H is build according to Σ_o and the modality of the transitions in $\mathcal{M}$.
  - NS Condition : Sec is opaque w.r.t. Σ_a in every $G \models \mathcal{M}$ if and only if $\forall w : \delta_H((s_0, E_0), w) = (s, E) \ \Rightarrow\ E \not\subseteq S^F$.
  - Solution : $K^\dagger$ is obtained by removing all access (w.r.t. controllabilty) paths to $(s, E),\ E \subseteq S^F$

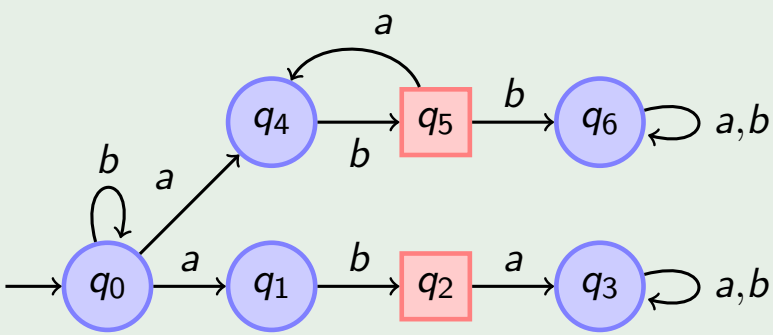
Add a filter between the attacker and the system

Example

- Static Filter : $\Sigma_o = \{a\}$ or $\Sigma_o = \{b\} \Rightarrow F$ is opaque
- Dynamic Filter : Hide "b" after the observation of a "a" and keep everything observable after the observation of a "a"

Add a filter between the attacker and the system

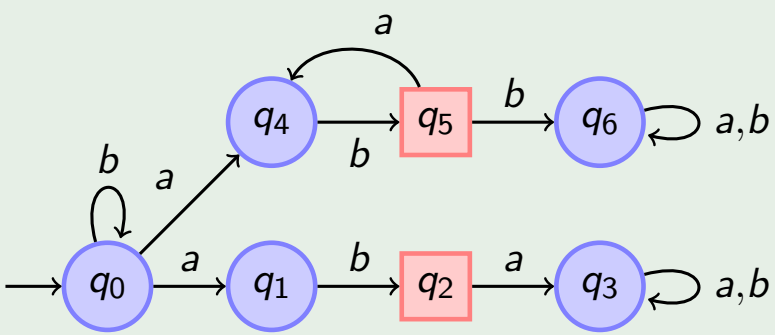
Example



- Static Filter : $\Sigma_o = \{a\}$ or $\Sigma_o = \{b\} \Rightarrow F$ is opaque
- Dynamic Filter : Hide "b" after the observation of a "a" and keep everything observable after the observation of a "a"

Add a filter between the attacker and the system

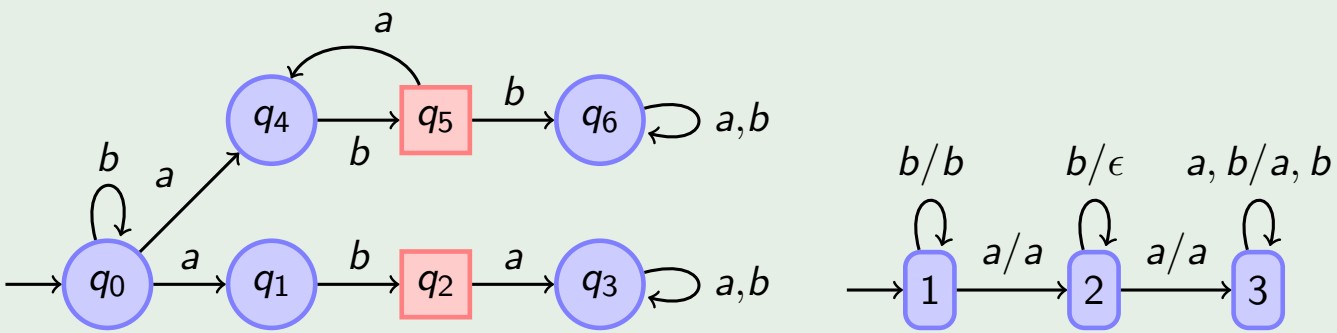
Example



- Static Filter : $\Sigma_o = \{a\}$ or $\Sigma_o = \{b\} \Rightarrow F$ is opaque
- Dynamic Filter : Hide "b" after the observation of a "a" and keep everything observable after the observation of a "a"

