Supervisory Control of Discrete Event Systems

Part 4: State Estimation, Fault Diagnosis, and Opacity

Dr Andrea Paoli
Associate Professor in Robotics and Automation

Loughborough University
Wolfson School of Mechanical, Electrical and Manufacturing Engineering

Fall semester 2026-2027
University of Ferrara

Outline

1 State Estimation and Observer Automata

2 Fault Diagnosis and the Sampath Diagnoser

3 Security, Privacy, and Opacity in DES

4 Conclusions, Review Questions & Self-Assessment

Part 4: Intended Learning Outcomes (ILOs)

1. **Construct Deterministic State Observers**:
   Formalize state uncertainty under sensory masking ($\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$) by constructing observer automata using unobservable reach (ε-reach) and subset construction.
2. **Analyze Observability and Detectability**:
   Evaluate whether a discrete-event system can disambiguate critical physical states and reconstruct its current state trajectory from observed event records ($P(s) \in \Sigma_o^*$).
3. **Synthesize Fault Diagnosers (Sampath Framework)**:
   Build diagnoser automata to detect unobservable failure events ($f \in \Sigma_f \subseteq \Sigma_{uo}$) and formally verify language diagnosability by checking for indeterminate cycles.
4. **Verify Security and System Opacity**:
   Formulate and test privacy guarantees (Current-State, Initial-State, and Language Opacity) against a passive adversary observing through a projection mask.

Section 4.1
State Estimation and Observer Automata

Core Focus:
Reconstructing physical plant states from partial event observation sequences

State Estimation and Observer Automata

A Fundamental Paradigm Shift

In Part 3, the central engineering question was:

- *"Which controllable events must be disabled to enforce safety?"*

In Section 4.1, the focus shifts to **state inference**:

- An external agent monitors the plant via remote telemetry or limited sensors.
- Some transitions execute **silently** without triggering any sensor notification.
- **Core Question:** *"Given an observed sequence $t \in \Sigma_o^*$, what are all possible current physical states of G?"*

Section 4.1 Objectives

- **Information Masking:** Formalize the observation channel via natural projection $P : \Sigma^* \to \Sigma_o^*$.
- **Uncertainty Quantification:** Define current-state estimate $\mathcal{C}(s) \subseteq X$.
- **Unobservable Reach (UR):** Characterize silent drift between sensor triggers.
- **Observer Automaton (G_{obs}):** Build a deterministic state estimator via subset construction.
- **Asymptotic Properties:** Distinguish state observability from detectability.

Sensory Partitioning & Natural Projection

Sensory Partitioning: $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$

The plant alphabet is partitioned based on sensor presence:

- **Observable (Σ_o):** Monitored transitions producing direct telemetry signals (limit switches, optical flags, barcode scans, explicit control commands).
- **Unobservable (Σ_{uo}):** Silent transitions generating no direct sensor trigger (internal state drift, mechanical wear, unmonitored local actions, silent faults).

The Natural Projection (P)

The causal sensor filter $P : \Sigma^* \to \Sigma_o^*$ formalizes telemetry masking:

$$P(\epsilon) = \epsilon$$

$$P(\sigma) = \begin{cases} \sigma & \text{if } \sigma \in \Sigma_o \\ \epsilon & \text{if } \sigma \in \Sigma_{uo} \end{cases}$$

$$P(s\sigma) = P(s)P(\sigma), \quad \forall s \in \Sigma^*,\ \sigma \in \Sigma$$

- For a language $L \subseteq \Sigma^*$, the projected language is:

$$P(L) = \{P(s) \in \Sigma_o^* \mid s \in L\}$$

The Observation Gap: Plant Trace vs. Sensor Record

The Observation Gap

Let $s \in L(G)$ be the true physical execution trace and $t = P(s) \in \Sigma_o^*$ the received record.

- The observer only perceives t, never the internal string s.
- Distinct physical paths collapse to the identical telemetry log:

$$P^{-1}(\{t\}) \cap L(G) = \{s' \in L(G) \mid P(s') = t\}$$

- Multiple internal execution paths become completely indistinguishable to the remote supervisor.

