# Analysis and Control of Partially-Observed Discrete-Event Systems: Introduction and Recent Advances

Xiang Yin

EECS Department, University of Michigan

Department of Automation, Shanghai Jiao-Tong University

May 27, 2016, Shanghai, China

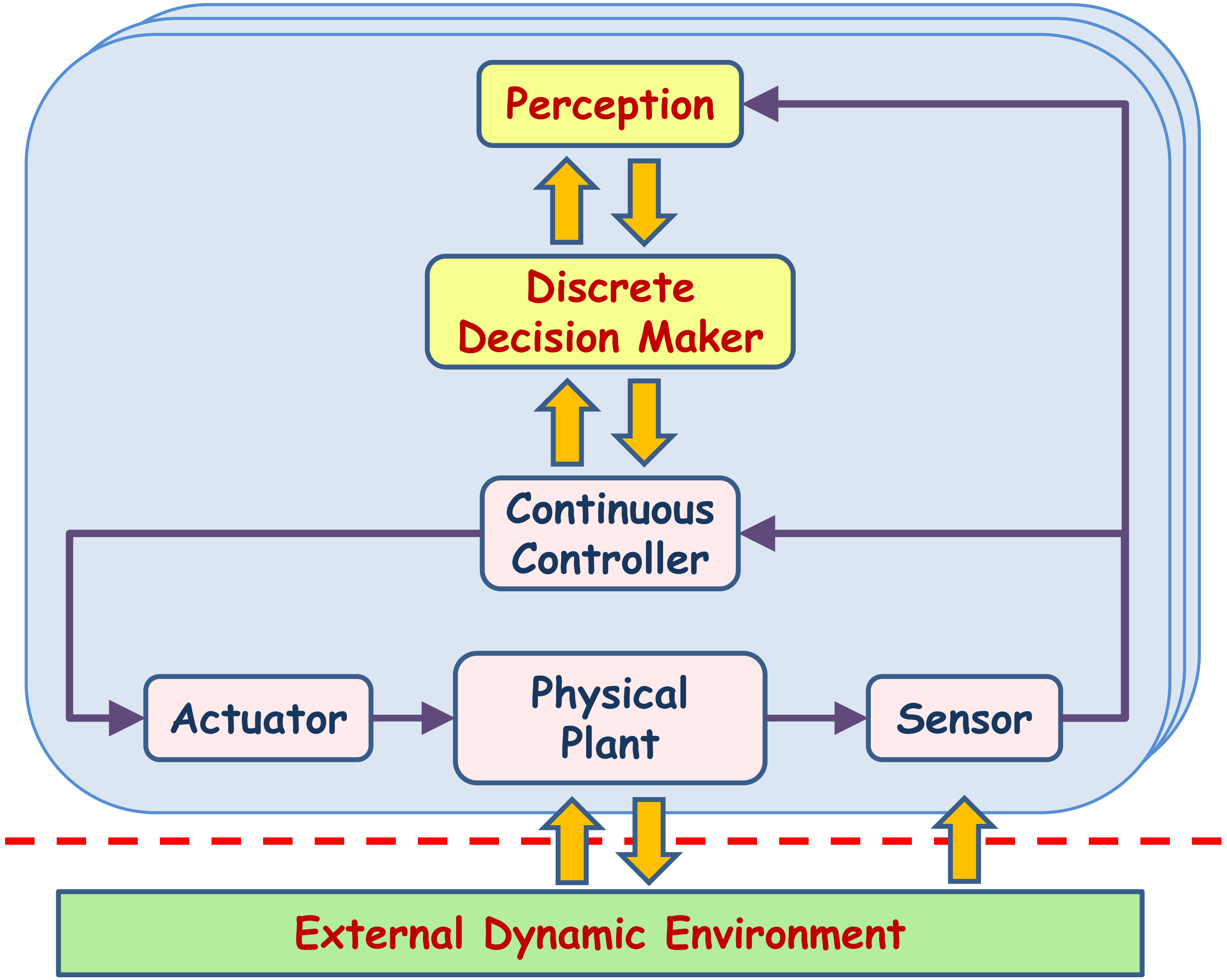
# Myself

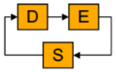
- **Name:** 殷翔 **Born:** Jan 1991, Hefei, Anhui
- **Education**
  - **Zhejiang University**, College of Electrical Engineering
    Bachelor of Engineering, Major: Power Electronics June 2012
  - **University of Michigan**, Ann Arbor, Department of EECS
    * Master of Science, Major: Control & Math Dec 2013
    * PhD Candidate, Major: Control & Math April 2017 (expected)
    * Advisor: Prof. Stephane Lafortune
    * Thesis Committee: D. Teneketzis, D. Tilbury & N. Ozay
- **Research**
  - Control of discrete-event/hybrid systems
  - Model-based fault diagnosis/prognosis
  - Privacy and security in cyber-physical systems

# Outline

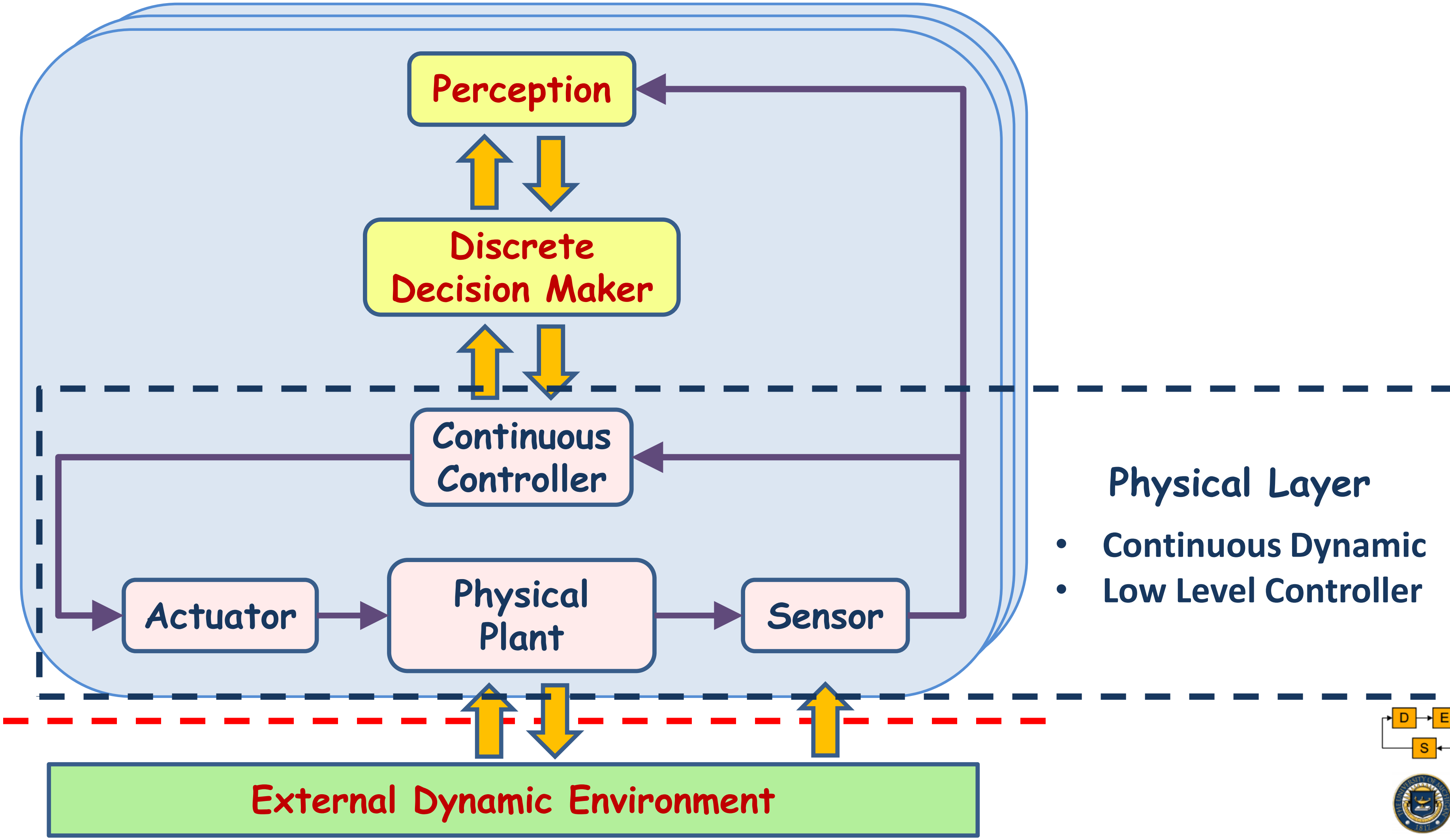
- Motivation: Why we study discrete-event system
- Partially-Observed Discrete-Event Systems
- Analysis of Partially-Observed DES
  - Verification of Security/Diagnosability/Prognosability
- Control of Partially-Observed DES
  - Synthesis of supervisory control strategies
  - Synthesis of sensor activation strategies
- Applications:
  - Location-Based Services (analysis, security issue)
  - Vehicular Electrical Power Systems (control, safety-critical systems)
- Conclusion and Future Directions

## Cyber-Physical Control Systems

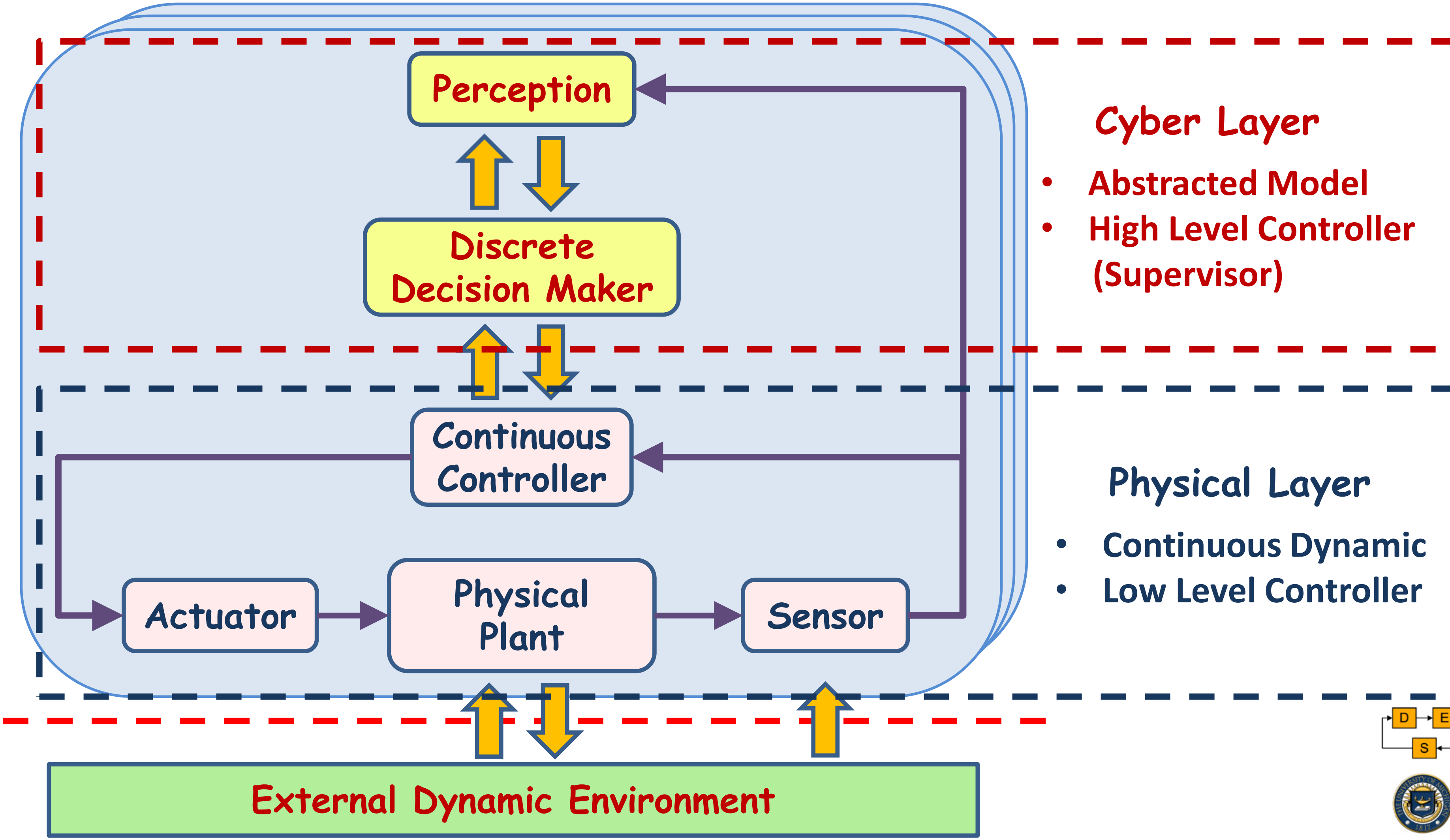


# Cyber-Physical Control Systems



# Cyber-Physical Control Systems



# Continuous v.s. Discrete

## physical, continuous

$$\dot{x}_p = f_p(x_p, u, \eta)$$
$$s = g_p(x_p, u, \mu)$$

$$\dot{x}_c = f_p(x_c, s)$$
$$u = g_p(x_c, s)$$

**Model:** Differential Equation

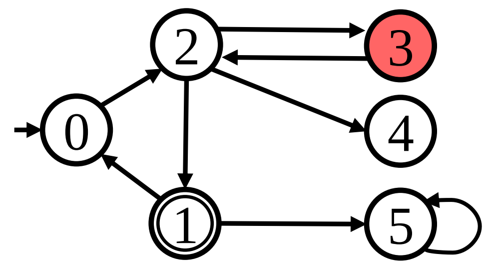
**Specification:** Stability,
reference tracking, optimality…

## physical, continuous

$$\dot{x}_p = f_p(x_p, u, \eta)$$
$$s = g_p(x_p, u, \mu)$$

$$\dot{x}_c = f_p(x_c, s)$$
$$u = g_p(x_c, s)$$

## computational, discrete




$$S: Obs(L(G)) \to 2^E$$

**Model:** Differential Equation

**Specification:** Stability, reference tracking, optimality…

**Model:** Discrete-event systems, automata, transition systems, formal languages

**Specification:** Safety, liveness, diagnosability, security

# Current Practice

## Current Control Design Process for Cyber-Physical Systems

- Given some spec (plain English) use art of design (engineering intuition, experience) and extensive testing to come up with a single solution
- Ad hoc approaches, Large lists of “if-then-else” rules
- Little or no formal guarantees on correctness

# Current Practice

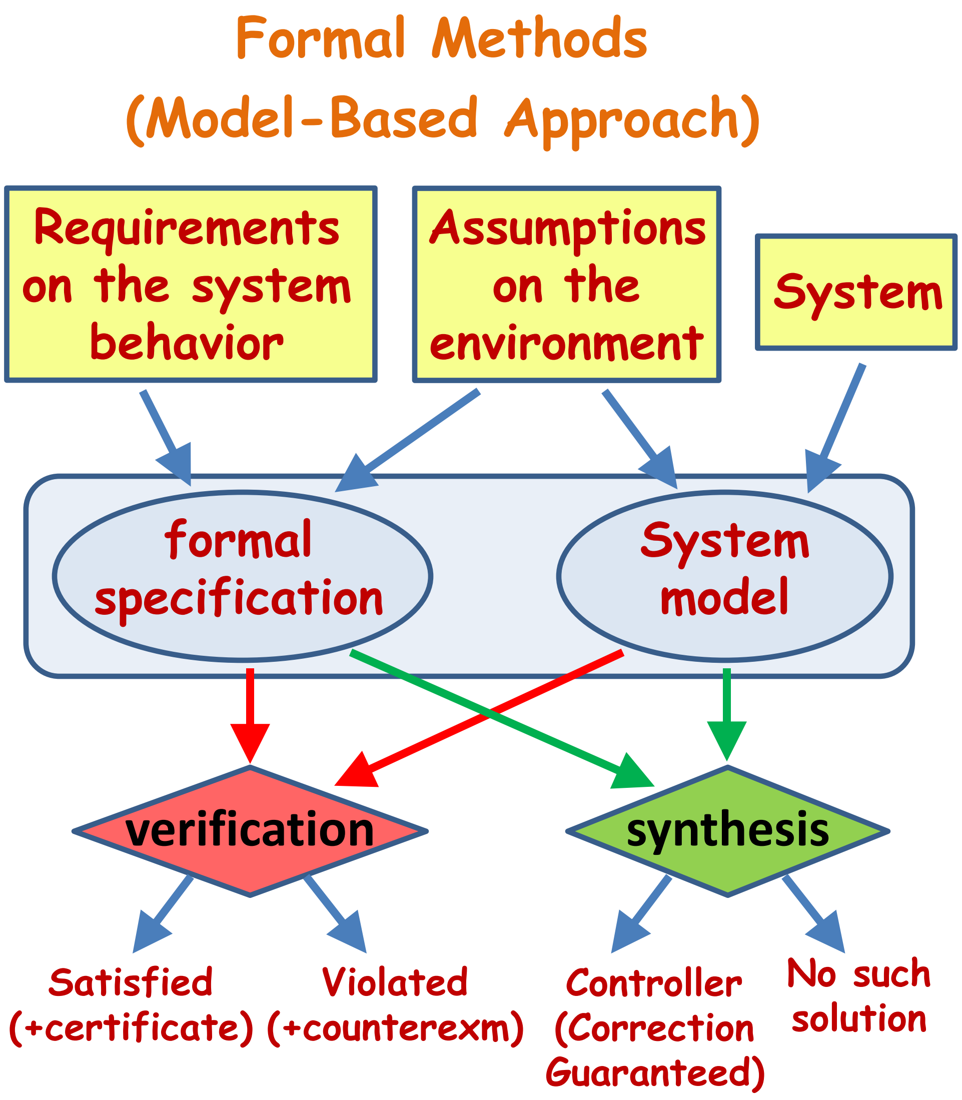
## Current Control Design Process for Cyber-Physical Systems

- Given some spec (plain English) use art of design (engineering intuition, experience) and extensive testing to come up with a single solution
- Ad hoc approaches, Large lists of “if-then-else” rules
- Little or no formal guarantees on correctness

## Better Alternative

- Formal Methods!