Add a filter between the attacker and the system

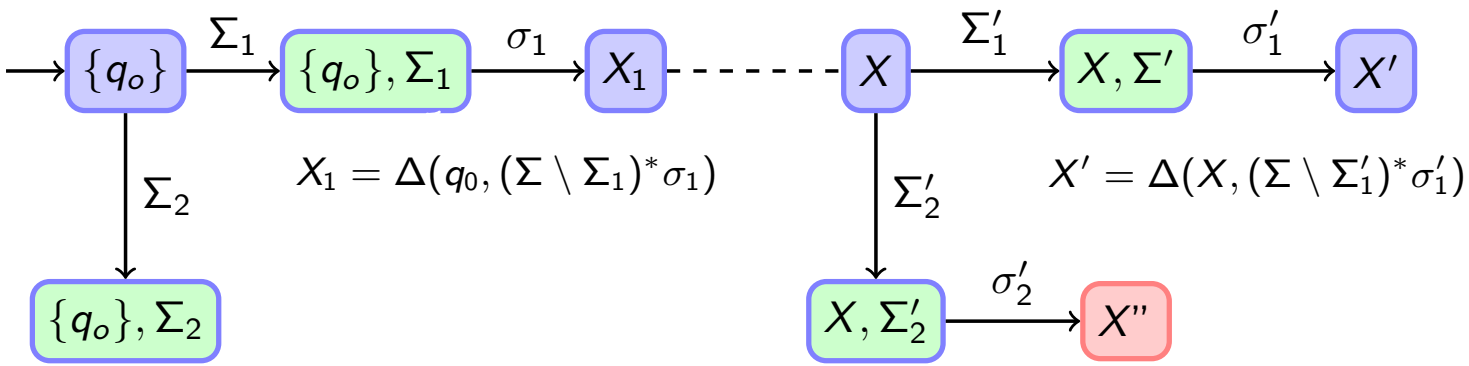
Example



- Static Filter : $\Sigma_o = \{a\}$ or $\Sigma_o = \{b\} \Rightarrow F$ is opaque
- Dynamic Filter : Hide "b" after the observation of a "a" and keep everything observable after the observation of a "a"

Dynamic Filter Synthesis Problem as a Game Problem

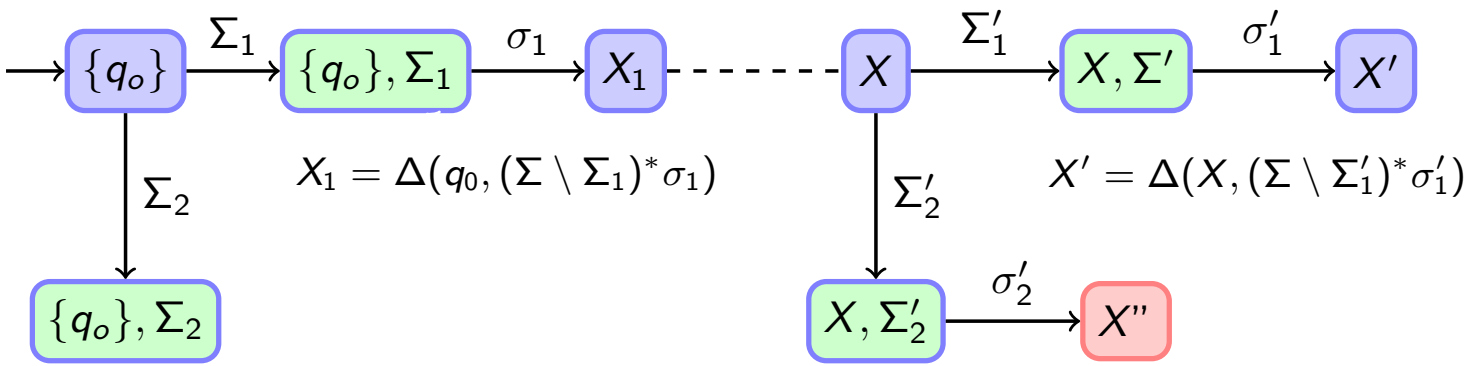
- Reduce the Synthesis Problem to a safety 2-player game
  - Player 1 play the role of a dynamical projection (what remains observable)
    - Player 1 State $\subseteq 2^Q$: estimate of the attacker after an observation
  - Player 2 what the attacker observes.
    - Player 2 State $\subseteq 2^Q\Sigma$: what the attacker could observe in a state of player 1