Untimed Silent Interleaving Hazard

Because DES execution is untimed and asynchronous:

- An arbitrary finite sequence of silent events $u \in \Sigma_{uo}^*$ can occur:
  - Prior to the first observed event,
  - In between two consecutive observed events,
  - After the most recent observed event.
- The observer must account for all possible unobservable interleavings consistent with the plant transition graph.

Current-State Uncertainty Set: $\mathcal{C}(s)$

Formal Definition of Uncertainty

Given a physical trace $s \in L(G)$ with observed record $t = P(s) \in \Sigma_o^*$, the **current-state uncertainty set** is:

$$\mathcal{C}(s) := \{x \in X \mid \exists s' \in L(G) : P(s') = P(s) \wedge f(x_0, s') = x\}$$

- $\mathcal{C}(s)$ is a subset of physical plant states: $\mathcal{C}(s) \subseteq X$.
- It bounds where the plant can physically be at the current instant.

Set-Valued Estimation Paradigm

- **Singleton Set ($|\mathcal{C}(s)| = 1$):** Perfect state identification (zero uncertainty).
- **Ambiguity Set ($|\mathcal{C}(s)| > 1$):** Multiple physical states are observationally consistent.
- **Consistency Property:**

$$P(s_1) = P(s_2) \implies \mathcal{C}(s_1) = \mathcal{C}(s_2)$$

The estimate depends solely on the received sensor trace $t = P(s)$, not on the internal trajectory.

Unobservable Reach: Silent State Exploration

Single-State Unobservable Reach

For a state $x \in X$, its **unobservable reach** $\mathrm{UR}(x)$ (or ε-reach) is the set of states reachable via silent traces:

$$\mathrm{UR}(x) := \{x' \in X \mid \exists u \in \Sigma_{uo}^* : f(x, u) = x'\}$$

- **Trivial Reach:** Since $\epsilon \in \Sigma_{uo}^*$ and $f(x, \epsilon) = x$, we always have:

$$x \in \mathrm{UR}(x)$$

- $\mathrm{UR}(x)$ accounts for autonomous drift without sensor activity.

Physical Meaning in Plant Operation

- When the system enters state x, internal physical processes may continue to trigger silent actions.
- The plant moves along unmonitored paths before the next sensor event fires.
- **Telemetry Rule:** Even when no sensor event is detected, the observer knows the physical state belongs to the set $\mathrm{UR}(x)$.

Unobservable Reach: Set Extension & Fixed Point

Extension to State Subsets

For any arbitrary state estimate subset $B \subseteq X$, the operator extends naturally as:

$$\mathrm{UR}(B) := \bigcup_{x \in B} \mathrm{UR}(x)$$

- **Idempotence:** $\mathrm{UR}(\mathrm{UR}(B)) = \mathrm{UR}(B)$
- **Monotonicity:** $B_1 \subseteq B_2 \implies \mathrm{UR}(B_1) \subseteq \mathrm{UR}(B_2)$
- Initial state uncertainty before any sensor event:

$$x_{0,obs} = \mathrm{UR}(\{x_0\})$$

Silent Cycle Hazard: Σ_{uo}-Loops

- If the subgraph restricted to Σ_{uo} contains directed cycles, Σ_{uo}^* generates infinite silent sequences.
- However, because $|X| < \infty$, the set $\mathrm{UR}(B)$ remains strictly finite.
- **Computational Rule:** $\mathrm{UR}(B)$ reaches its fixed point in at most $|X| - 1$ search steps via Breadth-First Search (BFS).

