

Synthesis of Non-Interferent Distributed Systems

Franck Cassez¹

John Mullins²

Olivier H. Roux¹

¹CNRS/IRCCyN
Nantes, France

²Ecole Polytechnique
Montréal, Canada

MMS-ACNS'07
September 2007

St. Petersburg, Russia

Outline of the talk

- ▶ **Non-Interference & Control**
 - Non-Interference
 - Checking Non-Interference
 - Control vs. Verification
 - Non-Interference Control Problems

- ▶ **Control of Non-Interference**
 - Ramadge & Wonham Approach to Control
 - SNNI Control Problem
 - μ -Calculus Control Problem
 - BSNNI Control Problem

- ▶ **Conclusion**

Outline of the talk

- ▶ **Non-Interference & Control**
 - Non-Interference
 - Checking Non-Interference
 - Control vs. Verification
 - Non-Interference Control Problems

- ▶ **Control of Non-Interference**
 - Ramadge & Wonham Approach to Control
 - SNNI Control Problem
 - μ -Calculus Control Problem
 - BSNNI Control Problem

- ▶ **Conclusion**

Outline of the talk

- ▶ **Non-Interference & Control**
 - Non-Interference
 - Checking Non-Interference
 - Control vs. Verification
 - Non-Interference Control Problems

- ▶ **Control of Non-Interference**
 - Ramadge & Wonham Approach to Control
 - SNNI Control Problem
 - μ -Calculus Control Problem
 - BSNNI Control Problem

- ▶ **Conclusion**

Outline

- ▶ **Non-Interference & Control**
 - Non-Interference
 - Checking Non-Interference
 - Control vs. Verification
 - Non-Interference Control Problems

- ▶ Control of Non-Interference

- ▶ Conclusion

Non-Interference, Modelling & Specification

Source: [Focardi & Gorrieri, FOSAD'01]

Non-Interference is a security property

Non-Interference:

- ▶ A High Level User vs. a Low Level User
- ▶ Non-Interference: high level information never flows to low level
effects of actions of high level user are not visible by low level user

Numerous applications:

- ▶ cryptography
- ▶ confidentiality,
- ▶ anonymity ...

Formal Methods for Non-Interference:

- ▶ Users are modelled by (finite) labeled transition systems (LTS)
- ▶ A Non-Interference property = property of LTS

Non-Interference is not in the scope of safety/liveness properties

Non-Interference Properties

Input: a non-deterministic LTS A with labels in $\Sigma_c \cup \Sigma_u$
 Σ_c = high level actions, Σ_u = low level actions

Assumption: high level actions are not visible by low level

- ▶ **Language (or Trace) Based Non-Interference**
 - ▶ Strong Non Deterministic Non-Interference **SNNI**

check that $\mathcal{L}(A/\Sigma_c) = \mathcal{L}(A \setminus \Sigma_c)$

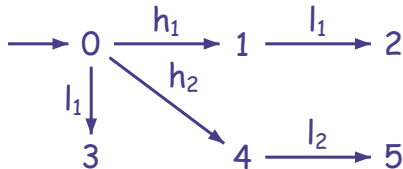
- ▶ P-SNNI = **persistent** SNNI, T-SNNI = **Test** SNNI, ...
- ▶ **Bisimulation Based Non-Interference**
 - ▶ Bisimulation Strong Non Deterministic Non-Interference **BSNNI**

check that A/Σ_c is weakly bisimilar to $A \setminus \Sigma_c$

- ▶ P-BSNNI, ...
- ▶ BNDC, P-BNDC

Small Example

$$\Sigma_h = \{h_1, h_2\}, \Sigma_l = \{l_1, l_2\}$$

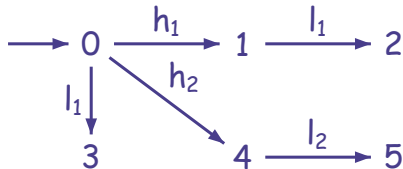


SNNI ?

BSNNI ?

Small Example

$$\Sigma_h = \{h_1, h_2\}, \Sigma_l = \{l_1, l_2\}$$

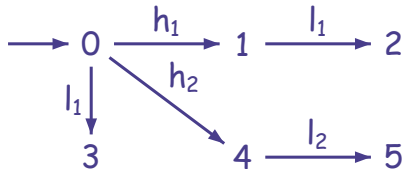


SNNI ?

BSNNI ?

Small Example

$$\Sigma_h = \{h_1, h_2\}, \Sigma_l = \{l_1, l_2\}$$



SNNI ?

BSNNI ?