- Bad states : $Bad = 2^F$
  - Player 2 : find a strategy allowing to reach Bad : *reachability game*
  - Player 1 : find a strategy that avoids Bad : *safety game*

Dynamic Filter Synthesis Problem as a Game Problem

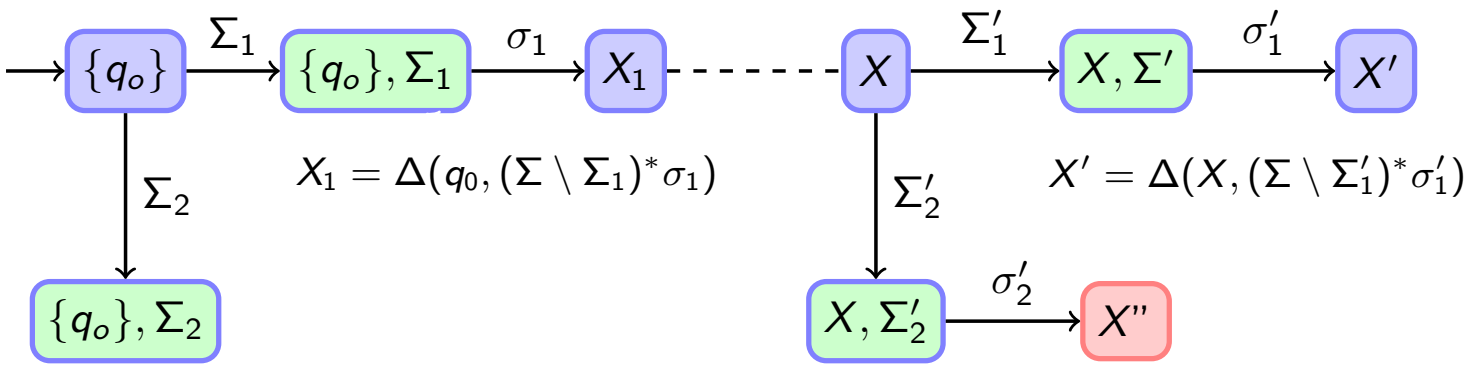
- Reduce the Synthesis Problem to a safety 2-player game
  - Player 1 play the role of a dynamical projection (what remains observable)
    - Player 1 State $\subseteq 2^Q$: estimate of the attacker after an observation
  - Player 2 what the attacker observes.
    - Player 2 State $\subseteq 2^Q\Sigma$: what the attacker could observe in a state of player 1



- Bad states : $Bad = 2^F$
  - Player 2 : find a strategy allowing to reach Bad : *reachability game*
  - Player 1 : find a strategy that avoids Bad : *safety game*

Dynamic Filter Synthesis Problem as a Game Problem

- Reduce the Synthesis Problem to a safety 2-player game
  - Player 1 play the role of a dynamical projection (what remains observable)
    - Player 1 State $\subseteq 2^Q$: estimate of the attacker after an observation
  - Player 2 what the attacker observes.
    - Player 2 State $\subseteq 2^Q\Sigma$: what the attacker could observe in a state of player 1



- Bad states : $Bad = 2^F$
  - Player 2 : find a strategy allowing to reach Bad : *reachability game*
  - Player 1 : find a strategy that avoids Bad : *safety game*

Outline

Other aspects

- Control :
  - Control of opacity with several attackers
  - Enforcement of opacity
  - Enforcement of opacity properties using insertion
- Other models :
  - Timed automata
  - Petri Nets
  - Recursive tiles systems
- Quantitative aspects ?
  - Stochastic DES
  - Prevent different users to infer secret information from other users