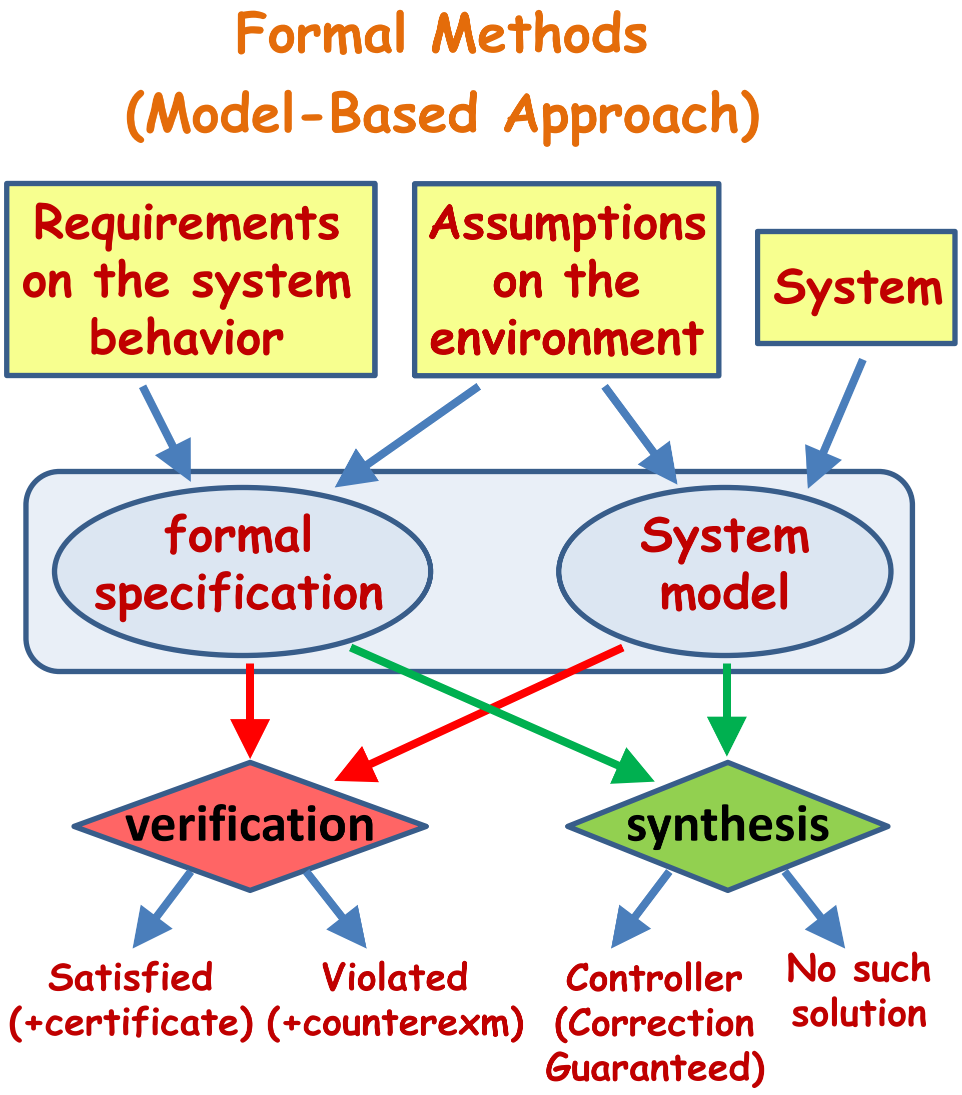
# Formal Approach: Verification and Synthesis



# Formal Approach: Verification and Synthesis

## Discrete-event systems

- Model: Automata
- Specification: Formal Languages



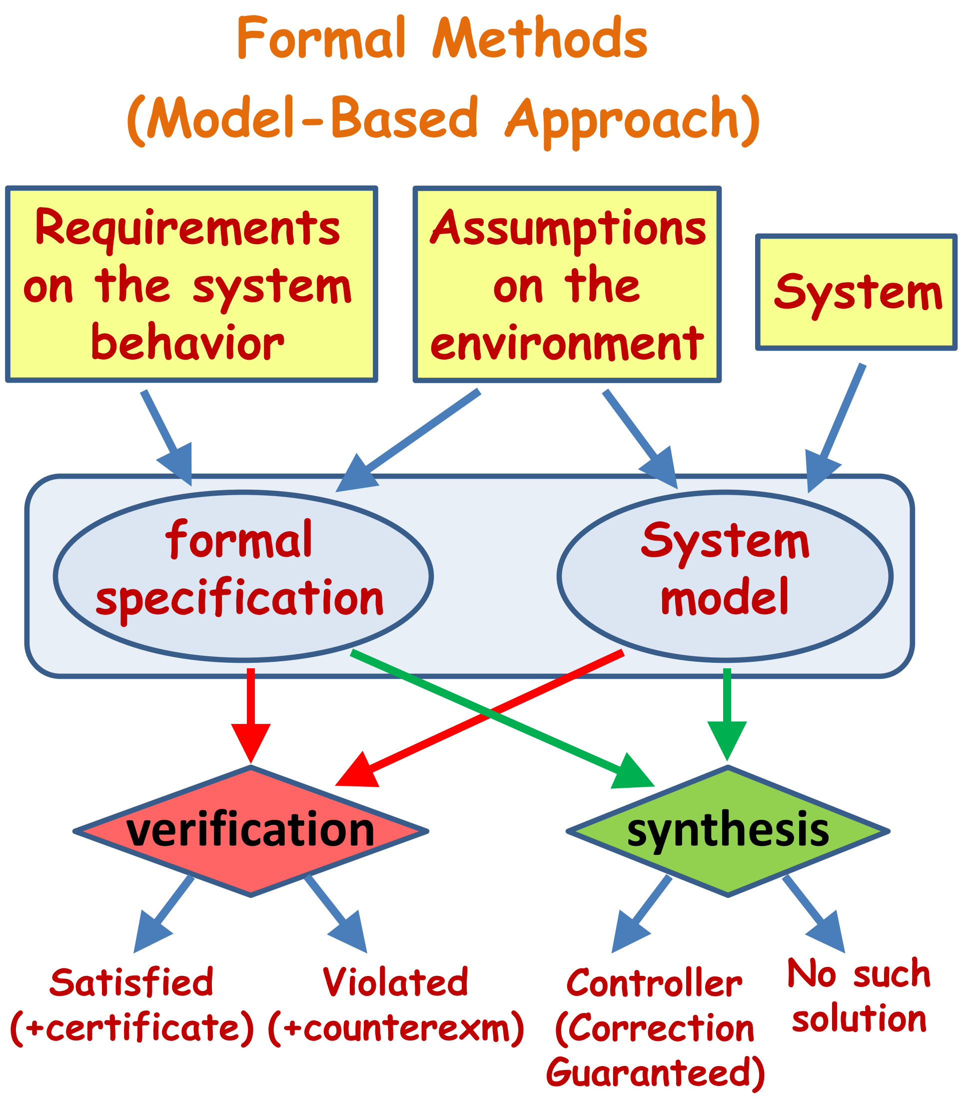
# Formal Approach: Verification and Synthesis

## Discrete-event systems

- Model: Automata
- Specification: Formal Languages

## Verification (Analysis)

- Formal guarantee for specification



# Formal Approach: Verification and Synthesis

## Discrete-event systems

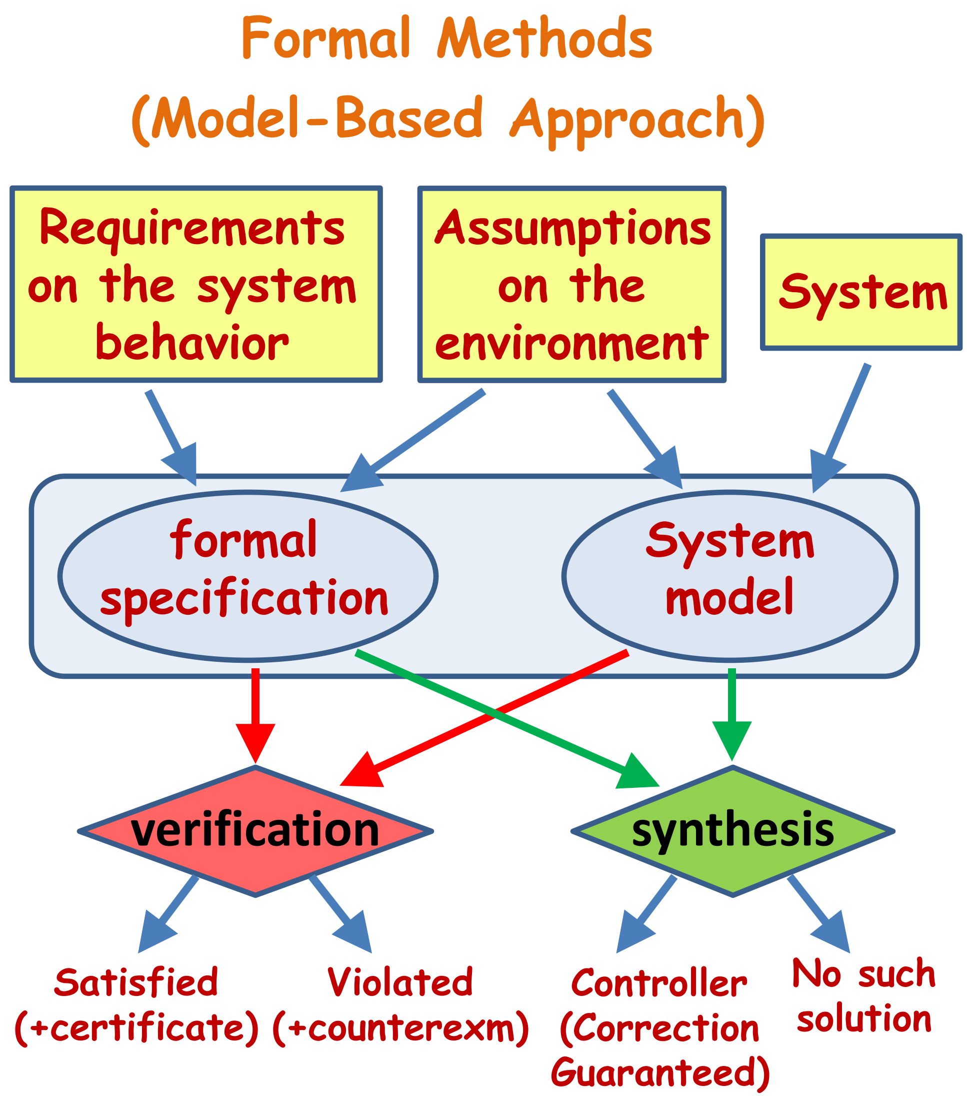
- Model: Automata
- Specification: Formal Languages

## Verification (Analysis)

- Formal guarantee for specification

## Synthesis (Control Design)

- Reactive to environment, e.g., uncontrollability & unobservability
- **Correct-by-construction!**
  **(No need to verify)**



# Why Discrete-Event Models

## Why Discrete-Event Models

- Many systems are ***Inherently Event-Driven*** and have ***Discrete State-Spaces***

Manufacturing Systems, Software Systems, PLCs, Protocols

- Z.-W. Li,, and M.-C. Zhou. "Elementary siphons of Petri nets and their application to deadlock prevention in flexible manufacturing systems." *IEEE Trans Systems, Man and Cybernetics, Part A*, 34.1, 2004.
- Y. Pencolé, and M. Cordier. "A formal framework for the decentralised diagnosis of large scale discrete event systems and its application to telecommunication networks." *Artificial Intelligence,* 164.1, 2005.
- H.-W. Liao, et al. "Eliminating concurrency bugs in multithreaded software: A new approach based on discrete-event control." *IEEE Trans Control Systems Technology,* 21.6, 2013.

# Why Discrete-Event Models

## Why Discrete-Event Models

- Many systems are ***Inherently Event-Driven*** and have ***Discrete State-Spaces***

  Manufacturing Systems, Software Systems, PLCs, Protocols

  - Z.-W. Li,, and M.-C. Zhou. "Elementary siphons of Petri nets and their application to deadlock prevention in flexible manufacturing systems." *IEEE Trans Systems, Man and Cybernetics, Part A*, 34.1, 2004.
  - Y. Pencolé, and M. Cordier. "A formal framework for the decentralised diagnosis of large scale discrete event systems and its application to telecommunication networks." *Artificial Intelligence,* 164.1, 2005.
  - H.-W. Liao, et al. "Eliminating concurrency bugs in multithreaded software: A new approach based on discrete-event control." *IEEE Trans Control Systems Technology,* 21.6, 2013.

- DES Model comes from ***Finite Abstraction*** of the original continuous system

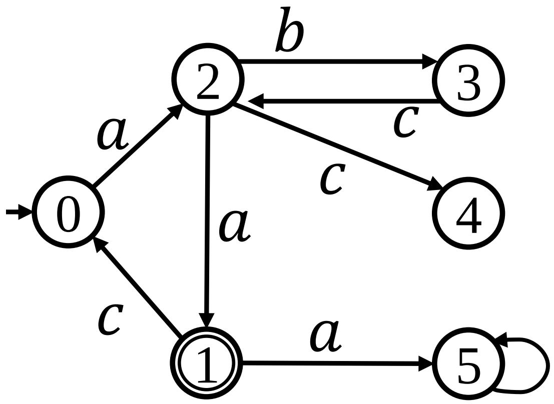
  Linear Systems, Nonlinear Systems, Stochastic Systems, Networked Systems

  - P. Tabuada and G. Pappas. "Linear time logic control of discrete-time linear systems." *IEEE Trans Automatic Control,* 51.12, 2006.
  - A. Girard, G. Pola, and P. Tabuada. "Approximately bisimilar symbolic models for incrementally stable switched systems." *IEEE Trans Automatic Control*, 55.1, 2010.
  - M. Zamani, A. Abate, and A. Girard. "Symbolic models for stochastic switched systems: a discretization and a discretization-free approach." Automatica, 55,2015.
  - M. Lahijanian, S. Andersson, and C. Belta. "Formal verification and synthesis for discrete-time stochastic systems." *IEEE Trans Automatic Control* 60.8, 2015
  - J. Liu, and N. Ozay. "Finite abstractions with robustness margins for temporal logic-based control synthesis." *Nonlinear Analysis: Hybrid Systems,* 22, 2016.

- **System Model**

$G = (X, E, f, x_0, X_m)$ is a *deterministic* FSA

- $X$ is the finite set of states
- $E$ is the finite set of events
- $f: X \times E \to X$ is the partial transition function
- $x_0$ is the initial state;
- $X_m$ is the set of marked states.

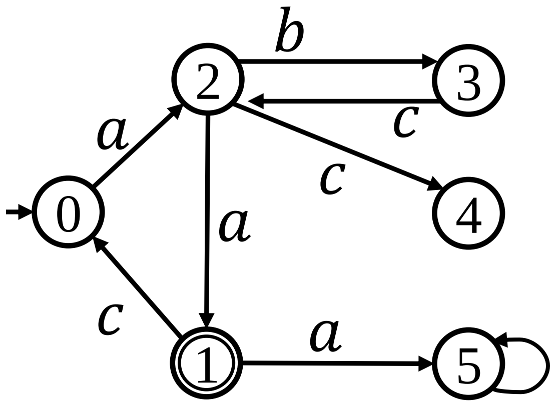



Plant $G$

- **System Model**

$G = (X, E, f, x_0, X_m)$ is a *deterministic* FSA

- $X$ is the finite set of states
- $E$ is the finite set of events
- $f: X \times E \to X$ is the partial transition function
- $x_0$ is the initial state;
- $X_m$ is the set of marked states.



Plant $G$

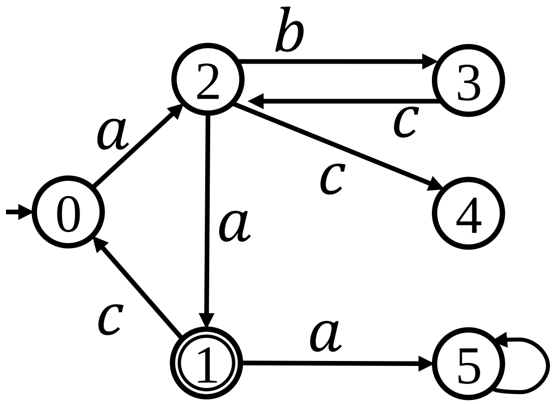
- **System's Behaviors**

- String: a sequence of events, e.g., $abccab$….

- **System Model**

  $G = (X, E, f, x_0, X_m)$ is a *deterministic* FSA

  - $X$ is the finite set of states
  - $E$ is the finite set of events
  - $f : X \times E \rightarrow X$ is the partial transition function
  - $x_0$ is the initial state;
  - $X_m$ is the set of marked states.



Plant $G$

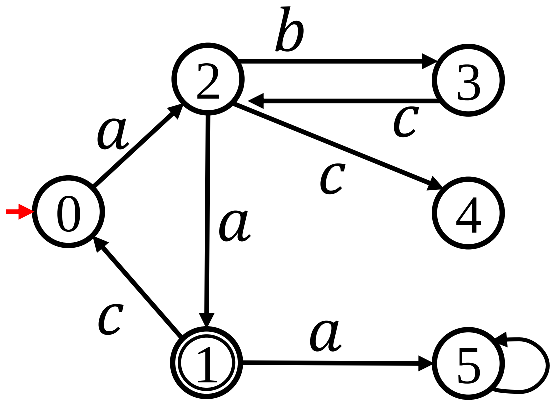
- **System's Behaviors**

  - String: a sequence of events, e.g., $abccab$....
  - Language: a set of strings

- **System Model**

$G = (X, E, f, x_0, X_m)$ is a *deterministic* FSA

- $X$ is the finite set of states
- $E$ is the finite set of events
- $f: X \times E \rightarrow X$ is the partial transition function
- $x_0$ is the initial state;
- $X_m$ is the set of marked states.




Plant $G$

- **System's Behaviors**

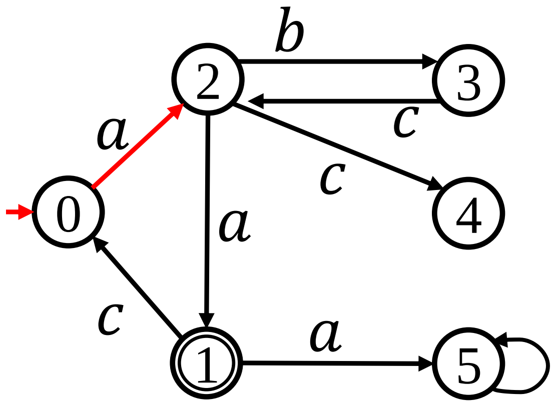
- String: a sequence of events, e.g., $abccab$....
- Language: a set of strings
- Generated language: $\mathcal{L}(G) = \{s \in E^*: f(x_0, s)!\}$

$$\mathcal{L}(G) = \{\epsilon, a, aa, ab, abc, \ldots\}$$

- **System Model**

$G = (X, E, f, x_0, X_m)$ is a *deterministic* FSA

- $X$ is the finite set of states
- $E$ is the finite set of events
- $f: X \times E \rightarrow X$ is the partial transition function
- $x_0$ is the initial state;
- $X_m$ is the set of marked states.




Plant $G$

- **System's Behaviors**

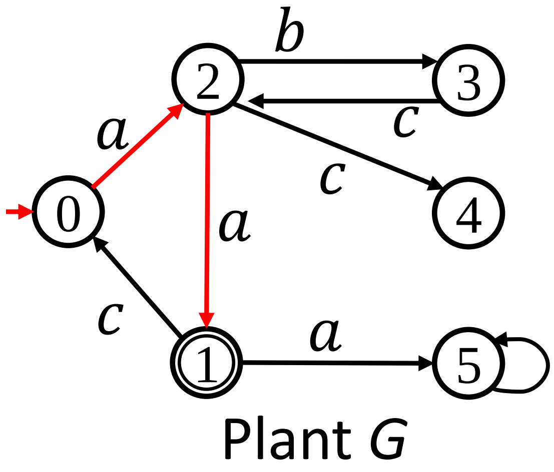
- String: a sequence of events, e.g., $abccab$....
- Language: a set of strings
- Generated language: $\mathcal{L}(G) = \{s \in E^*: f(x_0, s)!\}$

$$\mathcal{L}(G) = \{\boldsymbol{\epsilon}, \boldsymbol{a}, aa, ab, abc, \dots\}$$

- **System Model**

$G = (X, E, f, x_0, X_m)$ is a *deterministic* FSA

- $X$ is the finite set of states
- $E$ is the finite set of events
- $f: X \times E \to X$ is the partial transition function
- $x_0$ is the initial state;
- $X_m$ is the set of marked states.




- **System's Behaviors**

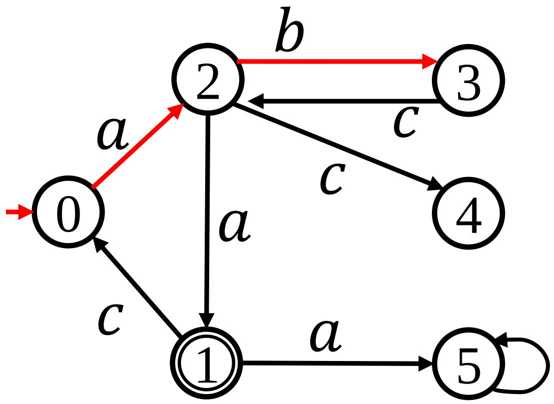
- String: a sequence of events, e.g., $abccab$....
- Language: a set of strings
- Generated language: $\mathcal{L}(G) = \{s \in E^*: f(x_0, s)!\}$

$$\mathcal{L}(G) = \{\boldsymbol{\epsilon}, \boldsymbol{a}, \boldsymbol{aa}, ab, abc, \dots\}$$

- **System Model**

  $G = (X, E, f, x_0, X_m)$ is a *deterministic* FSA

  - $X$ is the finite set of states
  - $E$ is the finite set of events
  - $f: X \times E \to X$ is the partial transition function
  - $x_0$ is the initial state;
  - $X_m$ is the set of marked states.