Algorithms for Checking SNNI & BSNNI

- ▶ Non-Interference Verification Problems:
 Input: a LTS A , actions partitioned in $\Sigma_c \cup \Sigma_u$
 - SNNI-VP $\mathcal{L}(A/\Sigma_c) = \mathcal{L}(A \setminus \Sigma_c)$?
 - BSNNI-VP Is A/Σ_c weakly bisimilar to $A \setminus \Sigma_c$?
- ▶ Algorithms for Non-Interference Verification Problems
 [Van der Meyden & Zhang, VODCA'06] and
 [Focardi & Gorrieri, FOSAD'01]
 - ▶ SNNI-VP: language equivalence for non-deterministic LTS
 PSPACE
 - ▶ BSNNI-VP: polynomial in size of A/Σ_c
 PTIME

Verification and Control

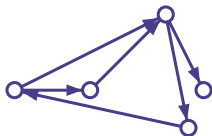
Verification and Control

Does the system meet the specification ?

Verification and Control

Does the system meet the specification ?

Modelling



S

\models

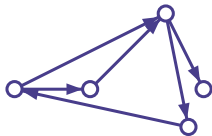
ψ

(not bad)

Verification and Control

Does the system meet the specification ?

Modelling



□ (not bad)

$S \models \varphi$

Verification/Model Checking Problem

Does the closed system S satisfy φ ?

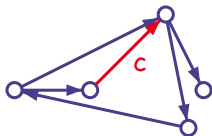
Verification and Control

Can we enforce the system to meet the specification ?

Verification and Control

Can we enforce the system to meet the specification ?

Modelling



S

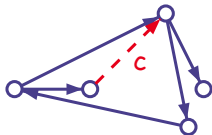
(not bad)

ψ

Verification and Control

Can we enforce the system to meet the specification ?

Modelling



S

(not bad)

ψ

Control Problem

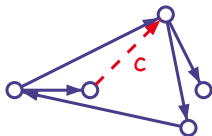
Can the **open system** S be **restricted** to satisfy ψ ?

Is there a **Controller** C s.t. $(S \parallel C) \models \psi$?

Verification and Control

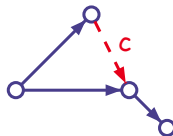
Can we enforce the system to meet the specification?

Modelling



S

\parallel



C

\parallel

\models

φ

□ (not bad)

Control Problem

Can the **open system** S be **restricted** to satisfy φ ?

Is there a **Controller** C s.t. $(S \parallel C) \models \varphi$?

Non-Interference Control Problems

SNNI-Control Problem (SNNI-CP)

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: Is there a controller C s.t. $(C \times A)/\Sigma_c$ and $(C \times A) \setminus \Sigma_c$ are language equivalent?

BSNNI-Control Problem (BSNNI-CP)

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: Is there a controller C s.t. $(C \times A)/\Sigma_c$ is weakly bisimilar to $(C \times A) \setminus \Sigma_c$?

Synthesis Problem

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: If the answer to SNNI-CP or BSNNI-CP is "yes" can we effectively compute a controller?

Non-Interference Control Problems

SNNI-Control Problem (SNNI-CP)

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: Is there a controller C s.t. $(C \times A)/\Sigma_c$ and $(C \times A) \setminus \Sigma_c$ are language equivalent?

BSNNI-Control Problem (BSNNI-CP)

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: Is there a controller C s.t. $(C \times A)/\Sigma_c$ is weakly bisimilar to $(C \times A) \setminus \Sigma_c$?

Synthesis Problem

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: If the answer to SNNI-CP or BSNNI-CP is "yes" can we effectively compute a controller?

Non-Interference Control Problems

SNNI-Control Problem (SNNI-CP)

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: Is there a controller C s.t. $(C \times A)/\Sigma_c$ and $(C \times A) \setminus \Sigma_c$ are language equivalent?

BSNNI-Control Problem (BSNNI-CP)

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: Is there a controller C s.t. $(C \times A)/\Sigma_c$ is weakly bisimilar to $(C \times A) \setminus \Sigma_c$?

Synthesis Problem

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: If the answer to SNNI-CP or BSNNI-CP is "yes" can we effectively compute a controller?

Non-Interference Control Problems

SNNI-Control Problem (SNNI-CP)

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: Is there a controller C s.t. $(C \times A)/\Sigma_c$ and $(C \times A) \setminus \Sigma_c$ are **language equivalent** ?

BSNNI-Control Problem (BSNNI-CP)

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: Is there a controller C s.t. $(C \times A)/\Sigma_c$ is **weakly bisimilar** to $(C \times A) \setminus \Sigma_c$?

Synthesis Problem

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: If the answer to SNNI-CP or BSNNI-CP is "yes" can we **effectively compute** a controller ?