The Observer Automaton: Formal Definition

The Observer 6-Tuple (G_{obs})

Given plant $G = (X, \Sigma, f, \Gamma, x_0, X_m)$ with partition $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$:

$$G_{obs} := (X_{obs}, \Sigma_o, f_{obs}, \Gamma_{obs}, x_{0,obs}, X_{m,obs})$$

- **States (X_{obs})**: Power set elements $q \in 2^X \setminus \{\emptyset\}$.
- **Alphabet**: Strictly observable events Σ_o.
- **Initial State**: $x_{0,obs} = \mathrm{UR}(\{x_0\})$.
- **Marked States ($X_{m,obs}$)**: Subsets containing at least one plant marked state:

$$X_{m,obs} = \{q \in X_{obs} \mid q \cap X_m \neq \emptyset\}$$

Transition Function (f_{obs})

For an observer state $q \subseteq X$ and event $\sigma \in \Sigma_o$:

$$f_{obs}(q, \sigma) := \mathrm{UR}\big(\{x' \in X \mid \exists x \in q : f(x, \sigma) = x'\}\big)$$

- **Active Events (Γ_{obs})**: Event $\sigma \in \Sigma_o$ is active at q if at least one state in q enables σ in G:

$$\Gamma_{obs}(q) = \{\sigma \in \Sigma_o \mid \exists x \in q : \sigma \in \Gamma(x)\}$$

- G_{obs} is deterministic by construction over alphabet Σ_o.

Observer Synthesis: Subset Construction Algorithm

Step-by-Step Iterative Algorithm

1. **Initialization:** Compute root state:

$$q_0 = \mathrm{UR}(\{x_0\})$$

Set $X_{obs} \leftarrow \{q_0\}$, queue $Q \leftarrow \{q_0\}$.

2. **Exploration Loop:** While $Q \neq \emptyset$, pop state q:
   - For each $\sigma \in \Sigma_o$ such that $\exists x \in q$ with $f(x, \sigma)!$:
   - Compute successor state:

$$q' = \mathrm{UR}\big(\{x' \in X \mid \exists x \in q : f(x, \sigma) = x'\}\big)$$

   - Add edge $f_{obs}(q, \sigma) = q'$.
   - If $q' \notin X_{obs}$, insert into X_{obs} and push to Q.

Algorithmic Properties & Complexity

- **Worst-Case Complexity:** Nominal upper bound $|X_{obs}| \leq 2^{|X|} - 1$.
- **Practical Reachability:** In realistic engineering systems, physical coupling drastically restricts the accessible subset space $|X_{obs}| \ll 2^{|X|}$.
- **Language Equivalence Theorem:**

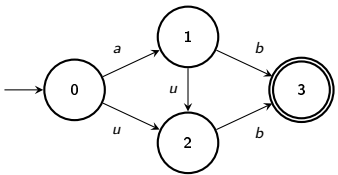
$$L(G_{obs}) = P(L(G))$$

$$L_m(G_{obs}) = P(L_m(G))$$

The observer accepts exactly the projected plant language.

State Estimation Benchmark: Plant Setup

Plant Automaton G



System Specifications

- State space: $X = \{0, 1, 2, 3\}$, $x_0 = 0$
- Marked states: $X_m = \{3\}$
- Alphabet: $\Sigma = \{a, b, u\}$

Observation Partitioning

- **Observable ($\Sigma_o = \{a, b\}$):** External triggers detected by field sensors.
- **Unobservable ($\Sigma_{uo} = \{u\}$):** Silent transition internal to the system.

The Diagnostic Dilemma

- Notice that u can occur silently from state 0 into 2, as well as from state 1 into 2.
- If sensory event a fires, did the system stay in state 1, or did it silently slip into state 2?

Example: Observer Construction Task

Task for Students

Construct the deterministic observer automaton G_{obs} for plant G over observable alphabet $\Sigma_o = \{a, b\}$:

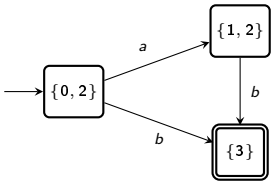
$$G_{obs} = (X_{obs}, \{a, b\}, f_{obs}, \Gamma_{obs}, x_{0,obs}, X_{m,obs})$$

Analytical Questions

1. **Initial Uncertainty:**
   - Evaluate UR($\{0\}$). Does the plant start in a singleton state or an uncertainty set?
2. **Transitions from Root State:**
   - At state $x_{0,obs}$, which observable events in Σ_o are active?
   - When $b \in \Sigma_o$ occurs from the root, which physical state is entered?
   - When $a \in \Sigma_o$ occurs from the root, compute the direct destination and apply UR.
3. **Subsequent Branching & Termination:**
   - From the state reached via a, can event b fire? Where does it transition?
   - Determine all elements of X_{obs} and identify which observer states are marked ($X_{m,obs}$).