Plant $G$

- **System's Behaviors**

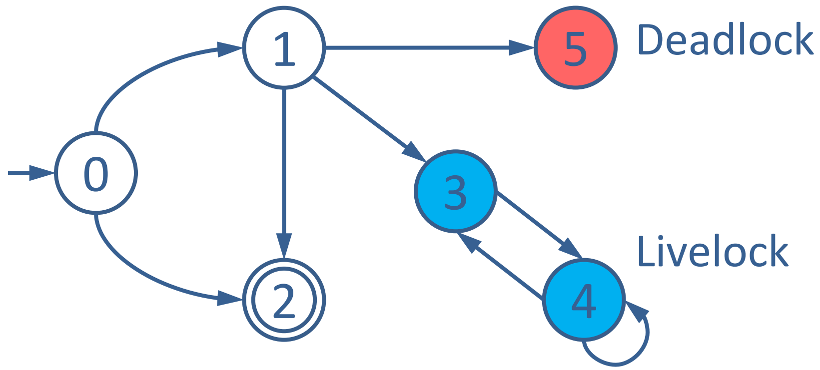
  - String: a sequence of events, e.g., $abccab$....
  - Language: a set of strings
  - Generated language: $\mathcal{L}(G) = \{s \in E^*: f(x_0, s)!\}$

    $\mathcal{L}(G) = \{\boldsymbol{\epsilon}, \boldsymbol{a}, \boldsymbol{aa}, \boldsymbol{ab}, abc, \ldots\}$

- **Formal Specifications**

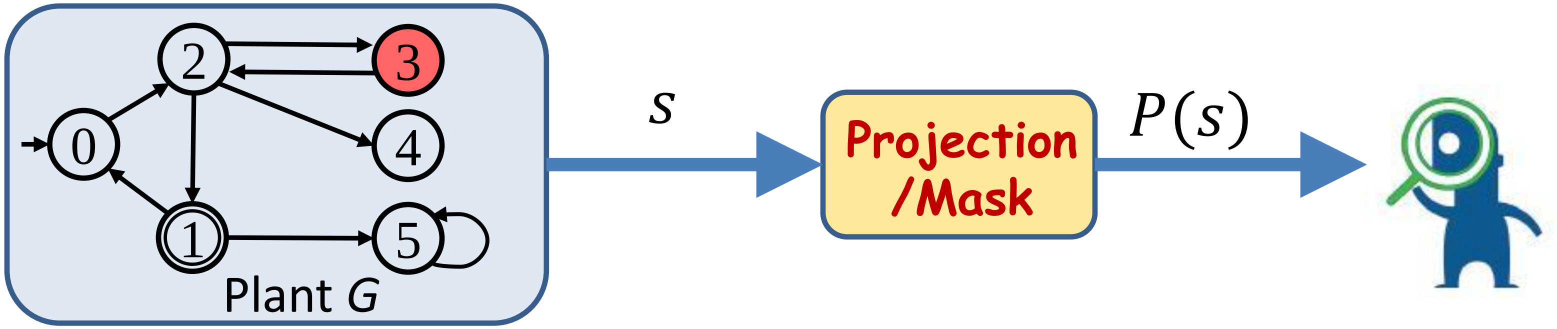
  - Safety: Regular language $L_{am}$

  - Non-blockingness: no deadlocks or livelocks



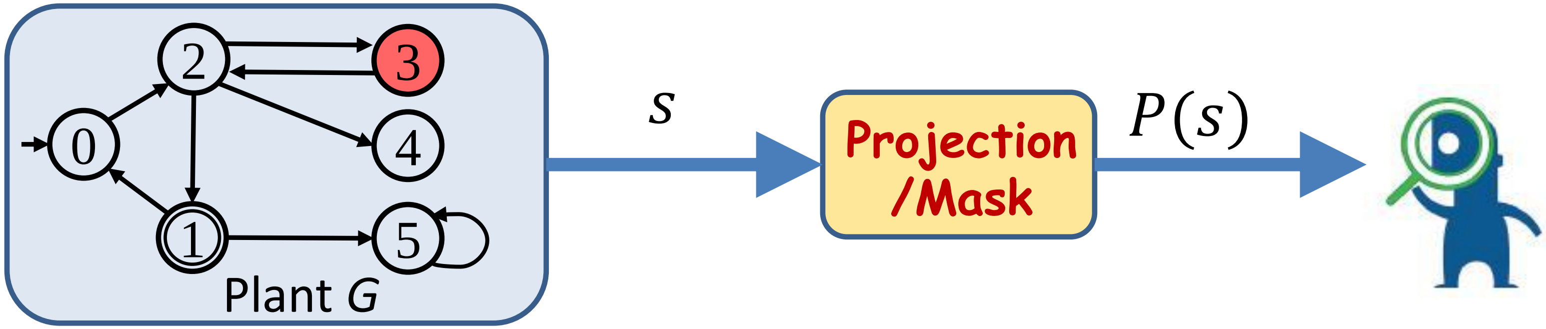

  - Other properties: Observation properties, Temporal logics

# Partially-Observed Discrete-Event Systems




- **Not all behaviors can be observed**
  - Internal behavior
  - Limited sensor capability: energy, communication constraint

# Partially-Observed Discrete-Event Systems




- **Not all behaviors can be observed**

  - Internal behavior

  - Limited sensor capability: energy, communication constraint

- **Observation Model**

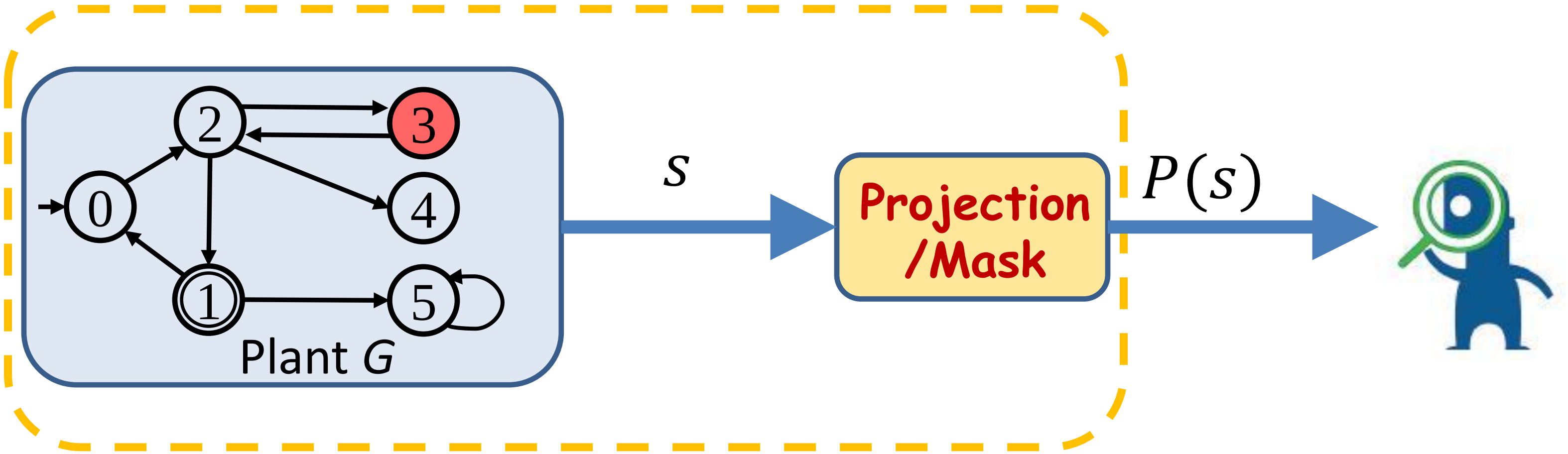
  $E = E_o \dot{\cup} E_{uo}$

- **Natural Projection** $P: E^* \to E_o^*$ erase events in $E_{uo}$

  - $E = \{a, b, c\}, E_o = \{a, b\}, P(abcca) = aba$

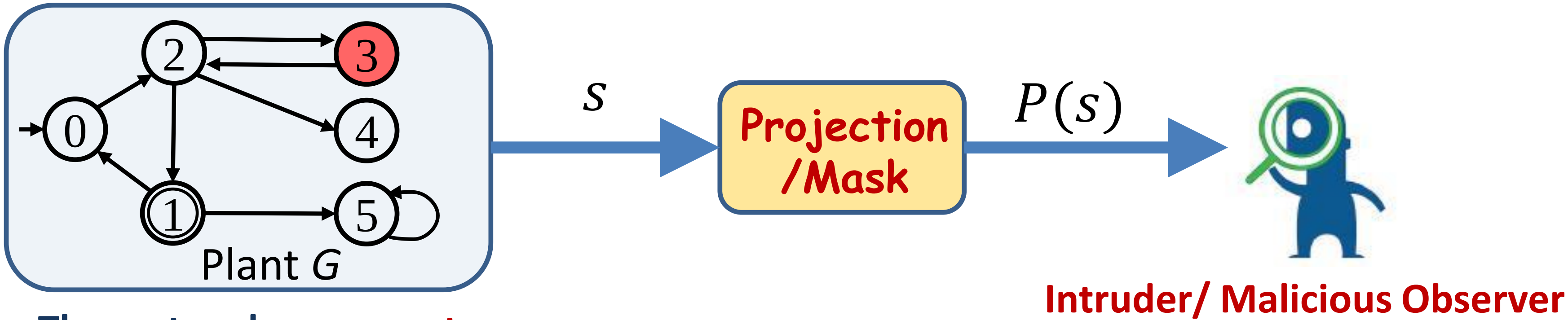
  - $P(L(G))$ is the behavior we can observe

# Property Verification of Partially-Observed DES




Does the system satisfy some ***property*** ?

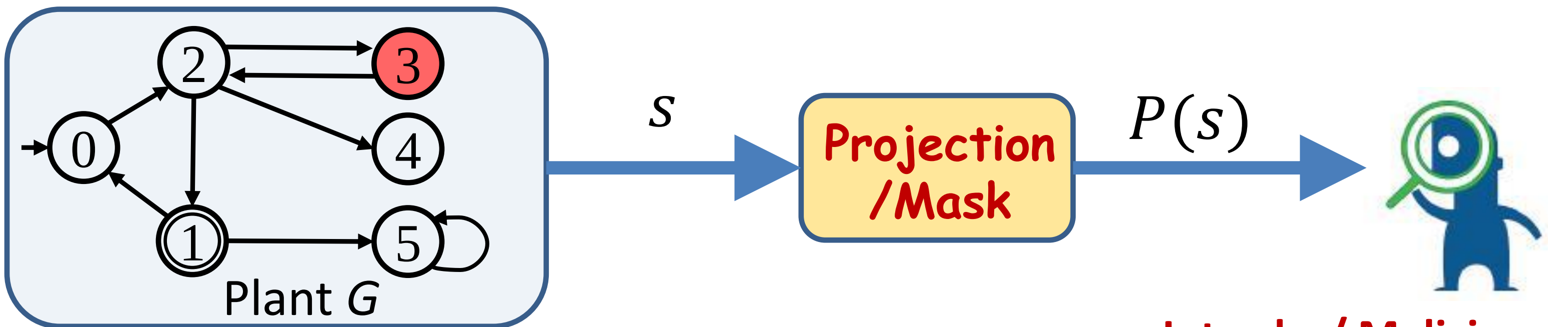
- **Opacity**: Security and privacy issue in information-flow
- **Diagnosability**: Fault detection and isolation
- **Prognosability**: Fault prediction and alarm




**The system has a secret**

- **Opacity**

The system's **secret** cannot be revealed based on the intruder's observation.




**The system has a secret**

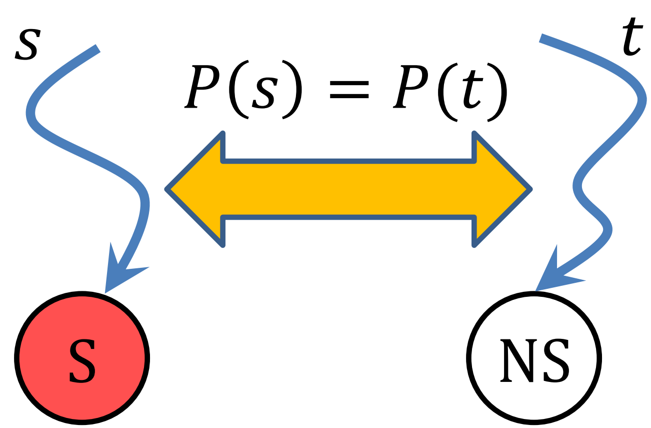
**Intruder/ Malicious Observer**

- **Opacity**

The system's **secret** cannot be revealed based on the intruder's observation.

**Current State Opacity**

- A set of secret states $X_S \subseteq X$
- The intruder never know the system is at secret state
- Ex: I know that you are visiting hospital



# *K*-Step Opacity and Infinite-Step Opacity

- **K-Step Opacity**

The intruder cannot infer that the system was at a secret state for some specific instant ***K-step ahead*** in the past.

- **Infinite-Step Opacity**

The intruder cannot infer that the system was at a secret state for **any specific instant** in the past.

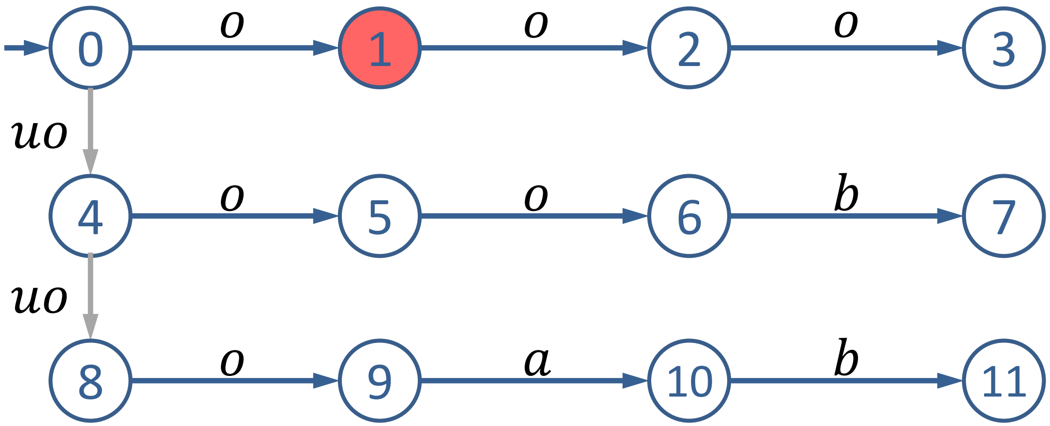
# $K$-Step Opacity and Infinite-Step Opacity

- **K-Step Opacity**

The intruder cannot infer that the system was at a secret state for some specific instant ***K-step ahead*** in the past.

- **Infinite-Step Opacity**

The intruder cannot infer that the system was at a secret state for **any specific instant** in the past.




$E_o = \{o, a, b\}$

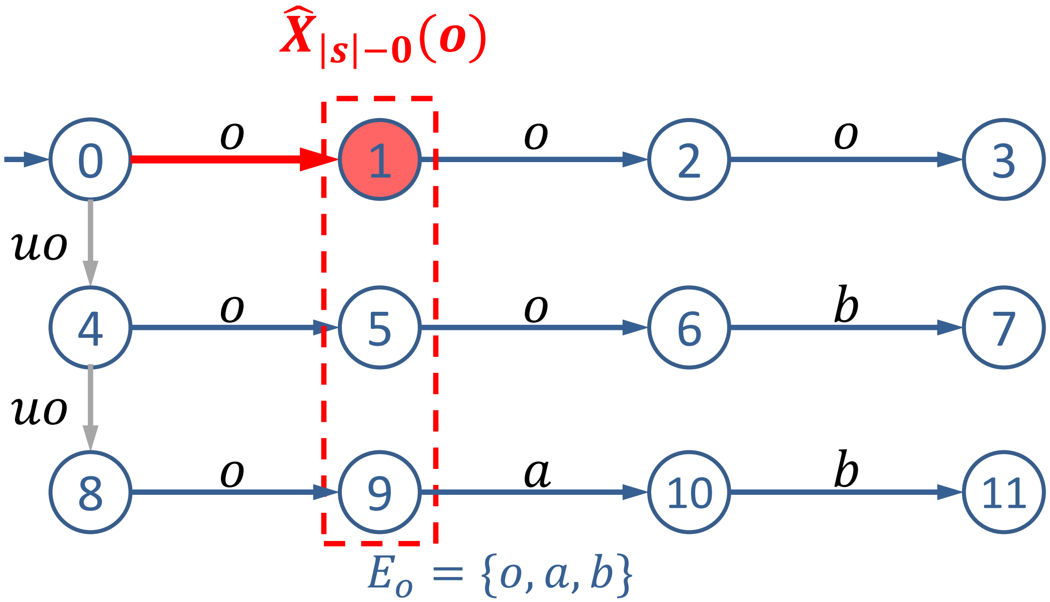
# $K$-Step Opacity and Infinite-Step Opacity

- **K-Step Opacity**

The intruder cannot infer that the system was at a secret state for some specific instant ***K-step ahead*** in the past.

- **Infinite-Step Opacity**

The intruder cannot infer that the system was at a secret state for **any specific instant** in the past.

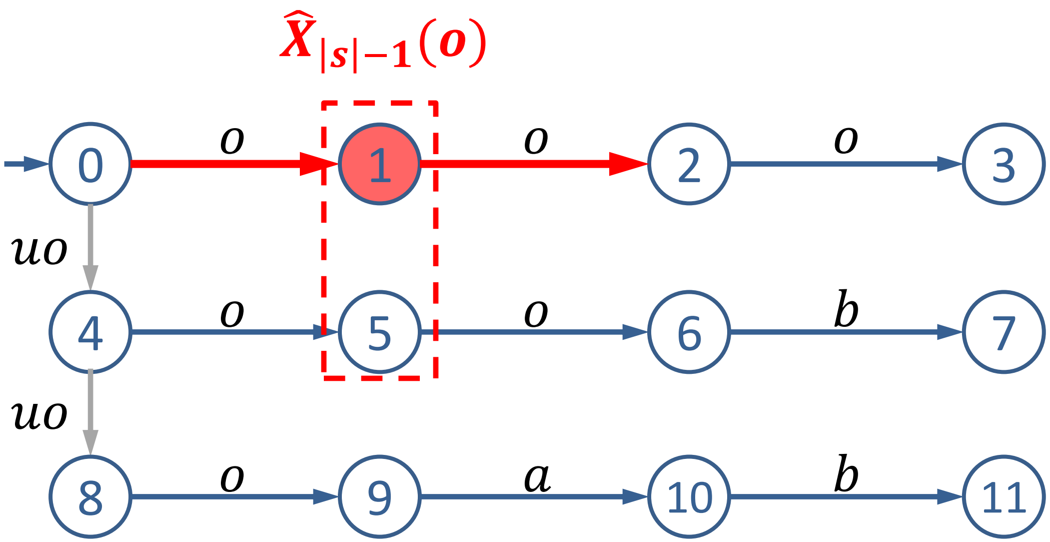


- **K-Step Opacity**

The intruder cannot infer that the system was at a secret state for some specific instant ***K-step ahead*** in the past.

- **Infinite-Step Opacity**

The intruder cannot infer that the system was at a secret state for **any specific instant** in the past.




$E_o = \{o, a, b\}$

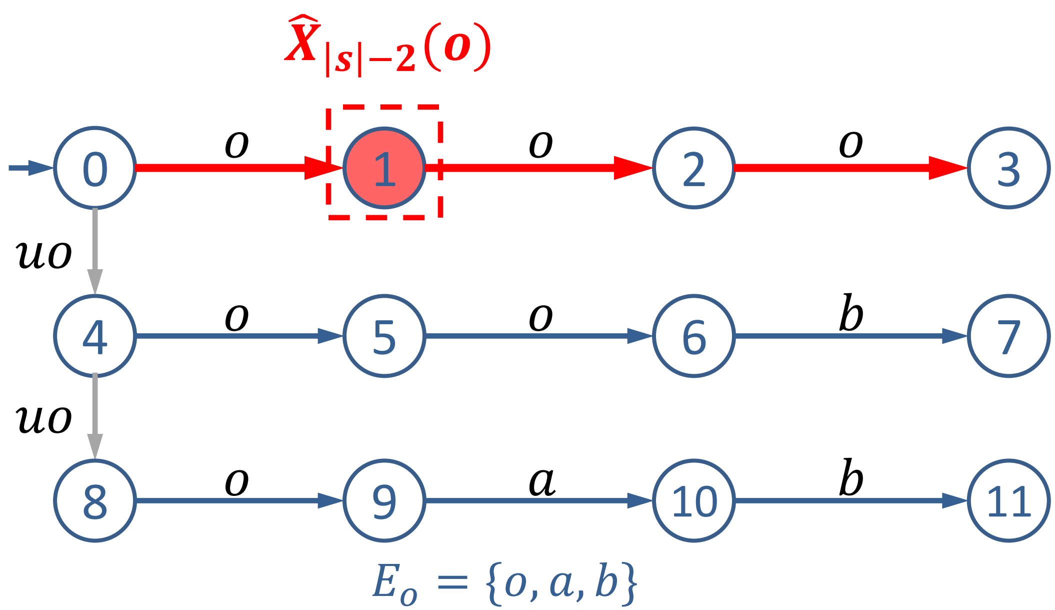
# $K$-Step Opacity and Infinite-Step Opacity

- **K-Step Opacity**

The intruder cannot infer that the system was at a secret state for some specific instant ***K-step ahead*** in the past.