Outline

- ▶ Non-Interference & Control
- ▶ **Control of Non-Interference**
 - Ramadge & Wonham Approach to Control
 - SNNI Control Problem
 - μ -Calculus Control Problem
 - BSNNI Control Problem
- ▶ Conclusion

Control à la Ramadge & Wonham [R&W 87, R&W'89]

K-Control Problem

Input: a (prefix closed) language L , $\Sigma_c \cup \Sigma_u$, a specification K

Problem: Is there a controller C s.t. $\mathcal{L}(C \times A) = K$?

Supervisory Control Problem (SupCP)

Input: a (prefix closed) language L , $\Sigma_c \cup \Sigma_u$, a specification K

Problem: Compute the largest $K' \subseteq K$ s.t. there is a controller C s.t. $\mathcal{L}(C \times A) = K'$.

Algorithm for SupCP:

- ▶ K and A are given by a non-deterministic finite LTS
- ① Compute $A \times \text{Complement}(K)$
Bad states are accepting states of $\text{Complement}(K)$
- ② use standard techniques for games to compute a **most liberal** and **memoryless** controller for $A \times \text{Complement}(K)$
- ③ SupCP is **exponential** in the size of $|A|$ and $|K|$

Control à la Ramadge & Wonham [R&W 87, R&W'89]

K-Control Problem

Input: a (prefix closed) language L , $\Sigma_c \cup \Sigma_u$, a specification K

Problem: Is there a controller C s.t. $\mathcal{L}(C \times A) = K$?

Supervisory Control Problem (SupCP)

Input: a (prefix closed) language L , $\Sigma_c \cup \Sigma_u$, a specification K

Problem: Compute the largest $K' \subseteq K$ s.t. there is a controller C s.t. $\mathcal{L}(C \times A) = K'$.

Algorithm for SupCP:

- ▶ K and A are given by a non-deterministic finite LTS
- ① Compute $A \times \text{Complement}(K)$
Bad states are accepting states of $\text{Complement}(K)$
- ② use standard techniques for games to compute a **most liberal** and **memoryless** controller for $A \times \text{Complement}(K)$
- ③ SupCP is **exponential** in the size of $|A|$ and $|K|$

Control à la Ramadge & Wonham [R&W 87, R&W'89]

K-Control Problem

Input: a (prefix closed) language L , $\Sigma_c \cup \Sigma_u$, a specification K

Problem: Is there a controller C s.t. $\mathcal{L}(C \times A) = K$?

Supervisory Control Problem (SupCP)

Input: a (prefix closed) language L , $\Sigma_c \cup \Sigma_u$, a specification K

Problem: Compute the largest $K' \subseteq K$ s.t. there is a controller C s.t. $\mathcal{L}(C \times A) = K'$.

Algorithm for SupCP:

- ▶ K and A are given by a non-deterministic finite LTS
- ① Compute $A \times \text{Complement}(K)$
Bad states are accepting states of $\text{Complement}(K)$
- ② use standard techniques for games to compute a **most liberal** and **memoryless** controller for $A \times \text{Complement}(K)$
- ③ SupCP is **exponential** in the size of $|A|$ and $|K|$

Control à la Ramadge & Wonham [R&W 87, R&W'89]

K-Control Problem

Input: a (prefix closed) language L , $\Sigma_c \cup \Sigma_u$, a specification K

Problem: Is there a controller C s.t. $\mathcal{L}(C \times A) = K$?

Supervisory Control Problem (SupCP)

Input: a (prefix closed) language L , $\Sigma_c \cup \Sigma_u$, a specification K

Problem: Compute the largest $K' \subseteq K$ s.t. there is a controller C s.t. $\mathcal{L}(C \times A) = K'$.

Algorithm for SupCP:

- ▶ K and A are given by a non-deterministic finite LTS
- ① Compute $A \times \text{Complement}(K)$
Bad states are accepting states of $\text{Complement}(K)$
- ② use standard techniques for games to compute a **most liberal** and **memoryless** controller for $A \times \text{Complement}(K)$
- ③ SupCP is **exponential** in the size of $|A|$ and $|K|$

Control à la Ramadge & Wonham [R&W 87, R&W'89]

K-Control Problem

Input: a (prefix closed) language L , $\Sigma_c \cup \Sigma_u$, a specification K

Problem: Is there a controller C s.t. $\mathcal{L}(C \times A) = K$?

Supervisory Control Problem (SupCP)

Input: a (prefix closed) language L , $\Sigma_c \cup \Sigma_u$, a specification K

Problem: Compute the largest $K' \subseteq K$ s.t. there is a controller C s.t. $\mathcal{L}(C \times A) = K'$.