Solution: Observer Graph Construction

Observer Automaton G_{obs}



Observer Components

- $X_{obs} = \{\{0,2\},\{1,2\},\{3\}\}$
- Observable alphabet: $\Sigma_o = \{a, b\}$
- Initial state: $x_{0,obs} = \{0,2\}$
- Marked states: $X_{m,obs} = \{\{3\}\}$

Step-by-Step Transition Analysis

- **Root Initialization:**

$$x_{0,obs} = \mathrm{UR}(\{0\}) = \{0,2\} \quad (\text{via } \epsilon, u)$$

- **Branching from** $\{0,2\}$:
  - On a: $f(0,a) = 1$, $f(2,a)!$ undef. $\implies \mathrm{UR}(\{1\}) = \{1,2\}$
  - On b: $f(0,b)!$ undef, $f(2,b) = 3$ $\implies \mathrm{UR}(\{3\}) = \{3\}$
- **Branching from** $\{1,2\}$:
  - On b: $f(1,b) = 3$ and $f(2,b) = 3$ $\implies \mathrm{UR}(\{3\}) = \{3\}$

Solution: Uncertainty Dissipation Analysis

Trace-to-State Mapping

Evaluating current uncertainty along observed traces:

- **Empty record ($t = \epsilon$):** $\mathcal{C}(\epsilon) = \{0, 2\}$
  The observer cannot know if silent event u occurred.
- **Single event ($t = a$):** $\mathcal{C}(a) = \{1, 2\}$
  The observer is still ambiguous between states 1 and 2.
- **Completion records ($t = b$ or $t = ab$):**
  $\mathcal{C}(b) = \mathcal{C}(ab) = \{3\}$
  Uncertainty completely collapses!

Key Structural Takeaways

- **Determinism:** Even though plant G contains nondeterministic silent branching, G_{obs} is strictly deterministic over Σ_o.
- **Information Convergence:** Observing event b definitively resolves all earlier ambiguity, uniquely placing the plant in marked state 3.
- **Marked Language Preserved:**

$$L_m(G_{obs}) = \{b, ab\} = P(L_m(G))$$

Properties of State Estimation: Observability

State Observability Condition

A system G is said to be **state-observable** with respect to projection P if every reachable state can be uniquely resolved:

$$\forall s \in L(G) : |\mathcal{C}(s)| = 1$$

- **Singleton Invariance:** Every accessible state in the observer automaton G_{obs} is a singleton set.
- The observer maintains zero ambiguity across all possible execution paths.

Language vs. State Observability

- **Language Observability:** Concerned strictly with control decisions: can the supervisor disambiguate when to enable or disable $\sigma \in \Sigma_c$?
- **State Observability:** A strictly stronger physical condition demanding complete reconstruction of the internal state coordinate.
- In practice, full state observability rarely holds; systems often operate under transient uncertainty.

Asymptotic Estimation: Detectability

Formal Notion of Detectability

When full state observability fails, we evaluate whether uncertainty dissipates over time:

- **Detectability:** The observer is guaranteed to reconstruct the exact current state after observing a finite number of sensor events.
- Once resolved, state uncertainty remains a singleton for all subsequent extensions.

Observer Graph Test

Inspect the directed cycles of G_{obs}:

- Every cycle must consist entirely of singleton states ($|q| = 1$).

The Unobservable Cycle Trap

Detectability is violated if G_{obs} contains:

- **Ambiguity Cycles:** A directed cycle among non-singleton states ($|q| > 1$).
- In this scenario, the plant can generate an infinite observed sequence while the supervisor remains perpetually uncertain about the true physical state.
- **Design Remedy:** Add physical sensors to break unobservable or ambiguous loops.