- **Infinite-Step Opacity**

The intruder cannot infer that the system was at a secret state for **any specific instant** in the past.



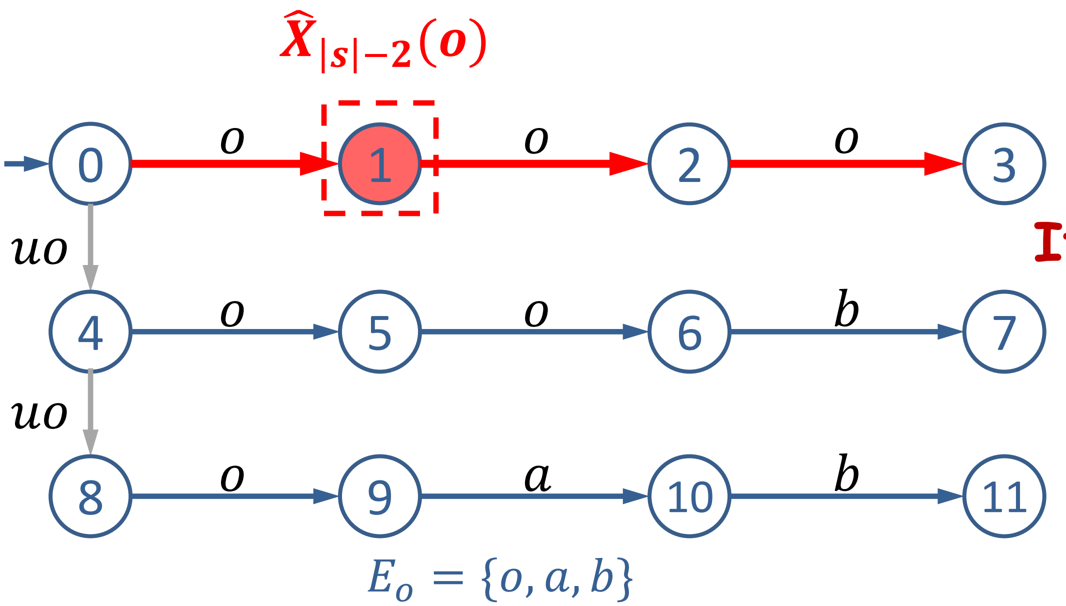
# $K$-Step Opacity and Infinite-Step Opacity

- **K-Step Opacity**

The intruder cannot infer that the system was at a secret state for some specific instant ***K-step ahead*** in the past.

- **Infinite-Step Opacity**

The intruder cannot infer that the system was at a secret state for **any specific instant** in the past.




**It is not 2-step opaque!**

# Verification of *K*-Step Opacity and Infinite-Step Opacity

- **Previous Result**
  - K-step opacity can be verified in $O(|E_o| \times 2^{|X|} \times (|E_o| + 1)^K)$ [Saboori & Hadjicostis, 2011]
  - Infinite-step opacity can be verified in $O(|E_o| \times 2^{|X|} \times 2^{|X|^2})$ [Saboori & Hadjicostis, 2013]
  - Different approaches for different properties

# Verification of *K*-Step Opacity and Infinite-Step Opacity

- **Previous Result**

  - K-step opacity can be verified in $O(|E_o| \times 2^{|X|} \times (|E_o| + 1)^K)$ [Saboori & Hadjicostis, 2011]

  - Infinite-step opacity can be verified in $O(|E_o| \times 2^{|X|} \times 2^{|X|^2})$ [Saboori & Hadjicostis, 2013]

  - Different approaches for different properties

- **Recent Advances**

  - New approach for the verification of K-step and infinite-step opacity

  - A unified approach based on a separation principle

  - K-Step: $O(|E_o| \times 2^{|X|} \times \mathbf{min\{|E_o|^K, 2^{|X|}\}})$ vs $O(|E_o| \times 2^{|X|} \times \mathbf{(|E_o| + 1)^K})$

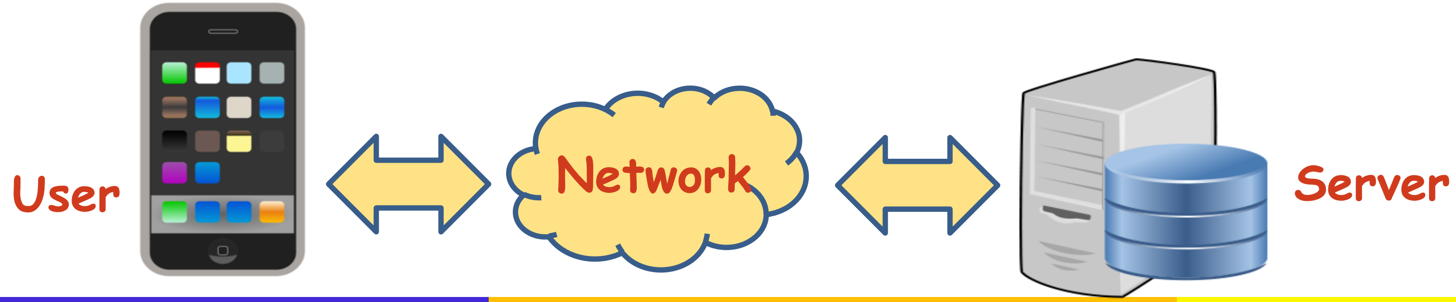
  - Infinite-Step: $O(|E_o| \times 2^{|X|} \times \mathbf{2^{|X|}})$ vs $O(|E_o| \times 2^{|X|} \times \mathbf{2^{|X|^2}})$

**X. Yin** and S. Lafortune. "A new approach for the verification of infinite-step and K-step opacity using two-way observer," ***Automatica***, under review, 2016.

**X. Yin** and S. Lafortune. "On two-way observer and its application to the verification of infinite-step and K-step opacity," ***13th Int. Workshop on Discrete Event Systems***, 2016.

## Location-Based Services

- Provide services to mobile users by exploiting their location information
- Finding nearby restaurants, tracking users' running routes, etc.
- May not be secure!

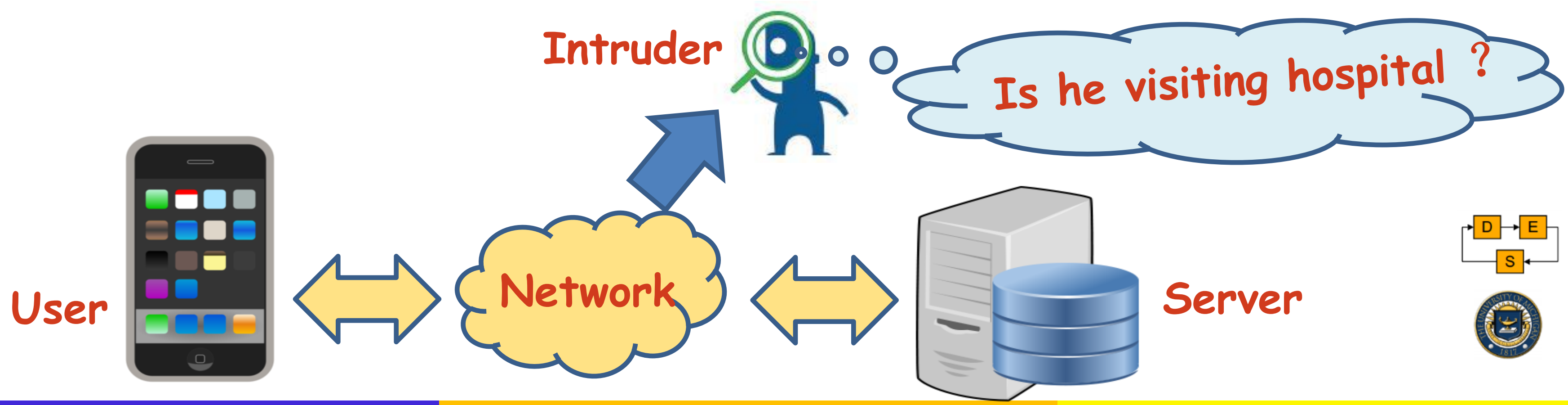


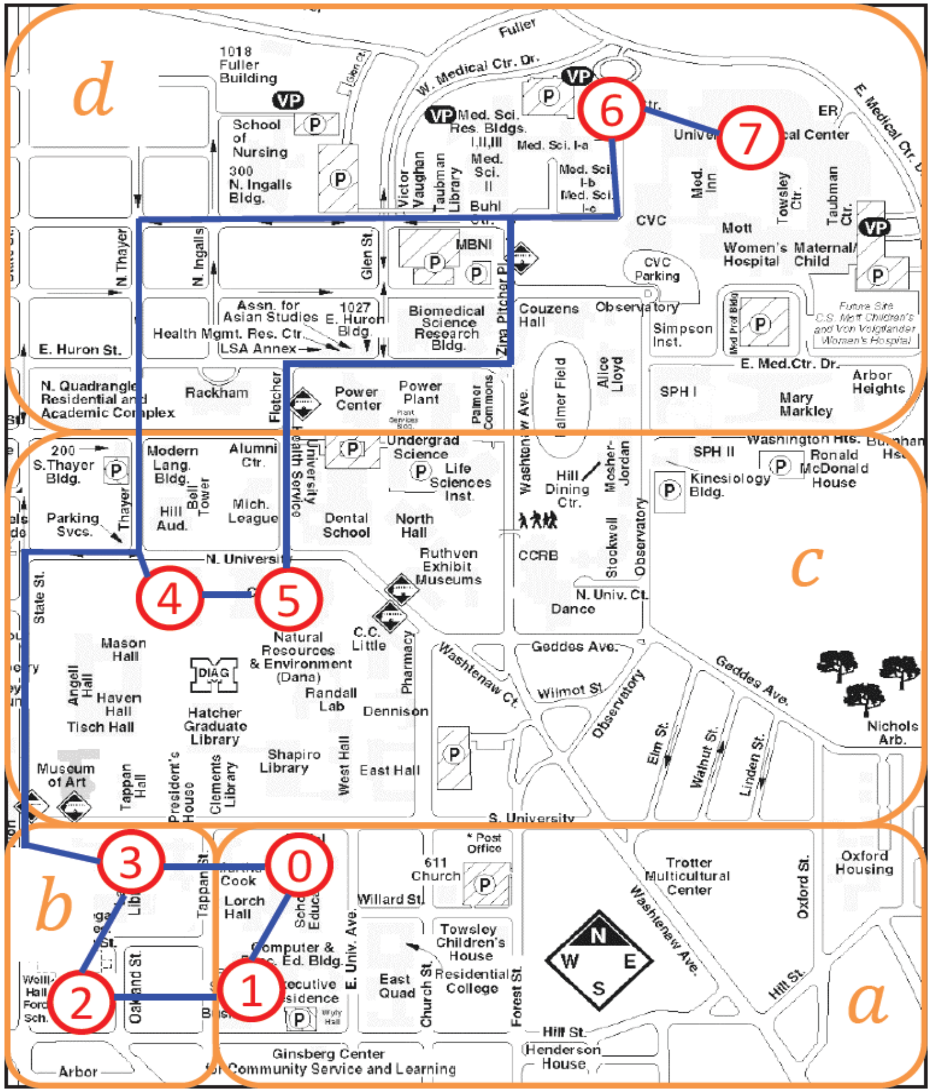
## Location-Based Services

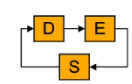
- Provide services to mobile users by exploiting their location information
- Finding nearby restaurants, tracking users' running routes, etc.
- May not be secure!

## Attack Model for the Intruder

- Is located at the LBS server
- Has mobility patterns of users
- Receives location information in LBS queries

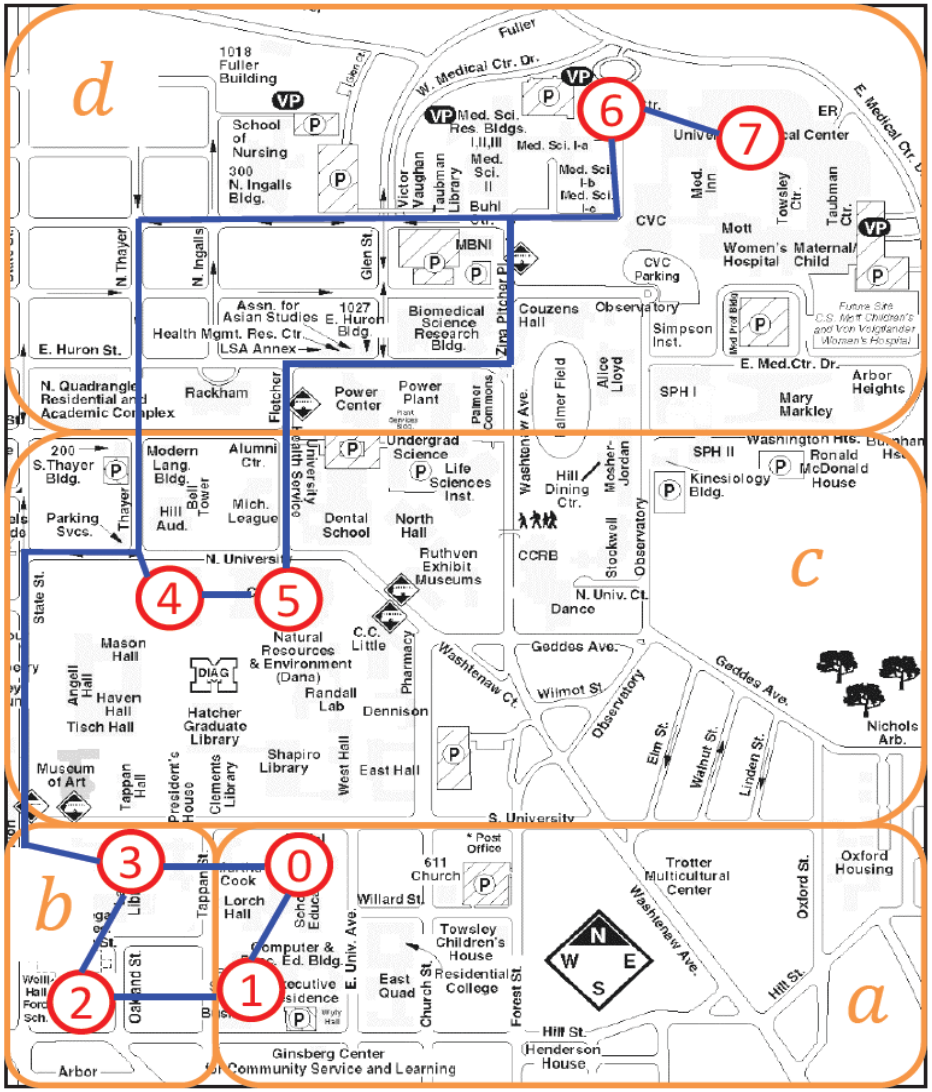


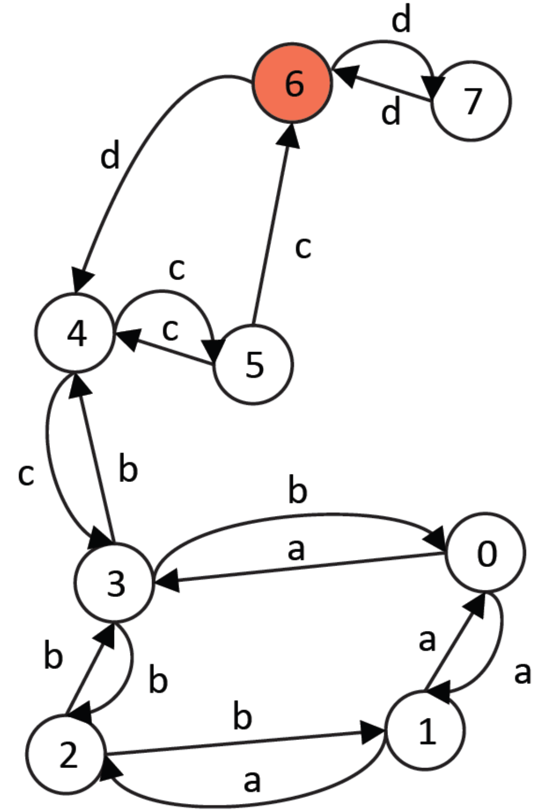



Y.-C. Wu, K. Sankararaman and S. Lafortune. "Ensuring privacy in location-based services: An approach based on opacity enforcement." WODES14, 47.2 (2014): 33-38.

# Application of Opacity: Location-Based Services




- Is state 6 (cancer center) opaque?
- No! Consider string *cdd*

Y.-C. Wu, K. Sankararaman and S. Lafortune. "Ensuring privacy in location-based services: An approach based on opacity enforcement." WODES14, 47.2 (2014): 33-38.

# Recent Advances on Fault Diagnosis and Fault Prognosis

**Diagnosability** [Sampath, et al, 1995]

The occurrence of any fault event can be *detected* unambiguously within a finite delay.

**Prognosability** [Genc & Lafortune, 2009, Kumar & Takai, 2011]

The occurrence of any fault event can be *predicted* with no miss-alarm and no false-alarm.

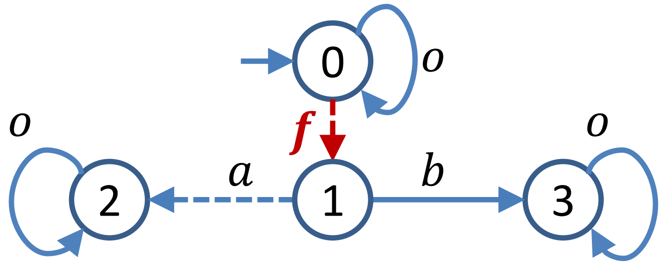
# Recent Advances on Fault Diagnosis and Fault Prognosis

**Diagnosability** [Sampath, et al, 1995]

The occurrence of any fault event can be *detected* unambiguously within a finite delay.

**Prognosability** [Genc & Lafortune, 2009, Kumar & Takai, 2011]

The occurrence of any fault event can be *predicted* with no miss-alarm and no false-alarm.



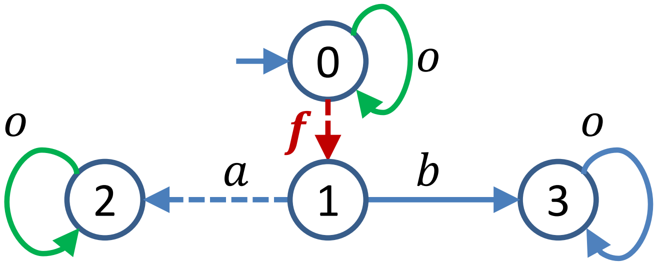
# Recent Advances on Fault Diagnosis and Fault Prognosis

**Diagnosability** [Sampath, et al, 1995]

The occurrence of any fault event can be *detected* unambiguously within a finite delay.

**Prognosability** [Genc & Lafortune, 2009, Kumar & Takai, 2011]

The occurrence of any fault event can be *predicted* with no miss-alarm and no false-alarm.



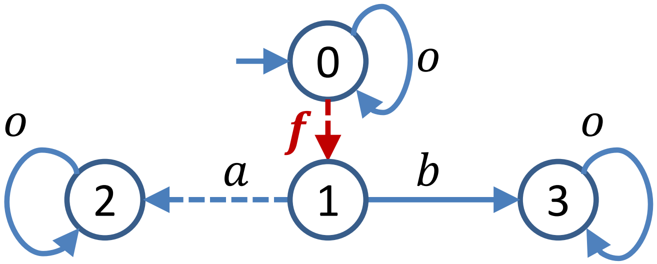
# Recent Advances on Fault Diagnosis and Fault Prognosis

**Diagnosability** [Sampath, et al, 1995]

The occurrence of any fault event can be *detected* unambiguously within a finite delay.