Algorithm for SupCP:

- ▶ K and A are given by a non-deterministic finite LTS
- ① Compute $A \times \text{Complement}(K)$
Bad states are accepting states of $\text{Complement}(K)$
- ② use standard techniques for games to compute a **most liberal** and **memoryless** controller for $A \times \text{Complement}(K)$
- ③ SupCP is **exponential** in the size of $|A|$ and $|K|$

Control à la Ramadge & Wonham [R&W 87, R&W'89]

K-Control Problem

Input: a (prefix closed) language L , $\Sigma_c \cup \Sigma_u$, a specification K

Problem: Is there a controller C s.t. $\mathcal{L}(C \times A) = K$?

Supervisory Control Problem (SupCP)

Input: a (prefix closed) language L , $\Sigma_c \cup \Sigma_u$, a specification K

Problem: Compute the largest $K' \subseteq K$ s.t. there is a controller C s.t. $\mathcal{L}(C \times A) = K'$.

Algorithm for SupCP:

- ▶ K and A are given by a non-deterministic finite LTS
- ① Compute $A \times \text{Complement}(K)$
Bad states are accepting states of $\text{Complement}(K)$
- ② use standard techniques for games to compute a **most liberal** and **memoryless** controller for $A \times \text{Complement}(K)$
- ③ SupCP is **exponential** in the size of $|A|$ and $|K|$

Control à la Ramadge & Wonham [R&W 87, R&W'89]

K-Control Problem

Input: a (prefix closed) language L , $\Sigma_c \cup \Sigma_u$, a specification K

Problem: Is there a controller C s.t. $\mathcal{L}(C \times A) = K$?

Supervisory Control Problem (SupCP)

Input: a (prefix closed) language L , $\Sigma_c \cup \Sigma_u$, a specification K

Problem: Compute the largest $K' \subseteq K$ s.t. there is a controller C s.t. $\mathcal{L}(C \times A) = K'$.

Algorithm for SupCP:

- ▶ K and A are given by a non-deterministic finite LTS
- ① Compute $A \times \text{Complement}(K)$
Bad states are accepting states of $\text{Complement}(K)$
- ② use standard techniques for games to compute a **most liberal** and **memoryless** controller for $A \times \text{Complement}(K)$
- ③ SupCP is **exponential** in the size of $|A|$ and $|K|$

From SNNI-CP to Supervisory Control

SNNI-Control Problem (SNNI-CP)

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: Is there a controller C s.t. $(C \times A)/\Sigma_c$ and $(C \times A) \setminus \Sigma_c$ are language equivalent ?

$$\mathcal{L}((C \times A)/\Sigma_c) = \mathcal{L}((C \times A) \setminus \Sigma_c) \iff \mathcal{L}(C \times A) \subseteq \mathcal{L}(A \setminus \Sigma_c \times \mathcal{U}) \cap \mathcal{L}(A).$$

Theorem

SNNI-CP is in EXPTIME.

From SNNI-CP to Supervisory Control

SNNI-Control Problem (SNNI-CP)

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: Is there a controller C s.t. $(C \times A)/\Sigma_c$ and $(C \times A) \setminus \Sigma_c$ are language equivalent ?

$$\mathcal{L}((C \times A)/\Sigma_c) = \mathcal{L}((C \times A) \setminus \Sigma_c) \iff \mathcal{L}(C \times A) \subseteq \mathcal{L}(A \setminus \Sigma_c \times \mathcal{U}) \cap \mathcal{L}(A).$$

Theorem

SNNI-CP is in EXPTIME.

From SNNI-CP to Supervisory Control

SNNI-Control Problem (SNNI-CP)

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: Is there a controller C s.t. $(C \times A)/\Sigma_c$ and $(C \times A) \setminus \Sigma_c$ are language equivalent ?

$$\mathcal{L}((C \times A)/\Sigma_c) = \mathcal{L}((C \times A) \setminus \Sigma_c) \iff \mathcal{L}(C \times A) \subseteq \mathcal{L}(A \setminus \Sigma_c \times \mathcal{U}) \cap \mathcal{L}(A).$$

Theorem

SNNI-CP is in EXPTIME.

μ -Calculus Control Problem

[Arnold et al., TCS'03, Riedweg & Pinchinat, MFCS'03]

μ -Calculus Control Problem (μ -CP)

Input: a LTS A , $\Sigma_c \cup \Sigma_u$, a closed μ -calculus formula φ

Problem: Is there a controller C s.t. $C \times A \models \varphi$?