Section 4.1 Summary & The Diagnostic Dilemma

Summary: State Observers (G_{obs})

- **Role:** Reconstructs current physical state uncertainty $\mathcal{C}(s) \subseteq X$ using observed telemetry records $t = P(s) \in \Sigma_o^*$.
- **Structure:** Deterministic automaton over alphabet Σ_o, built via unobservable reach (UR) and subset construction.
- **Behavior:** Tracks *where the system is right now*.

The Limitation of Standard Observers

- An observer tracks the current state, but it **forgets past silent occurrences**.
- If a critical failure event $f \in \Sigma_{uo}$ occurred in the past, but the plant subsequently returned to a nominal state x, the observer output is simply $\{x\}$.
- **The Missing Question:** *"Did an unobservable fault event occur along the history of this execution?"*
- This necessitates the **Diagnoser** framework (Section 4.2).

Section 4.2
Fault Diagnosis and the Sampath Diagnoser

Core Focus:
Inferring the occurrence of silent, unobservable failures with finite detection delay

Fault Modeling: Unobservable Failure Events

The Nature of Physical Faults

In complex industrial systems, critical component failures rarely announce themselves with direct sensor flags:

- Actuator jams, valve leaks, memory corruption, and sensor bias occur silently.
- **Mathematical Representation:** Faults are modeled as events belonging to a dedicated failure set:

$$\Sigma_f \subseteq \Sigma_{uo}$$

The Diagnostic Objective

- Because $f \in \Sigma_f$ is unobservable, its occurrence cannot be detected at the instant it fires.
- **Detection Mechanism:** The health monitor must deduce that f occurred by observing the **subsequent sequence** of observable events.
- **Core Goal:** Provide definitive fault alarm within a bounded number of observed steps.

Fault Partitioning & Diagnostic Labels

Failure Event Partitioning

Failures are classified into m distinct physical fault types:

$$\Sigma_f = \Sigma_{f,1} \dot{\cup} \Sigma_{f,2} \dot{\cup} \ldots \dot{\cup} \Sigma_{f,m}$$

- $\Sigma_{f,i}$ represents a specific failure mode (e.g., motor overheat, valve stuck closed).
- Diagnosis may seek to isolate the exact fault type rather than just detecting anomaly.

Diagnostic Status Labels

To track past system health, we assign symbolic labels:

- **Normal (N):** No failure event has occurred along the execution history.
- **Faulty (F_i):** At least one failure of type $\Sigma_{f,i}$ has occurred.
- Status alphabet:

$$\Delta = \{N\} \cup \{F_1, F_2, \ldots, F_m\}$$

Diagnoser State Space: Extended State-Label Pairs

State-Label Pair Representation

To remember past unobservable events, each physical state $x \in X$ is paired with a diagnostic label $\ell \in \Delta$:

$$(x, \ell) \in X \times \Delta$$

- $x \in X$ indicates the current physical location.
- $\ell = N$ certifies no fault occurred on the trajectory leading to x.
- $\ell = F_i$ records that a failure of type i occurred in the past.

Diagnoser Macro-States ($Q \subseteq X \times \Delta$)

A state Q in the diagnoser is a non-empty set of pairs:

$$Q = \{(x_1, \ell_1), (x_2, \ell_2), \ldots, (x_k, \ell_k)\}$$

- The diagnoser simultaneously tracks:
  - Physical state uncertainty (which x_i is active),
  - Diagnostic health uncertainty (which ℓ_i holds).
- If all pairs share label F_i, the fault is diagnosed with certainty!

Label Propagation Dynamics

Propagation Rules Along Transitions

When the plant moves from (x, ℓ) to $x' = f(x, \sigma)$ via event $\sigma \in \Sigma$, the new label ℓ' is updated as:

$$\ell' = \begin{cases} F_i & \text{if } \ell = F_i \text{ (failure is permanent)} \\ F_i & \text{if } \ell = N \text{ and } \sigma \in \Sigma_{f,i} \\ N & \text{if } \ell = N \text{ and } \sigma \notin \Sigma_f \end{cases}$$

- **Irreversibility of Faults:** Once labeled F_i, the history remains faulty indefinitely.