**Prognosability** [Genc & Lafortune, 2009, Kumar & Takai, 2011]

The occurrence of any fault event can be *predicted* with no miss-alarm and no false-alarm.




**Not diagnosable if we cannot see event *a***

# Recent Advances on Fault Diagnosis and Fault Prognosis

**Diagnosability** [Sampath, et al, 1995]

The occurrence of any fault event can be *detected* unambiguously within a finite delay.

**Prognosability** [Genc & Lafortune, 2009, Kumar & Takai, 2011]

The occurrence of any fault event can be *predicted* with no miss-alarm and no false-alarm.

## *Recent Advances*

- Diagnosability and observability are equivalent

  - **X. Yin** and S. Lafortune, "Codiagnosability and coobservability under dynamic observations: transformation and verification." ***Automatica***, vol.61, pp. 241-252, 2015. (**Regular Paper**)

- Performance and reliability issue in decentralized fault prognosis

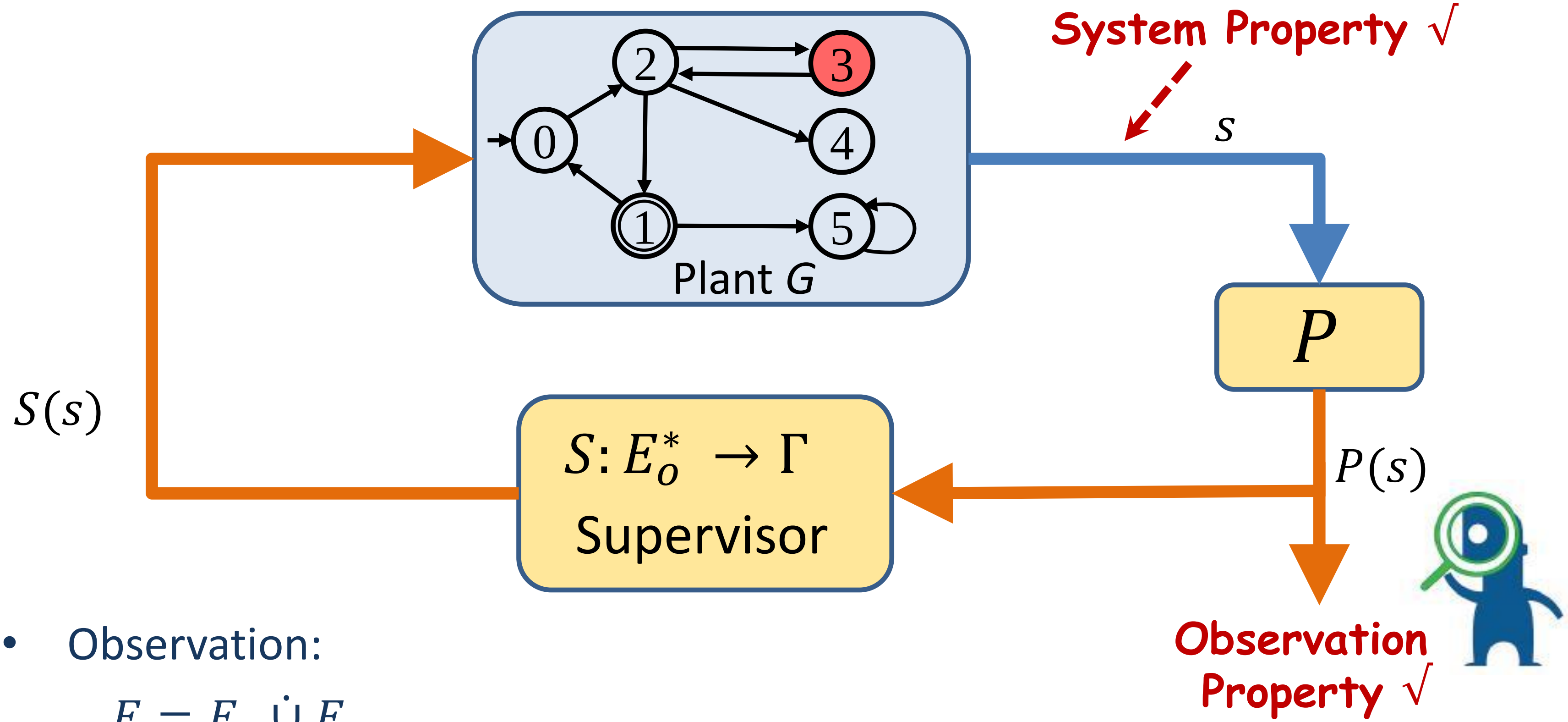
  - **X. Yin** and Z.-J. Li. "Decentralized fault prognosis of discrete event systems with guaranteed performance bound," ***Automatica***, vol.69, pp. 375-379, 2016.
  - **X. Yin** and Z.-J. Li. "Reliable decentralized fault prognosis of discrete-event systems," ***IEEE Trans. Systems, Man, and Cybernetics: Systems***, vol.46, no.8, 2016.

- **What if Verification Fails?**
  - For example: LBS example

- **What if Verification Fails?**
  - For example: LBS example

- **Synthesis!**
  - **Synthesis of *supervisory control* strategies**
  - **Synthesis of *sensor activation* strategies**

- **Property Enforcement via Supervisory Control**




- Observation:

  $E = E_o \dot{\cup} E_{uo}$

- Supervisor:

  $E = E_c \dot{\cup} E_{uc}$, $E_{uc}$ uncontrollable events (environment)

  Disable events in $E_c$ based on its observations

- **System Property**
  - Safety: never visited illegal states
  - Non-blockingness: no deadlocks or livelocks

# Formal Specifications

- **System Property**
  - Safety: never visited illegal states
  - Non-blockingness: no deadlocks or livelocks

- **Observation Property**
  - Opacity, Diagnosability, Prognosability, Observability

# Formal Specifications

- **System Property**
  - Safety: never visited illegal states
  - Non-blockingness: no deadlocks or livelocks

- **Observation Property**
  - Opacity, Diagnosability, Prognosability, Observability

- **Maximal Permissiveness**
  - Optimality criterion is set inclusion.
    Only disable an event if absolutely necessary

# Formal Specifications

- **System Property**
  - Safety: never visited illegal states
  - Non-blockingness: no deadlocks or livelocks

Standard Supervisory Control [Ramadge & Wonham, 1980s]

- **Observation Property**
  - Opacity, Diagnosability, Prognosability, Observability

- **Maximal Permissiveness**
  - Optimality criterion is set inclusion.
    Only disable an event if absolutely necessary

# Property Enforcing Supervisory Control Problem

| Property | Safety | Opacity | Diagnosability | Detectability | Anonymity | Attractability |
|---|---|---|---|---|---|---|
| **Previous Work** | [1]-[3] | [4],[5] | [6] | [7] | None | [8] |
| **Previous Assumptions** | None | $E_a \subseteq E_o$<br>$E_c \subseteq E_o$ | $E_c \subseteq E_o$ | $E_c \subseteq E_o$ | N/A | $E_c \subseteq E_o$ |


[1] [Lin and Wonham, 1988]
[2] [Cieslak et al., 1988]
[3] [Ben Hadj-Alouane et al., 1996]
[4] [Dubreil et al., 2010]
[5] [Saboori and Hadjicostis, 2011]
[6] [Sampath et al., 1998]
[7] [Shu and Lin, 2013]
[8] [Schmidt and Breindl, 2014]

# Property Enforcing Supervisory Control Problem

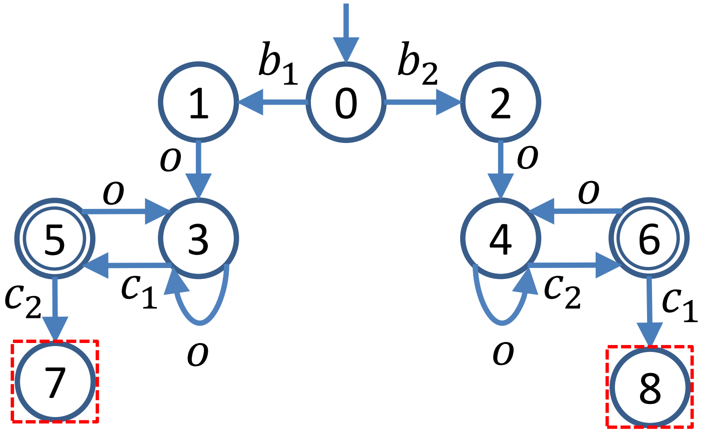
| Property | Safety | Opacity | Diagnosability | Detectability | Anonymity | Attractability |
|---|---|---|---|---|---|---|
| Previous Work | [1]-[3] | [4],[5] | [6] | [7] | None | [8] |
| Previous Assumptions | None | $E_a \subseteq E_o$ $E_c \subseteq E_o$ | $E_c \subseteq E_o$ | $E_c \subseteq E_o$ | N/A | $E_c \subseteq E_o$ |
| Our Assumption | None | $E_a = E_o$ | None | None | $E_a = E_o$ | None |


[1] [Lin and Wonham, 1988]
[2] [Cieslak et al., 1988]
[3] [Ben Hadj-Alouane et al., 1996]
[4] [Dubreil et al., 2010]
[5] [Saboori and Hadjicostis, 2011]
[6] [Sampath et al., 1998]
[7] [Shu and Lin, 2013]
[8] [Schmidt and Breindl, 2014]


## A Uniform Approach

**X. Yin** and S. Lafortune, "A uniform approach for synthesizing property-enforcing supervisors for partially-observed DES." ***IEEE Transactions Automatic Control***, vol.61, no.8, 2016. (**Regular Paper**)

- Information State: a set of states; $I = 2^X$.
- State Estimate: all possible states consistent with observation

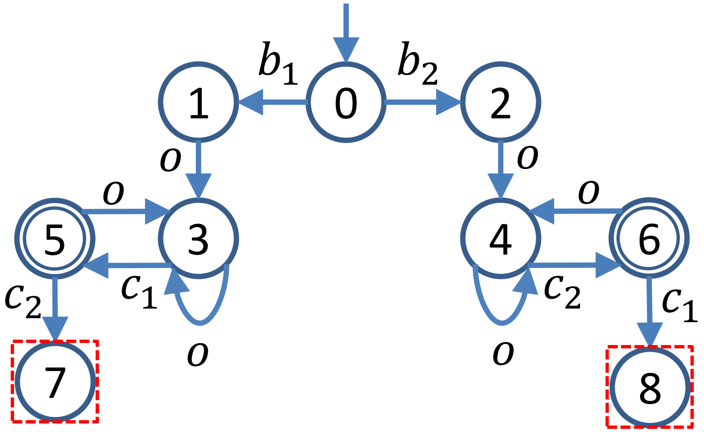



$E_c = \{c_1, c_2\}, E_o = \{o\}$

- Supervisor $S$ disables nothing
- $I(o) = \{3,4\}, I(oo) = \{5,6\}$

# A Uniform Approach for Property Enforcement

- Information State: a set of states; $I = 2^X$.
- State Estimate: all possible states consistent with observation
- Information-State Based Property: $\varphi: 2^X \to \{0,1\}$
- It contains: safety, opacity, diagnosability, detectability, attractability, anonimity, etc.



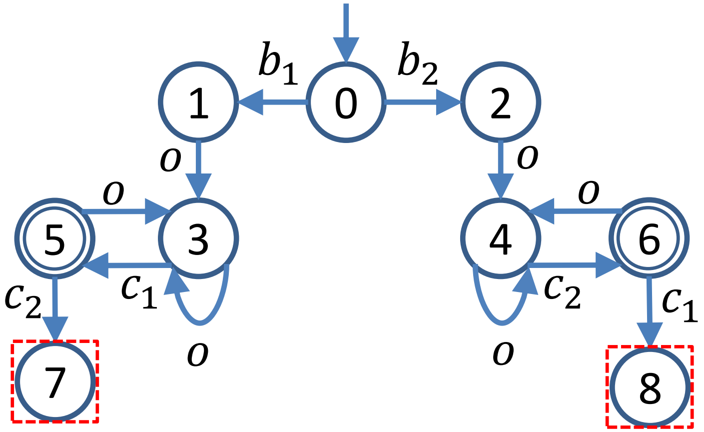

$E_c = \{c_1, c_2\}, E_o = \{o\}$

- Supervisor $S$ disables nothing
- $I(o) = \{3,4\}, I(oo) = \{5,6\}$

# A Uniform Approach for Property Enforcement

- Information State: a set of states; $I = 2^X$.
- State Estimate: all possible states consistent with observation
- Information-State Based Property: $\varphi: 2^X \rightarrow \{0,1\}$
- It contains: safety, opacity, diagnosability, detectability, attractability, anonimity, etc.

$$\varphi(i) = 0 \Leftrightarrow i \cap BAD \neq \emptyset$$



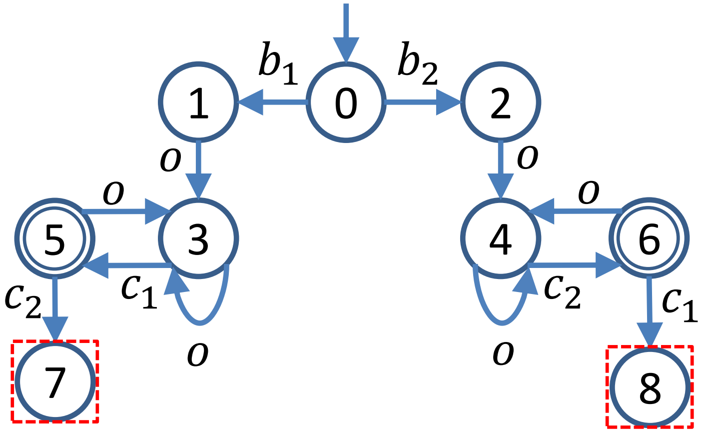

$E_c = \{c_1, c_2\}, E_o = \{o\}$

- Supervisor $S$ disables nothing
- $I(o) = \{3,4\}, I(oo) = \{5,6\}$

# A Uniform Approach for Property Enforcement

- Information State: a set of states; $I = 2^X$.
- State Estimate: all possible states consistent with observation
- Information-State Based Property: $\varphi: 2^X \to \{0,1\}$
- It contains: safety, opacity, diagnosability, detectability, attractability, anonimity, etc.
- **Key Result:**

  Any IS-based property can be enforced by an IS-based supervisor




$E_c = \{c_1, c_2\}, E_o = \{o\}$

- Supervisor $S$ disables nothing
- $I(o) = \{3,4\}, I(oo) = \{5,6\}$

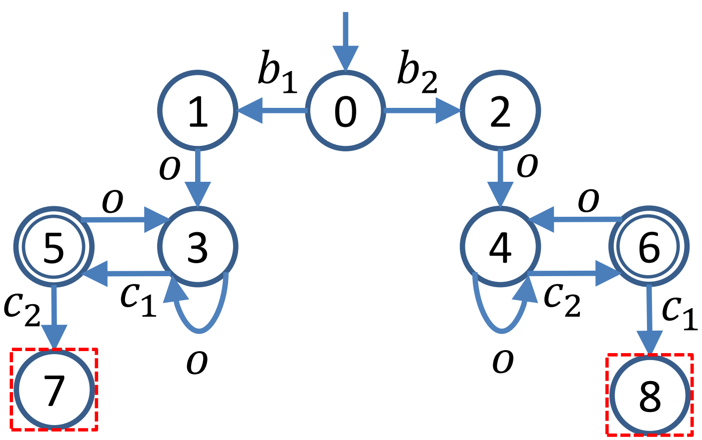
# A Uniform Approach for Property Enforcement

- **Basic Idea:** Construct an information structure that captures all possible controlled behaviors of the system
- **All Inclusive Controller:**

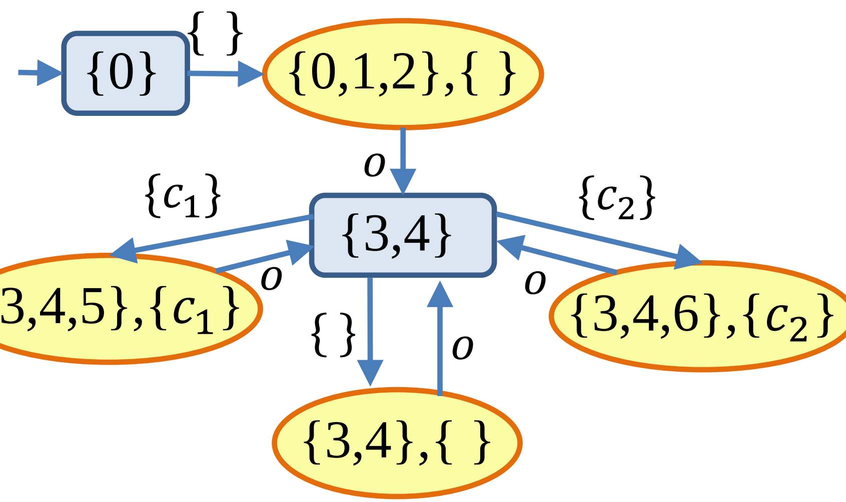
  - A "Game" between environment and controller

  - Two kinds of states: Y-states and Z-states

  - It embeds (infinite many) solutions in its finite structure

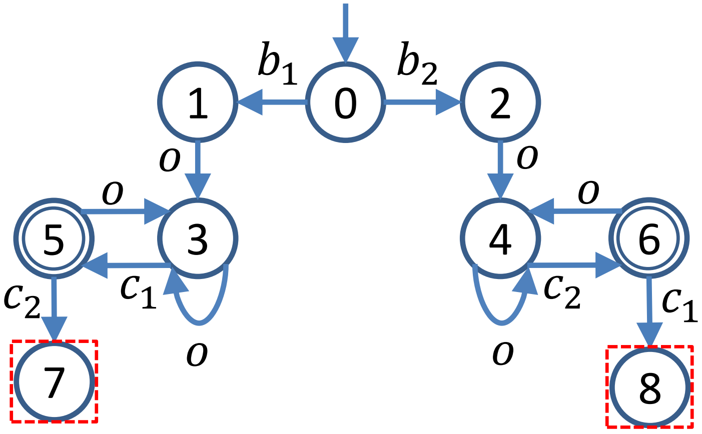



$$E_c = \{c_1, c_2\}, E_o = \{o\}$$

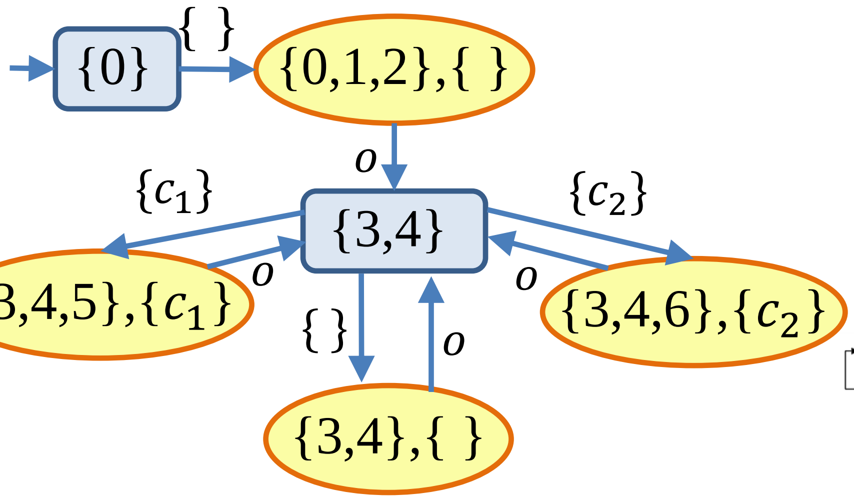


# A Uniform Approach for Property Enforcement

- **Basic Idea:** Construct an information structure that captures all possible behaviors
- **All Inclusive Controller:**
  - A "Game" between environment and controller
  - Two kinds of states: Y-states and Z-states
  - It embeds (infinite many) solutions in its finite structure
- **Synthesis:** Pick a *locally maximal* control decision at each Y-state