Algorithm for μ -CP:

▶ K and A are given by a non-deterministic finite LTS

① Construct a **modal-loop μ -formula** $\varphi(A)$

$\varphi(A)$ is a quotient formula

② $\varphi(A)$ characterizes the good controllers:

$$C \times A \models \varphi \iff C \models \varphi(A)$$

③ transform $\varphi(A)$ into a **deterministic loop automaton**

④ **synthesize** a controller for this automaton

⑤ may cause an **exponential blow-up**

μ -Calculus Control Problem

[Arnold et al., TCS'03, Riedweg & Pinchinat, MFCS'03]

μ -Calculus Control Problem (μ -CP)

Input: a LTS A , $\Sigma_c \cup \Sigma_u$, a closed μ -calculus formula φ

Problem: Is there a controller C s.t. $C \times A \models \varphi$?

Algorithm for μ -CP:

▶ K and A are given by a non-deterministic finite LTS

① Construct a **modal-loop μ -formula** $\varphi(A)$

$\varphi(A)$ is a quotient formula

② $\varphi(A)$ characterizes the good controllers:

$$C \times A \models \varphi \iff C \models \varphi(A)$$

③ transform $\varphi(A)$ into a **deterministic loop automaton**

④ **synthesize** a controller for this automaton

⑤ may cause an **exponential blow-up**

μ -Calculus Control Problem

[Arnold et al., TCS'03, Riedweg & Pinchinat, MFCS'03]

μ -Calculus Control Problem (μ -CP)

Input: a LTS A , $\Sigma_c \cup \Sigma_u$, a closed μ -calculus formula φ

Problem: Is there a controller C s.t. $C \times A \models \varphi$?

Algorithm for μ -CP:

▶ K and A are given by a non-deterministic finite LTS

① Construct a **modal-loop μ -formula** $\varphi(A)$

$\varphi(A)$ is a quotient formula

② $\varphi(A)$ characterizes the good controllers:

$$C \times A \models \varphi \iff C \models \varphi(A)$$

③ transform $\varphi(A)$ into a **deterministic loop automaton**

④ **synthesize** a controller for this automaton

⑤ may cause an **exponential blow-up**

μ -Calculus Control Problem

[Arnold et al., TCS'03, Riedweg & Pinchinat, MFCS'03]

μ -Calculus Control Problem (μ -CP)

Input: a LTS A , $\Sigma_c \cup \Sigma_u$, a closed μ -calculus formula φ

Problem: Is there a controller C s.t. $C \times A \models \varphi$?

Algorithm for μ -CP:

▶ K and A are given by a non-deterministic finite LTS

① Construct a **modal-loop μ -formula** $\varphi(A)$

$\varphi(A)$ is a quotient formula

② $\varphi(A)$ characterizes the good controllers:

$$C \times A \models \varphi \iff C \models \varphi(A)$$

③ transform $\varphi(A)$ into a **deterministic loop automaton**

④ **synthesize** a controller for this automaton

⑤ may cause an **exponential blow-up**

μ -Calculus Control Problem

[Arnold et al., TCS'03, Riedweg & Pinchinat, MFCS'03]

μ -Calculus Control Problem (μ -CP)

Input: a LTS A , $\Sigma_c \cup \Sigma_u$, a closed μ -calculus formula φ

Problem: Is there a controller C s.t. $C \times A \models \varphi$?

Algorithm for μ -CP:

▶ K and A are given by a non-deterministic finite LTS

① Construct a **modal-loop μ -formula** $\varphi(A)$

$\varphi(A)$ is a quotient formula

② $\varphi(A)$ characterizes the good controllers:

$$C \times A \models \varphi \iff C \models \varphi(A)$$

③ transform $\varphi(A)$ into a **deterministic loop automaton**

④ **synthesize** a controller for this automaton

⑤ may cause an **exponential blow-up**

μ -Calculus Control Problem

[Arnold et al., TCS'03, Riedweg & Pinchinat, MFCS'03]

μ -Calculus Control Problem (μ -CP)

Input: a LTS A , $\Sigma_c \cup \Sigma_u$, a closed μ -calculus formula φ

Problem: Is there a controller C s.t. $C \times A \models \varphi$?

Algorithm for μ -CP:

▶ K and A are given by a non-deterministic finite LTS

① Construct a **modal-loop μ -formula** $\varphi(A)$

$\varphi(A)$ is a quotient formula

② $\varphi(A)$ characterizes the good controllers:

$$C \times A \models \varphi \iff C \models \varphi(A)$$

③ transform $\varphi(A)$ into a **deterministic loop automaton**

④ **synthesize** a controller for this automaton

⑤ may cause an **exponential blow-up**

μ -Calculus Control Problem

[Arnold et al., TCS'03, Riedweg & Pinchinat, MFCS'03]

μ -Calculus Control Problem (μ -CP)

Input: a LTS A , $\Sigma_c \cup \Sigma_u$, a closed μ -calculus formula φ

Problem: Is there a controller C s.t. $C \times A \models \varphi$?