Uncertainty Classification of Macro-States

For any diagnoser state $Q \subseteq X \times \Delta$:

- **F_i-certain (Q is safe-free):**

$$\forall (x, \ell) \in Q : \ell = F_i$$

  Fault F_i has definitively occurred.
- **N-certain (Q is fault-free):**

$$\forall (x, \ell) \in Q : \ell = N$$

  The plant is guaranteed completely healthy.
- **F_i-uncertain (Ambiguous):**

$$\exists (x, N) \in Q \quad \text{and} \quad \exists (x', F_i) \in Q$$

The Diagnoser Automaton: Formal Definition

The Diagnoser 6-Tuple (G_d)

The diagnoser is a deterministic automaton over Σ_o:

$$G_d := (Q_d, \Sigma_o, f_d, \Gamma_d, Q_0, Q_{m,d})$$

- **States (Q_d):** Subsets of $X \times \Delta$.
- **Alphabet:** Strictly observable events Σ_o.
- **Initial State:**

$$Q_0 = \mathrm{UR}_d(\{(x_0, N)\})$$

- **Transition Function (f_d):** Combines direct observation with unobservable reach and label update.

Algorithmic Step: Transition Computation

For diagnoser state Q and event $\sigma \in \Sigma_o$:

1. **Observation Step:** Apply observable transition σ to all compatible pairs:

$$Q_\sigma = \{(f(x,\sigma), \ell') \mid (x, \ell) \in Q \wedge f(x,\sigma)!\}$$

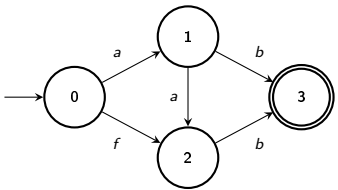
2. **Unobservable Reach Closure:**

$$f_d(Q, \sigma) = \mathrm{UR}_d(Q_\sigma)$$

Where $\mathrm{UR}_d(\cdot)$ follows all silent paths Σ_{uo}^*, continuously updating labels.

Diagnostic Benchmark: Plant with Unobservable Fault

Plant Automaton G



System Specifications

- State space: $X = \{0, 1, 2, 3\}$, $x_0 = 0$
- Marked states: $X_m = \{3\}$
- Alphabet: $\Sigma = \{a, b, f\}$

Sensory Partitioning

- **Observable Events ($\Sigma_o = \{a, b\}$):** External triggers detected by field sensors.
- **Unobservable Events ($\Sigma_{uo} = \{f\}$):** Silent transition representing an internal component fault.
- **Fault Type Set:** $\Sigma_f = \{f\}$ with single label F.

The Fault Masking Dilemma

- Event f can occur silently from state 0 directly into 2.
- If sensor a fires, did the plant move $0 \xrightarrow{a} 1$, or did it silently slip into 2 earlier, or execute $1 \xrightarrow{a} 2$?

Example: Diagnoser Construction Task

Task for Students

Construct the deterministic Sampath diagnoser G_d for plant G over observable alphabet $\Sigma_o = \{a, b\}$ and fault set $\Sigma_f = \{f\}$:

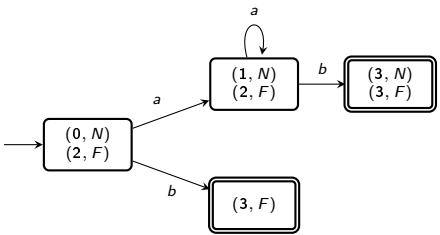
$$G_d = (Q_d, \{a, b\}, f_d, \Gamma_d, Q_0, Q_{m,d})$$

Step-by-Step Instructions

1. **Initial Root State (Q_0):**
   - Pair the initial state with label N: $(0, N)$.
   - Apply UR_d: since $0 \xrightarrow{f} 2$ is silent and $f \in \Sigma_f$, what pair does it add?
2. **Transitions from Q_0:**
   - On event b: which pair in Q_0 enables b? What is the destination and label?
   - On event a: evaluate transitions from $(0, N)$ and $(2, F)$. Is event a enabled from 2?
3. **Classify Diagnoser States:**
   - Identify which states are N-certain, F-certain, or F-uncertain.