$E_c = \{c_1, c_2\}, E_o = \{o\}$



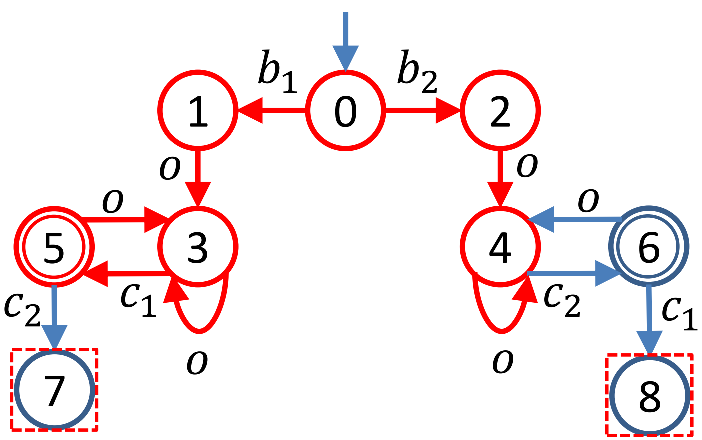
# A Uniform Approach for Property Enforcement

- **Basic Idea:** Construct an information structure that captures all possible behaviors
- ***All Inclusive Controller:***

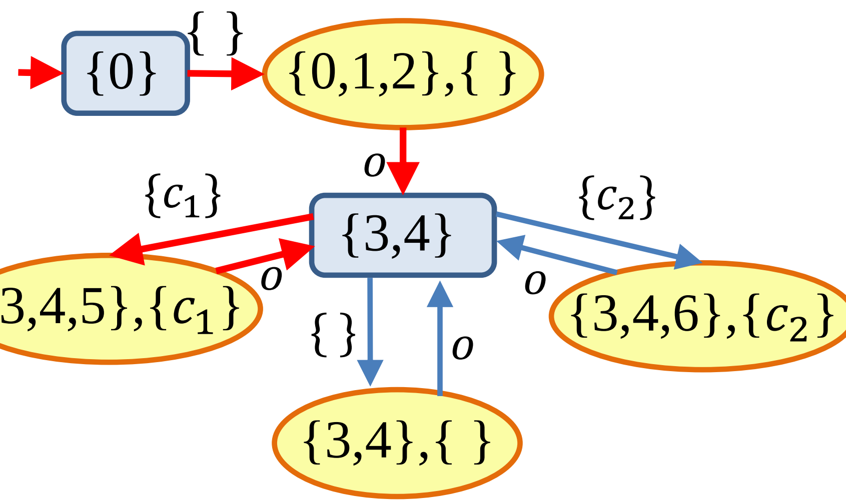
  - A "Game" between environment and controller

  - Two kinds of states: Y-states and Z-states

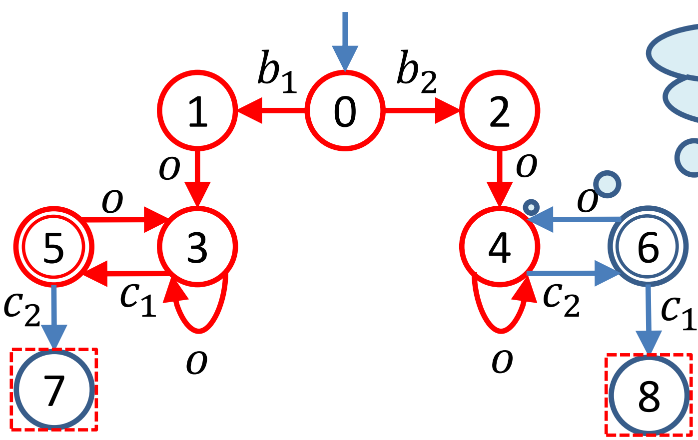
  - It embeds (infinite many) solutions in its finite structure
- **Synthesis:** Pick a *locally maximal* control decision at each Y-state



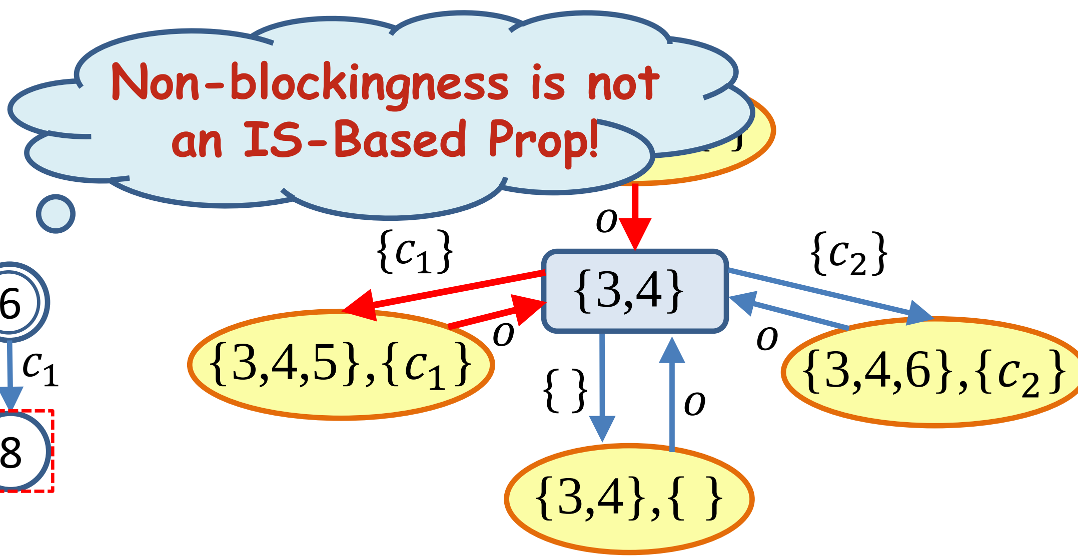

$E_c = \{c_1, c_2\}, E_o = \{o\}$



- **Basic Idea:** Construct an information structure that captures all possible behaviors
- **All Inclusive Controller:**
  - A "Game" between environment and controller
  - Two kinds of states: Y-states and Z-states
  - It embeds (infinite many) solutions in its finite structure
- **Synthesis:** Pick a *locally maximal* control decision at each Y-state




$E_c = \{c_1, c_2\}, E_o = \{o\}$



# Standard Supervisory Control Problem

| | Safety | Safe+Max | Safe+NB | Safe+NB+Max |
|---|---|---|---|---|
| **Centralized Upper Bound** | [1],[2],[3] | [4] | [5] | OPEN |
| **Centralized Range** | [1],[2],[3] | OPEN | OPEN | OPEN |
| **Decentralized Upper Bound** | [2],[6] | OPEN | Undecidable [7],[8] | Undecidable |
| **Decentralized Range** | [2],[6] | OPEN | Undecidable | Undecidable |


[1] [Lin and Wonham, 1988]
[2] [Cieslak et al., 1988]
[3][Rudie and Wonham, 1990]
[4][Ben Hadj-Alouane et al., 1996]
[5][Yoo and Lafortune, 2006]
[6][Rudie and Wonham, 1992]
[7][Tripakis, 2004]
[8][Thistle, 2005]

# Standard Supervisory Control Problem

| | Safety | Safe+Max | Safe+NB | Safe+NB+Max |
|---|---|---|---|---|
| Centralized Upper Bound | [1],[2],[3] | [4] | [5] | Solved √ |
| Centralized Range | [1],[2],[3] | Solved √ | OPEN | OPEN |
| Decentralized Upper Bound | [2],[6] | OPEN | Undecidable [7],[8] | Undecidable |
| Decentralized Range | [2],[6] | OPEN | Undecidable | Undecidable |


[1] [Lin and Wonham, 1988]
[2] [Cieslak et al., 1988]
[3][Rudie and Wonham, 1990]
[4][Ben Hadj-Alouane et al., 1996]
[5][Yoo and Lafortune, 2006]
[6][Rudie and Wonham, 1992]
[7][Tripakis, 2004]
[8][Thistle, 2005]


## Recent Advances

**X. Yin** and S. Lafortune, "Synthesis of maximally permissive supervisors for partially observed DES." ***IEEE Transactions Automatic Control***, vol.61, no.5, 2016. (**Regular Paper**)

**X. Yin** and S. Lafortune. "Synthesis of maximally-permissive supervisors for the range control problem," ***IEEE Transactions Automatic Control***, under review, 2016. (**Regular Paper**)

# Standard Supervisory Control Problem

| | Safety | Safe+Max | Safe+NB | Safe+NB+Max |
|---|---|---|---|---|
| Centralized Upper Bound | [1],[2],[3] | [4] | [5] | Solved √ |
| Centralized Range | [1],[2],[3] | Solved √ | OPEN | OPEN |
| Decentralized Upper Bound | [2],[6] | OPEN | Undecidable [7],[8] | Undecidable |
| Decentralized Range | [2],[6] | OPEN | Undecidable | Undecidable |


[1] [Lin and Wonham, 1988]
[2] [Cieslak et al., 1988]
[3][Rudie and Wonham, 1990]
[4][Ben Hadj-Alouane et al., 1996]
[5][Yoo and Lafortune, 2006]
[6][Rudie and Wonham, 1992]
[7][Tripakis, 2004]
[8][Thistle, 2005]

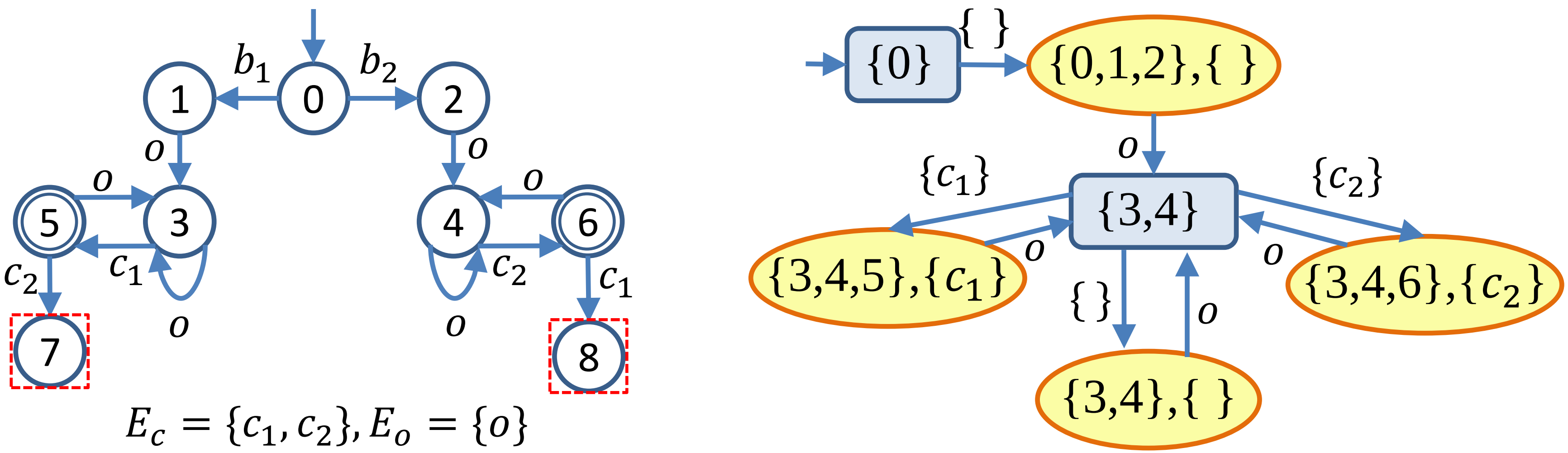

## Recent Advances

**X. Yin** and S. Lafortune, "Synthesis of maximally permissive supervisors for partially observed DES." ***IEEE Transactions Automatic Control***, vol.61, no.5, 2016. (**Regular Paper**)

**X. Yin** and S. Lafortune. "Synthesis of maximally-permissive supervisors for the range control problem," ***IEEE Transactions Automatic Control***, under review, 2016. (**Regular Paper**)

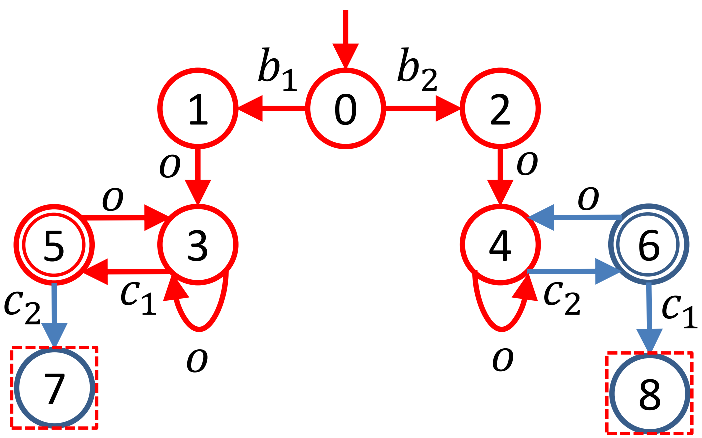
# Non-blocking Control Problem

- **Observation:** $2^X$ is not sufficient to make a decision

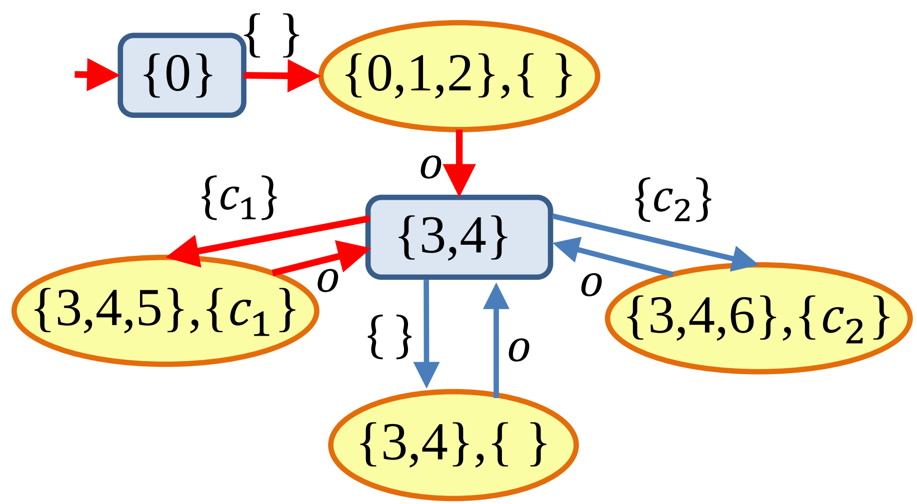


# Non-blocking Control Problem

- **Observation:** $2^X$ is not sufficient to make a decision

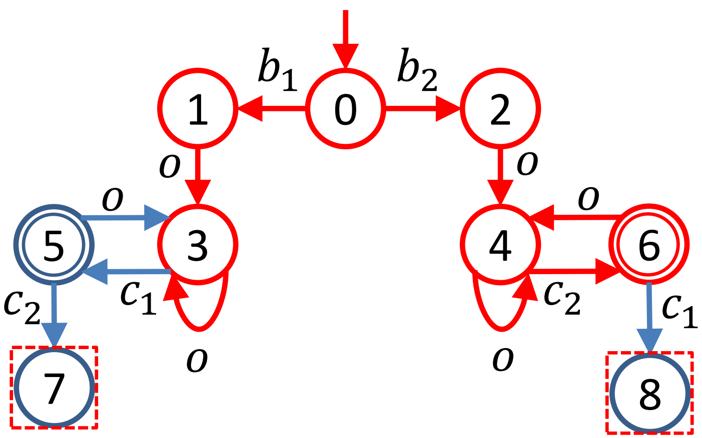



$E_c = \{c_1, c_2\}, E_o = \{o\}$

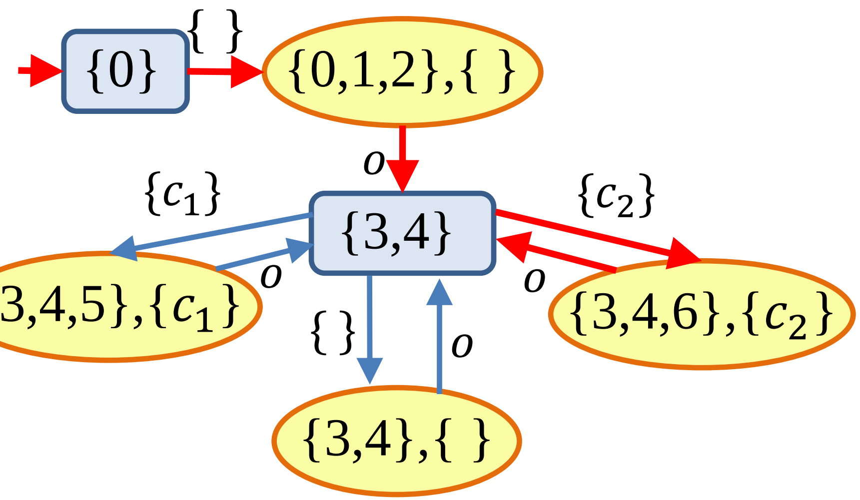


# Non-blocking Control Problem

- **Observation:** $2^X$ is not sufficient to make a decision

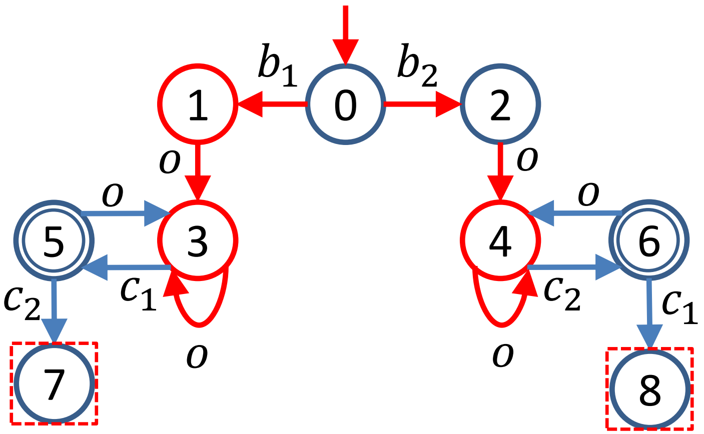


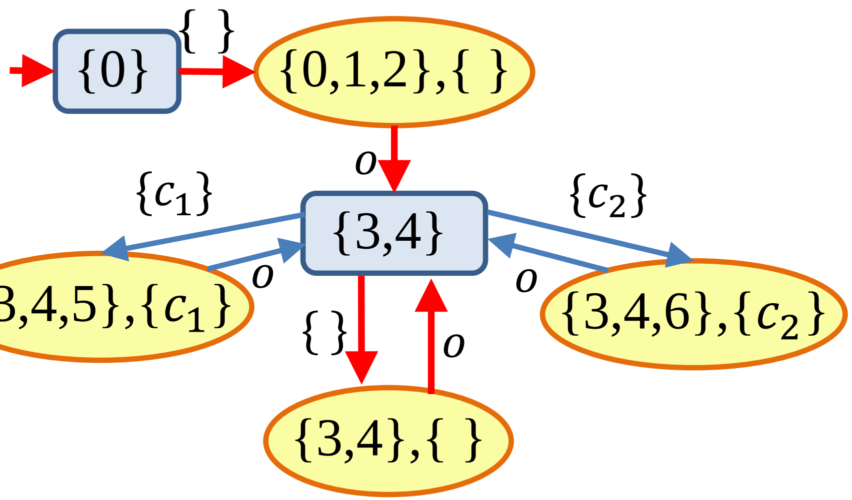

$$E_c = \{c_1, c_2\}, E_o = \{o\}$$



# Non-blocking Control Problem

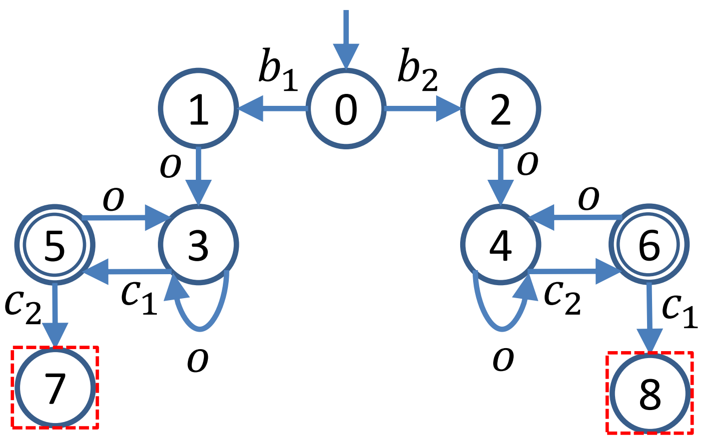
- **Observation:** $2^X$ is not sufficient to make a decision