Algorithm for μ -CP:

▶ K and A are given by a non-deterministic finite LTS

1 Construct a **modal-loop μ -formula** $\varphi(A)$

$\varphi(A)$ is a quotient formula

2 $\varphi(A)$ characterizes the good controllers:

$$C \times A \models \varphi \iff C \models \varphi(A)$$

3 transform $\varphi(A)$ into a **deterministic loop automaton**

4 **synthesize** a controller for this automaton

5 may cause an **exponential blow-up**

Characteristic Formula

- ▶ Given A with ε -transitions, A^ε is the equivalent of A without ε
- ▶ define $\models_\varepsilon: A \models_\varepsilon \varphi \iff A^\varepsilon \models \varphi; A \approx_{\mathcal{W}} B \iff A^\varepsilon \approx_S B^\varepsilon$

- 1 **Abstraction** of a set of actions:

$$\text{Let } L \subseteq \Sigma, A/L \models_\varepsilon \varphi \iff A \models \kappa_L(\varphi)$$

if φ is a μ -formula, $\kappa_L(\varphi)$ is a μ -formula

- 2 **Characteristic formula**: B with no ε -transitions.

$$\exists \text{CF}(B) \mu\text{-formula s.t. } B' \text{ is strongly bisimilar to } B \iff B' \models \text{CF}(B)$$

- 3 B an automaton without ε

$$A/L \approx_{\mathcal{W}} B \iff (A/L)^\varepsilon \approx_S B \iff (A/L)^\varepsilon \models \text{CF}(B)$$

$$\iff A/L \models_\varepsilon \text{CF}(B) \iff A \models \kappa_L(\text{CF}(B))$$

Characteristic Formula

- ▶ Given A with ε -transitions, A^ε is the equivalent of A without ε
- ▶ define $\models_\varepsilon: A \models_\varepsilon \varphi \iff A^\varepsilon \models \varphi; A \approx_{\mathcal{W}} B \iff A^\varepsilon \approx_S B^\varepsilon$

- 1 **Abstraction** of a set of actions:

$$\text{Let } L \subseteq \Sigma, A/L \models_\varepsilon \varphi \iff A \models_{\kappa_L}(\varphi)$$

if φ is a μ -formula, $\kappa_L(\varphi)$ is a μ -formula

- 2 **Characteristic formula:** B with no ε -transitions.

$$\exists CF(B) \mu\text{-formula s.t. } B' \text{ is strongly bisimilar to } B \iff B' \models CF(B)$$

- 3 B an automaton without ε

$$A/L \approx_{\mathcal{W}} B \iff (A/L)^\varepsilon \approx_S B \iff (A/L)^\varepsilon \models CF(B)$$

$$\iff A/L \models_\varepsilon CF(B) \iff A \models_{\kappa_L}(CF(B))$$

Characteristic Formula

- ▶ Given A with ε -transitions, A^ε is the equivalent of A without ε
- ▶ define \models_ε : $A \models_\varepsilon \varphi \iff A^\varepsilon \models \varphi$; $A \approx_{\mathcal{W}} B \iff A^\varepsilon \approx_S B^\varepsilon$

- 1 **Abstraction** of a set of actions:

$$\text{Let } L \subseteq \Sigma, A/L \models_\varepsilon \varphi \iff A \models \kappa_L(\varphi)$$

if φ is a μ -formula, $\kappa_L(\varphi)$ is a μ -formula

- 2 **Characteristic formula**: B with no ε -transitions.

$$\exists \text{CF}(B) \mu\text{-formula s.t. } B' \text{ is strongly bisimilar to } B \iff B' \models \text{CF}(B)$$

- 3 B an automaton without ε

$$A/L \approx_{\mathcal{W}} B \iff (A/L)^\varepsilon \approx_S B \iff (A/L)^\varepsilon \models \text{CF}(B)$$

$$\iff A/L \models_\varepsilon \text{CF}(B) \iff A \models \kappa_L(\text{CF}(B))$$

Characteristic Formula

- ▶ Given A with ε -transitions, A^ε is the equivalent of A without ε
- ▶ define $\models_\varepsilon: A \models_\varepsilon \varphi \iff A^\varepsilon \models \varphi; A \approx_{\mathcal{W}} B \iff A^\varepsilon \approx_S B^\varepsilon$

- 1 **Abstraction** of a set of actions:

$$\text{Let } L \subseteq \Sigma, A/L \models_\varepsilon \varphi \iff A \models_{\kappa_L}(\varphi)$$

if φ is a μ -formula, $\kappa_L(\varphi)$ is a μ -formula

- 2 **Characteristic formula**: B with no ε -transitions.