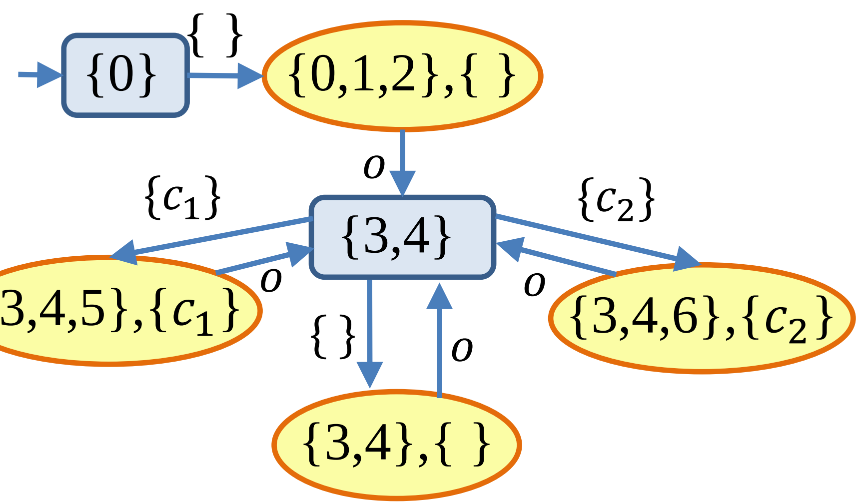



# Non-blocking Control Problem

- **Observation:** $2^X$ is not sufficient to make a decision
- **Basic Idea:** unfold the solution space until it converges
- **Key Result:** We need additional, but finite, information

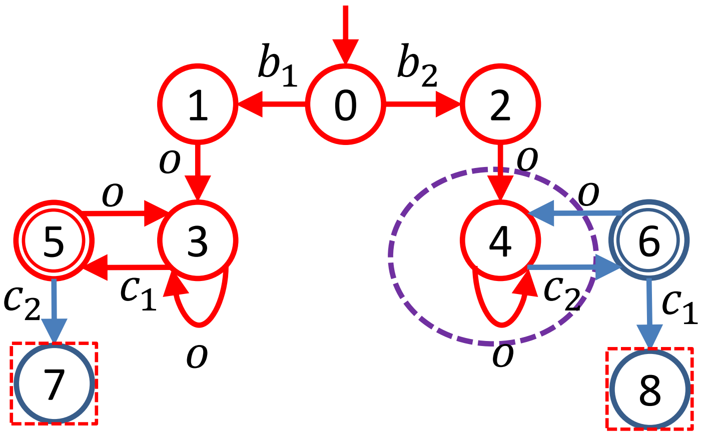



$$E_c = \{c_1, c_2\}, E_o = \{o\}$$

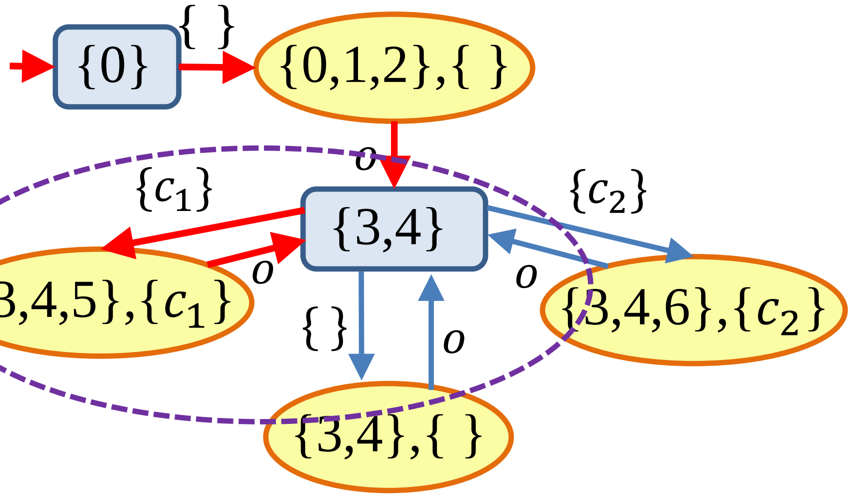


# Non-blocking Control Problem

- **Observation:** $2^X$ is not sufficient to make a decision
- **Basic Idea:** unfold the solution space until it converges
- **Key Result:** We need additional, but finite, information

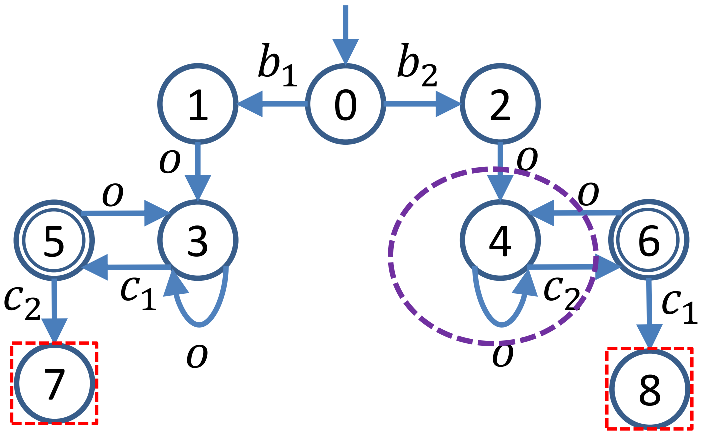



$E_c = \{c_1, c_2\}, E_o = \{o\}$

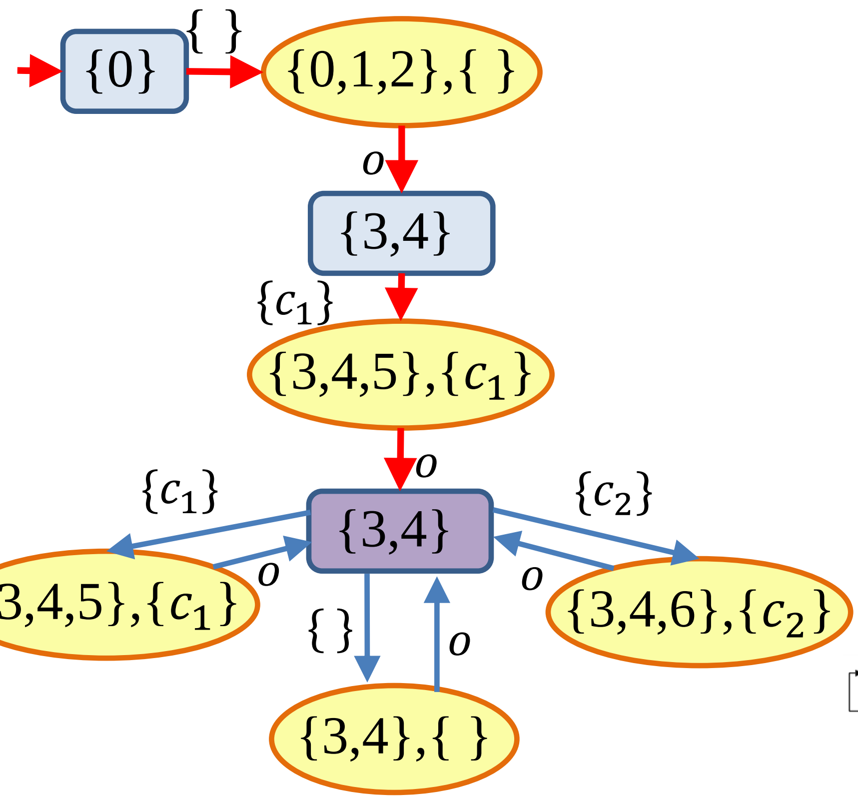


# Non-blocking Control Problem

- **Observation:** $2^X$ is not sufficient to make a decision
- **Basic Idea:** unfold the solution space until it converges
- **Key Result:** We need additional, but finite, information

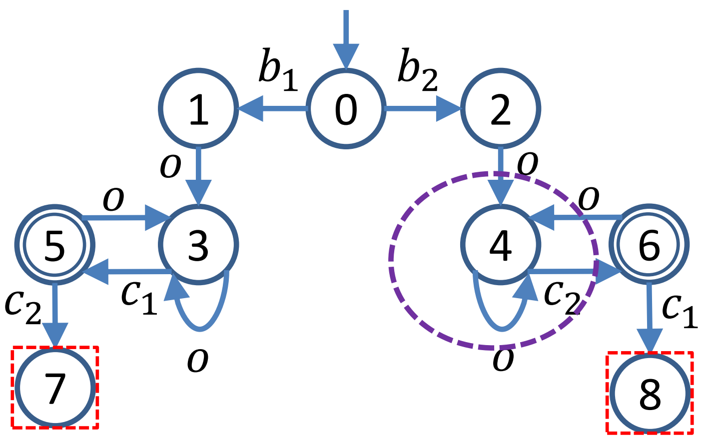



$E_c = \{c_1, c_2\}, E_o = \{o\}$

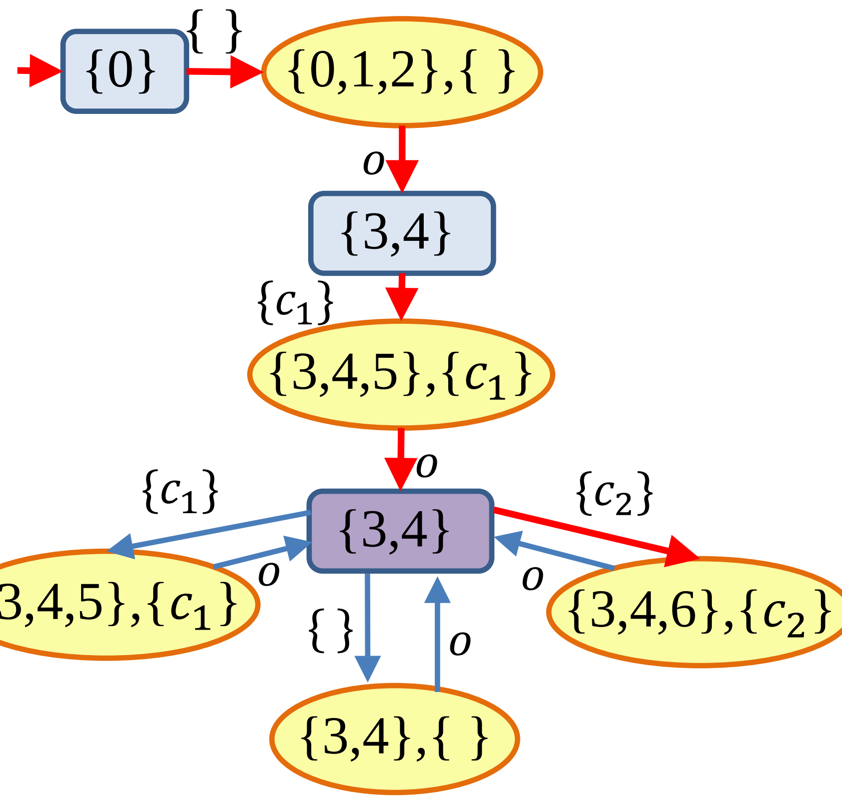


# Non-blocking Control Problem

- **Observation:** $2^X$ is not sufficient to make a decision
- **Basic Idea:** unfold the solution space until it converges
- **Key Result:** We need additional, but finite, information

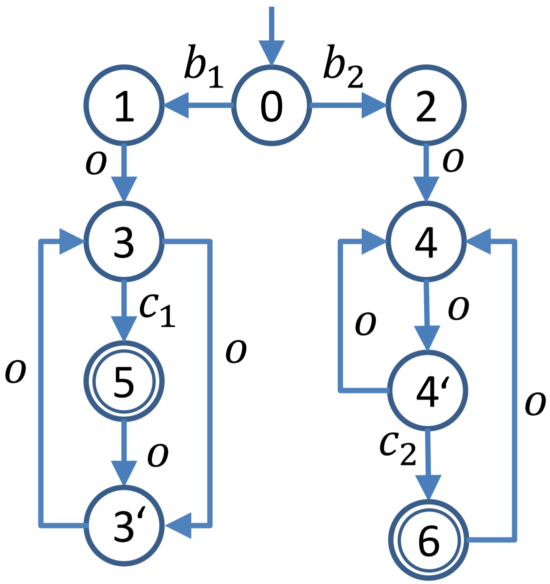


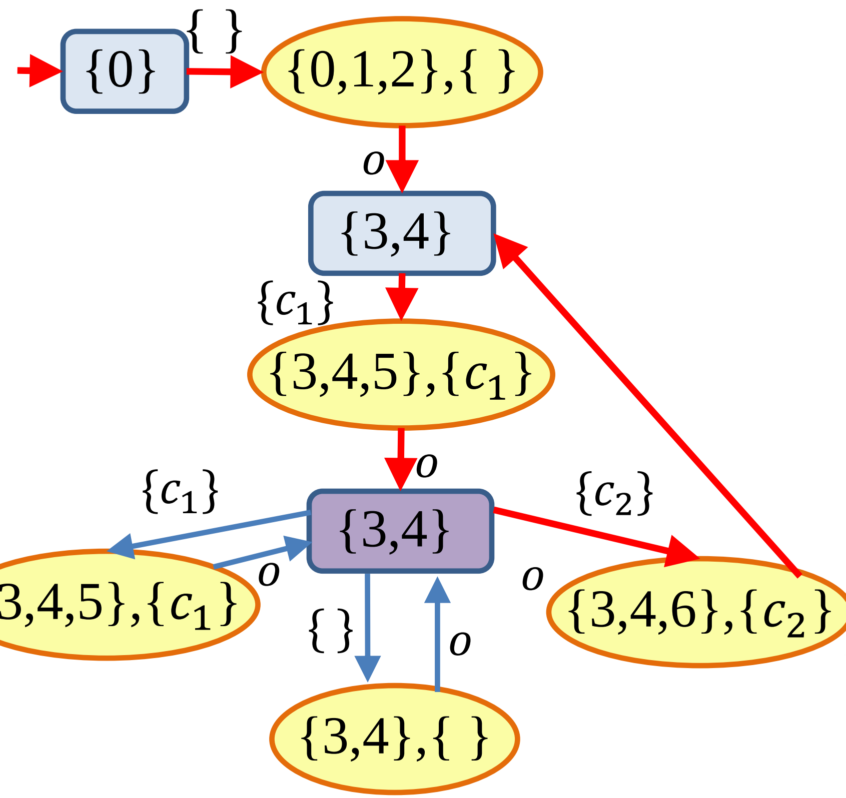

$E_c = \{c_1, c_2\}, E_o = \{o\}$



# Non-blocking Control Problem

- **Observation:** $2^X$ is not sufficient to make a decision
- **Basic Idea:** unfold the solution space until it converges
- **Key Result:** We need additional, but finite, information



# Centralized Sensor Activation Problem

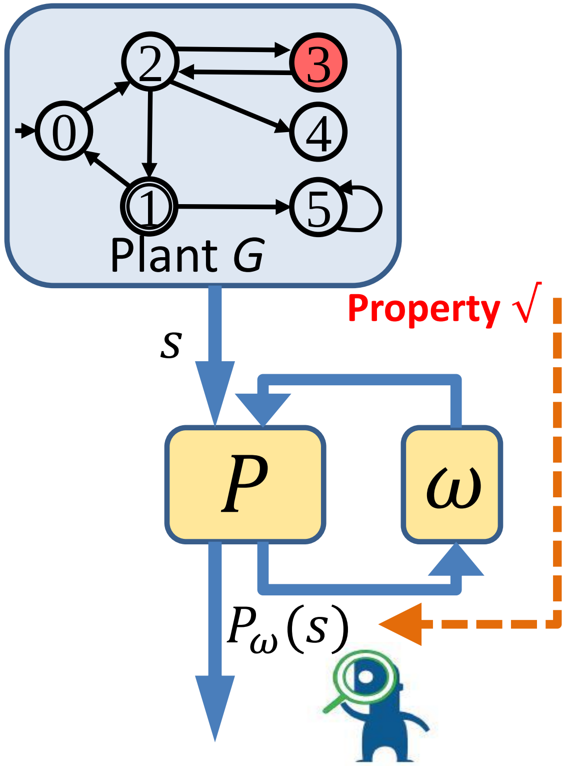
## Sensor activation policy

A function that determines which events to monitor next

## Dynamic Sensor Activation Problem

Find a sensor activation policy $\omega$ such that

- some property can be guaranteed
- It is optimal: numerical (average cost) or logical (set inclusion)



# Centralized Sensor Activation Problem

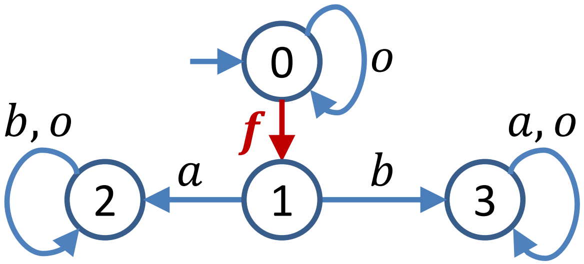
**Sensor activation policy**

A function that determines which events to monitor next

**Dynamic Sensor Activation Problem**

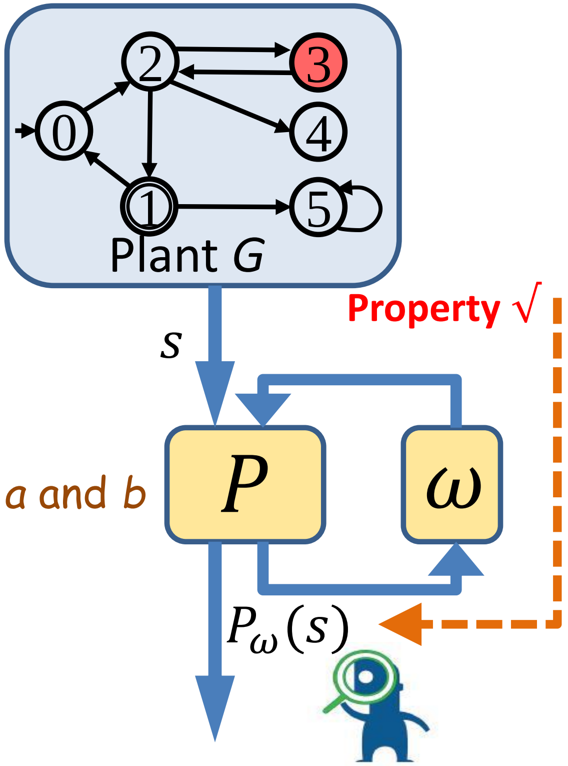
Find a sensor activation policy $\omega$ such that

- some property can be guaranteed
- It is optimal: numerical (average cost) or logical (set inclusion)




A Fault Diagnosis Problem

- **Static Sensors:** always observe $a$ and $b$



# Centralized Sensor Activation Problem

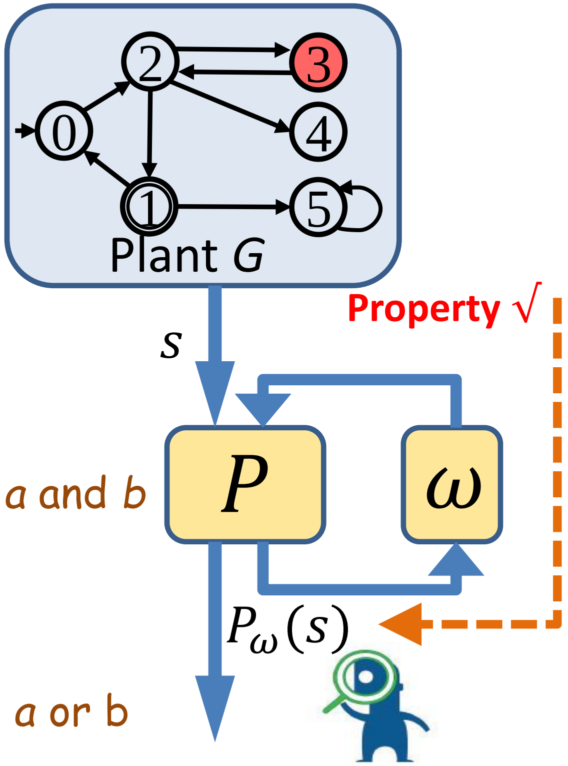
## Sensor activation policy

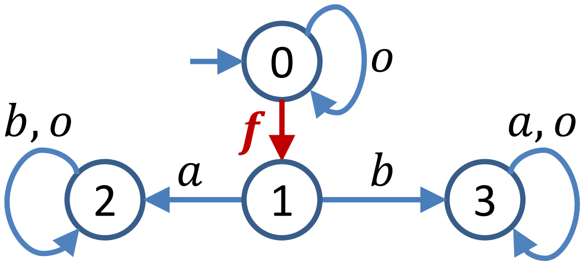
A function that determines which events to monitor next

## Dynamic Sensor Activation Problem

Find a sensor activation policy $\omega$ such that

- some property can be guaranteed
- It is optimal: numerical (average cost) or logical (set inclusion)




A Fault Diagnosis Problem

- **Static Sensors:** always observe $a$ and $b$
- **Dynamic Sensors:**
  - observe both $a$ and $b$ initially
  - turn off all sensors after seeing $a$ or b

# Centralized Sensor Activation Problem

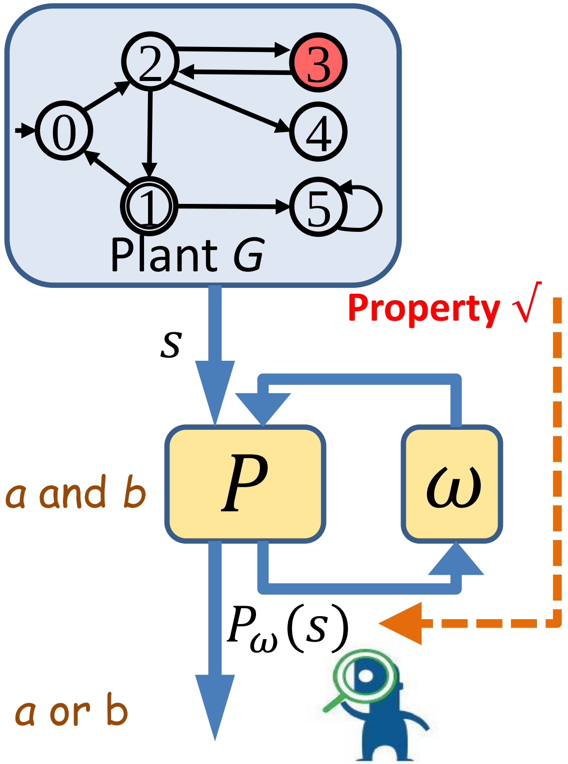
## Sensor activation policy

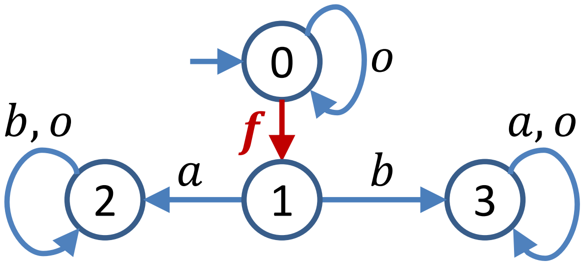
A function that determines which events to monitor next

## Dynamic Sensor Activation Problem

Find a sensor activation policy $\omega$ such that

- some property can be guaranteed
- It is optimal: numerical (average cost) or logical (set inclusion)




A Fault Diagnosis Problem

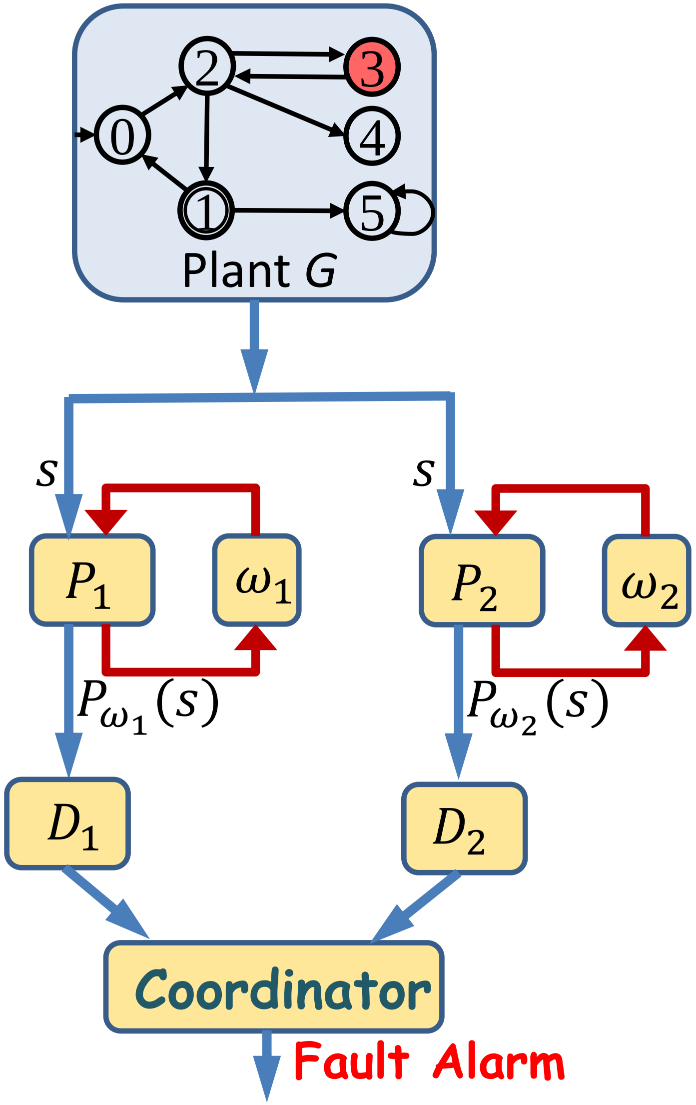
- **Static Sensors:** always observe *a* and *b*
- **Dynamic Sensors:**
  - observe both *a* and *b* initially
  - turn off all sensors after seeing a or b

## Recent Advances

- A general approach for solving sensor activation problem
- A new structure called the Most Permissive Observer
- A minimal sensor activation policy can be synthesized from the MPO

**X. Yin** and S. Lafortune. "A general approach for solving dynamic sensor activation problems for a class of properties," in ***54th IEEE Conference on Decision and Control***, pp. 3610-3615, 2015.

# Decentralized Sensor Activation Problem




## Decentralized Diagnosis Problem

- Large-scale systems
- Plant is monitored by multiple agents

## Synthesis Problem

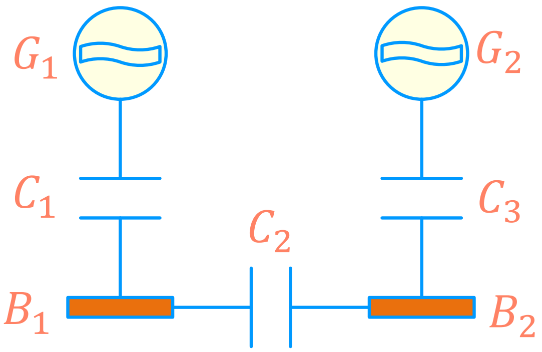
- Synthesis of local sensor activation strategies for each agent such that they are diagnose the fault as a group

## Solution Approach

- Person-by-person approach
- Iteration converge finitely
- It is an optimal solution

**X. Yin** and S. Lafortune. "Minimization of sensor activation in decentralized fault diagnosis of discrete event systems," in ***54th IEEE Conference on Decision and Control***, pp. 1014-1019, 2015

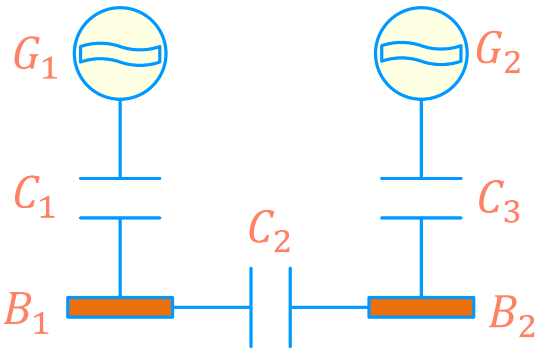
# Apply Synthesis Techniques to Vehicular Electrical Power Systems




**Assumption**

- Generators cannot fail at the same time
- Only one failure/recovery occurs within $T_{max}$
- A control action takes time $t_{trf}$

# Apply Synthesis Techniques to Vehicular Electrical Power Systems

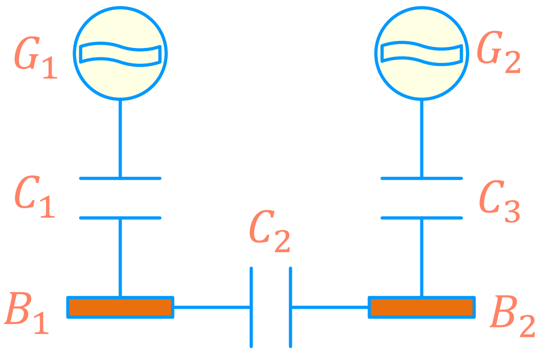



## Assumption

- Generators cannot fail at the same time
- Only one failure/recovery occurs within $T_{max}$
- A control action takes time $t_{trf}$

## Specification

- Generators paralleling is not allowed, i.e., no bus should be powered by more than one generators at the same time
- Buses should not be unpowered for more than $T_{max}$

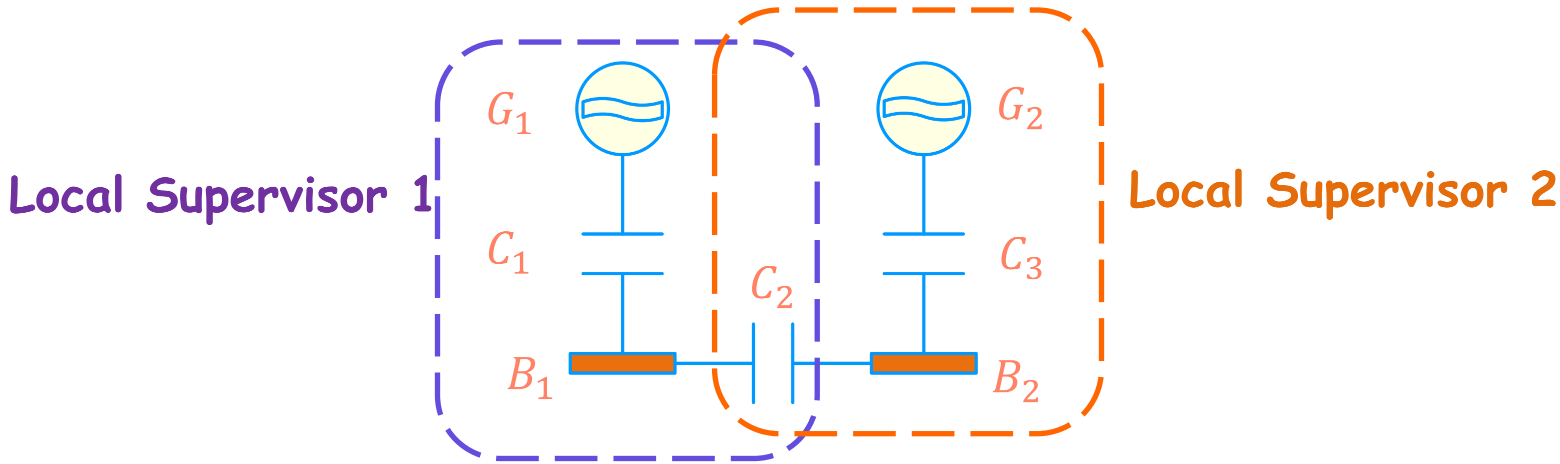



**Assumption**

- Generators cannot fail at the same time
- Only one failure/recovery occurs within $T_{max}$
- A control action takes time $t_{trf}$

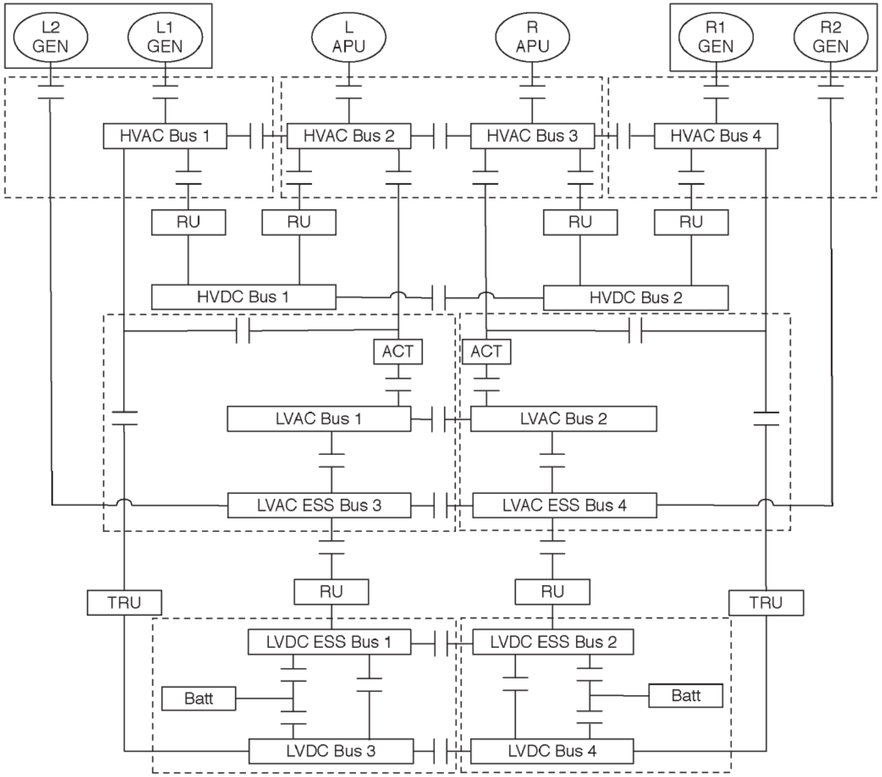
**Specification**

- Generators paralleling is not allowed, i.e., no bus should be powered by more than one generators at the same time
- Buses should not be unpowered for more that $T_{max}$




**Large-Scale System is Decentralized !**

# Apply Synthesis Techniques to Vehicular Electrical Power Systems

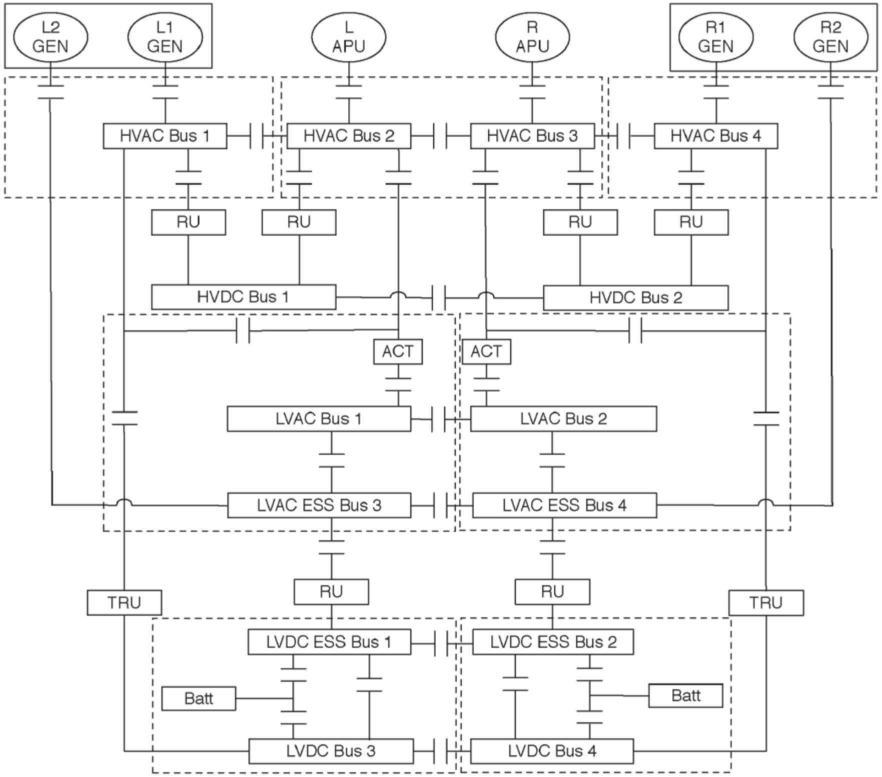



An aircraft EPS: Honeywell Inc. patent

## When the system is huge

- Safety-critical system
- Intuition is hard to handle
- Need formal synthesis techniques!

# Apply Synthesis Techniques to Vehicular Electrical Power Systems




An aircraft EPS: Honeywell Inc. patent

## When the system is huge

- Safety-critical system
- Intuition is hard to handle
- Need formal synthesis techniques!

## Our Results

- Build DES Model: the state-space is already discrete; discretize time
- Apply supervisor synthesis technique developed
- Algorithm implemented by Alloy*, an efficient model finder embedding SAT solver (On going)

# Future Works

## Summary

- Recent Advances on the verification and synthesis of partially-observed DES
- Verification: Opacity, Diagnosability, Prognosability
- Synthesis
  - Supervisory Control Strategies: a uniform approach & non-blockingness
  - Sensor Activation Strategies: centralized/decentralized solutions
- Two Applications: LBS and EPS

# Future Works

## Summary

- Recent Advances on the verification and synthesis of partially-observed DES
- Verification: Opacity, Diagnosability, Prognosability
- Synthesis

  - Supervisory Control Strategies: a uniform approach & non-blockingness

  - Sensor Activation Strategies: centralized/decentralized solutions
- Two Applications: LBS and EPS

## Future Directions

- More Properties: Temporal Logic, LTL, CTL*…, (Bi)Simulation
- More Models: Petri nets, Stochastic DES (Markov chains)
- More Applications to Cyber-Physical Systems:
  SCADA systems (PLC), Intelligent transportation systems, Cyber-security

# References


1. **X. Yin** and S. Lafortune, "Decentralized supervisory control with intersection-based architecture." ***IEEE Trans. Automatic Control***, to appear, 2017.
2. **X. Yin** and S. Lafortune, "A uniform approach for synthesizing property-enforcing supervisors for partially-observed discrete-event systems." ***IEEE Trans. Automatic Control***, vol.61, no.8, 2016. (**Regular Paper**)
3. **X. Yin** and S. Lafortune, "Synthesis of maximally permissive supervisors for partially observed discrete event systems." ***IEEE Trans. Automatic Control***, vol.61, no.5, 2016. (**Regular Paper**)
4. **X. Yin** and S. Lafortune, "Codiagnosability and coobservability under dynamic observations: transformation and verification." ***Automatica***, vol.61, pp. 241-252, 2015. (**Regular Paper**)
5. **X. Yin** and Z.-J. Li. "Decentralized fault prognosis of discrete event systems with guaranteed performance bound," ***Automatica***, vol.69, pp. 375-379, 2016.
6. **X. Yin** and Z.-J. Li. "Reliable decentralized fault prognosis of discrete-event systems," ***IEEE Trans. Systems, Man, and Cybernetics: Systems***, vol.46, no.8, 2016.
7. **X. Yin** and S. Lafortune. "Synthesis of maximally-permissive supervisors for the range control problem," ***IEEE Trans. Automatic Control***, under review, 2016.
8. **X. Yin** and S. Lafortune. "On the decidability and complexity of diagnosability for labeled Petri nets," ***IEEE Trans. Automatic Control***, under review, 2016.
9. **X. Yin** and S. Lafortune. "A new approach for the verification of infinite-step and K-step opacity using two-way observer," ***Automatica***, under review, 2016.
10. **X. Yin** and S. Lafortune. "On two-way observer and its application to the verification of infinite-step and K-step opacity," ***13th Int. Workshop on Discrete Event Systems***, 2016.
11. **X. Yin** and S. Lafortune. "Minimization of sensor activation in decentralized fault diagnosis of discrete event systems," in ***54th IEEE Conference on Decision and Control***, pp. 1014-1019, 2015
12. **X. Yin** and S. Lafortune. "A general approach for solving dynamic sensor activation problems for a class of properties," in ***54th IEEE Conference on Decision and Control***, pp. 3610-3615, 2015.