$$\exists \text{CF}(B) \text{ } \mu\text{-formula s.t. } B' \text{ is strongly bisimilar to } B \iff B' \models \text{CF}(B)$$

- 3 B an automaton without ε

$$A/L \approx_{\mathcal{W}} B \iff (A/L)^\varepsilon \approx_S B \iff (A/L)^\varepsilon \models \text{CF}(B)$$

$$\iff A/L \models_\varepsilon \text{CF}(B) \iff A \models_{\kappa_L}(\text{CF}(B))$$

Characteristic Formula

- ▶ Given A with ε -transitions, A^ε is the equivalent of A without ε
- ▶ define $\models_\varepsilon: A \models_\varepsilon \varphi \iff A^\varepsilon \models \varphi; A \approx_{\mathcal{W}} B \iff A^\varepsilon \approx_S B^\varepsilon$

- 1 **Abstraction** of a set of actions:

$$\text{Let } L \subseteq \Sigma, A/L \models_\varepsilon \varphi \iff A \models_{\kappa_L}(\varphi)$$

if φ is a μ -formula, $\kappa_L(\varphi)$ is a μ -formula

- 2 **Characteristic formula:** B with no ε -transitions.

$$\exists \text{CF}(B) \text{ } \mu\text{-formula s.t. } B' \text{ is strongly bisimilar to } B \iff B' \models \text{CF}(B)$$

- 3 B an automaton without ε

$$A/L \approx_{\mathcal{W}} B \iff (A/L)^\varepsilon \approx_S B \iff (A/L)^\varepsilon \models \text{CF}(B)$$

$$\iff A/L \models_\varepsilon \text{CF}(B) \iff A \models_{\kappa_L}(\text{CF}(B))$$

Characteristic Formula

- ▶ Given A with ε -transitions, A^ε is the equivalent of A without ε
- ▶ define $\models_\varepsilon: A \models_\varepsilon \varphi \iff A^\varepsilon \models \varphi; A \approx_{\mathcal{W}} B \iff A^\varepsilon \approx_S B^\varepsilon$

- 1 **Abstraction** of a set of actions:

$$\text{Let } L \subseteq \Sigma, A/L \models_\varepsilon \varphi \iff A \models \kappa_L(\varphi)$$

if φ is a μ -formula, $\kappa_L(\varphi)$ is a μ -formula

- 2 **Characteristic formula**: B with no ε -transitions.

$$\exists \text{CF}(B) \text{ } \mu\text{-formula s.t. } B' \text{ is strongly bisimilar to } B \iff B' \models \text{CF}(B)$$

- 3 B an automaton without ε

$$A/L \approx_{\mathcal{W}} B \iff (A/L)^\varepsilon \approx_S B \iff (A/L)^\varepsilon \models \text{CF}(B)$$

$$\iff A/L \models_\varepsilon \text{CF}(B) \iff A \models \kappa_L(\text{CF}(B))$$

From BSNNI to μ -CP

BSNNI-Control Problem (BSNNI-CP)

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: Is there a controller C s.t. $(C \times A)/\Sigma_c$ is **weakly bisimilar** to $(C \times A) \setminus \Sigma_c$?

Fact: $(C \times A) \setminus \Sigma_c = A \setminus \Sigma_c$

Hence

$$\begin{aligned} (C \times A)/\Sigma_c \approx_{\mathcal{W}} (C \times A) \setminus \Sigma_c &\iff (C \times A)/\Sigma_c \approx_{\mathcal{W}} A \setminus \Sigma_c \\ &\iff (C \times A) \models \kappa_{\Sigma_c}(CF(A \setminus \Sigma_c)) \end{aligned}$$

BSNNI-CP is reduced to the μ -CP: $\exists C$ s.t. $C \times A \models \kappa_{\Sigma_c}(CF(A \setminus \Sigma_c))$

From BSNNI to μ -CP

BSNNI-Control Problem (BSNNI-CP)

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: Is there a controller C s.t. $(C \times A)/\Sigma_c$ is **weakly bisimilar** to $(C \times A)\setminus\Sigma_c$?

Fact: $(C \times A)\setminus\Sigma_c = A\setminus\Sigma_c$

Hence

$$\begin{aligned} (C \times A)/\Sigma_c \approx_{\mathcal{W}} (C \times A)\setminus\Sigma_c &\iff (C \times A)/\Sigma_c \approx_{\mathcal{W}} A\setminus\Sigma_c \\ &\iff (C \times A) \models \kappa_{\Sigma_c}(CF(A\setminus\Sigma_c)) \end{aligned}$$

BSNNI-CP is reduced to the μ -CP: $\exists C$ s.t. $C \times A \models \kappa_{\Sigma_c}(CF(A\setminus\Sigma_c))$

From BSNNI to μ -CP

BSNNI-Control Problem (BSNNI-CP)

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: Is there a controller C s.t. $(C \times A)/\Sigma_c$ is **weakly bisimilar** to $(C \times A)\setminus\Sigma_c$?

Fact: $(C \times A)\setminus\Sigma_c = A\setminus\Sigma_c$

Hence

$$\begin{aligned} (C \times A)/\Sigma_c \approx_{\mathcal{W}} (C \times A)\setminus\Sigma_c &\iff (C \times A)/\Sigma_c \approx_{\mathcal{W}} A\setminus\Sigma_c \\ &\iff (C \times A) \models \kappa_{\Sigma_c}(CF(A\setminus\Sigma_c)) \end{aligned}$$

BSNNI-CP is reduced to the μ -CP: $\exists C$ s.t. $C \times A \models \kappa_{\Sigma_c}(CF(A\setminus\Sigma_c))$

From BSNNI to μ -CP

BSNNI-Control Problem (BSNNI-CP)

Input: a LTS A with $\Sigma_c \cup \Sigma_u$

Problem: Is there a controller C s.t. $(C \times A)/\Sigma_c$ is **weakly bisimilar** to $(C \times A)\setminus\Sigma_c$?

Fact: $(C \times A)\setminus\Sigma_c = A\setminus\Sigma_c$

Hence

$$\begin{aligned} (C \times A)/\Sigma_c \approx_{\mathcal{W}} (C \times A)\setminus\Sigma_c &\iff (C \times A)/\Sigma_c \approx_{\mathcal{W}} A\setminus\Sigma_c \\ &\iff (C \times A) \models \kappa_{\Sigma_c}(CF(A\setminus\Sigma_c)) \end{aligned}$$

BSNNI-CP is reduced to the μ -CP: $\exists C$ s.t. $C \times A \models \kappa_{\Sigma_c}(CF(A\setminus\Sigma_c))$

Outline

- ▶ Non-Interference & Control
- ▶ Control of Non-Interference
- ▶ **Conclusion**

Conclusion & Future Work

- ▶ Practical Aspects: treat real examples
 - ▶ SNNI-VP: need a tool that implements R&W
 - ▶ Synthesis implementing the theoretical setting of [Arnold et al., TCS'03] has been implemented [G. Point, Synthesis Tool, 2005]
- ▶ Theoretical Aspects:
 - ▶ Find the exact complexity of SNNI-CP and BSNNI-CP
 - ▶ Extend this work to other types of non-interference properties
 - ▶ Extend to systems modelled by Timed Automata

References

[Arnold et al., TCS'03]

André Arnold, Aymeric Vincent, and Igor Walukiewicz.
Games for synthesis of controllers with partial observation.
Theor. Comput. Sci., 1(303):7-34, 2003.

[Focardi & Gorrieri, FOSAD'01]

Riccardo Focardi and Roberto Gorrieri.
Classification of security properties (part I: Information flow).
 In Riccardo Focardi and Roberto Gorrieri, editors, *Foundations of Security Analysis and Design I: FOSAD 2000 Tutorial Lectures*, volume 2171 of *Lecture Notes in Computer Science*, pages 331-396, Heidelberg, 2001. Springer-Verlag.

[G. Point, Synthesis Tool, 2005]

Gérald Point.
The synthesis toolbox - from modal automata to controller synthesis.
 Technical Report 1342-05, LaBRI, 2005.
 available at <http://www.labri.fr/publications/mvtsi/2005/Poi05>.

[R&W 87]

P. Ramadge and W. Wonham.
Supervisory control of a class of discrete event processes.
SIAM J. Control Optim., 25(1), January 1987.

[R&W'89]

P. Ramadge and W. Wonham.
The control of discrete event systems.
Proceedings of the IEEE, January 1989.

[Riedweg & Pinchinat, MFCS'03]

Stéphane Riedweg and Sophie Pinchinat.
Quantified mu-calculus for control synthesis.
 In *28th International Symposium on Mathematical Foundations of Computer Science (MFCS'03)*, Bratislava, Slovakia, August 25-29, 2003, volume 2747 of *Lecture Notes in Computer Science*, pages 642-651, 2003.

[Van der Meyden & Zhang, VODCA'06]

Ron van der Meyden and Chenyi Zhang.
Algorithmic verification of noninterference properties.
 In *Proceedings of the Second International Workshop on Views on Designing Complex Architectures (VODCA 2006)*, volume 168 of *Electronic Notes in Theoretical Computer Science*, pages 61-75. Elsevier, 2006.