Solution: Diagnoser Graph Construction

Diagnoser Automaton G_d



State Uncertainty Categories

- F-**certain**: $\{(3, F)\}$ (Definitive alarm)
- F-**uncertain**: Q_0, Q_1, Q_3 (Ambiguous)

Transition Derivations

- **Root** $Q_0 = \{(0, N), (2, F)\}$:
  - $0 \xrightarrow{f} 2$ yields pair $(2, F)$.
  - Q_0 is already F-uncertain at start!
- **From** Q_0 **on** b:
  - Only $(2, F)$ enables $b \implies (3, F)$.
  - State is F-certain: fault confirmed!
- **From** Q_0 **on** a:
  - $(0, N) \xrightarrow{a} (1, N)$.
  - Self-loop/reach: $(1, N) \xrightarrow{a} (2, F)$ keeps $Q_1 = \{(1, N), (2, F)\}$ ambiguous.

Formal Definition of Language Diagnosability

Diagnosability Concept (Sampath Framework)

A prefix-closed language $L \subseteq \Sigma^*$ is **diagnosable** with respect to projection P and fault set Σ_f if:

- Any unobservable fault $f \in \Sigma_f$ can be detected with certainty after a **bounded number of observable steps**.
- Once the fault occurs, the system cannot generate an arbitrarily long execution trace without the observer declaring the alarm.

Mathematical Formulation

Language L is diagnosable iff:

$$(\exists n \in \mathbb{N})\,(\forall s \in \Psi(\Sigma_f))\,(\forall st \in L,\ \|t\| \geq n \implies \forall w \in L,\ P(w) = P(st) \implies \Sigma_f \in w)$$

- $\Psi(\Sigma_f)$: traces ending with a failure event.
- $\|t\|$: number of observable events in continuation t.
- Bound n: maximum detection delay (number of sensor events required to raise the alarm).

Diagnosability Test: Indeterminate Cycles

Indeterminate Cycles: Definition

A directed cycle in the diagnoser G_d is called an **indeterminate cycle** if:

- All states in the cycle are F-uncertain (contain both normal and faulty pairs).
- There exist two traces of arbitrary length mapped to the same cycle:
  - Trace s_1 containing failure f,
  - Trace s_2 completely free of failure events.
- The observer can remain trapped indefinitely in ambiguity!

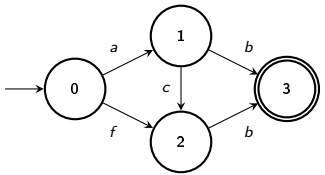
Structural Diagnosability Theorem

Assume the plant has no unobservable cycles of infinite length:

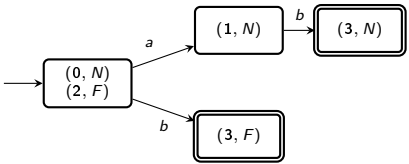
- **Theorem:** A system G is diagnosable if and only if its diagnoser G_d contains **no indeterminate cycles**.
- **Decidability:** The property can be verified in polynomial time with respect to the diagnoser graph size.
- If an indeterminate cycle exists, the detection delay cannot be bounded ($n \to \infty$).

Diagnosability Verified: Sensor Disambiguation

Modified Plant G' (Event $c \in \Sigma_o$)



Resulting Diagnoser G'_d



System Refinement

- $\Sigma_o = \{a, b, c\}, \quad \Sigma_{uo} = \Sigma_f = \{f\}$.
- The ambiguous transition $1 \xrightarrow{a} 2$ is replaced by an observable transition $1 \xrightarrow{c} 2$.

Uncertainty Collapse

- On event a: only $(0, N)$ can execute $\implies$ $Q_1 = \{(1, N)\}$ is **N-certain**!
- On event b: only $(2, F)$ can execute $\implies$ $Q_2 = \{(3, F)\}$ is **F-certain**!

Cycle Inspection & Bounded Delay Verification

Structural Cycle Analysis on G'_d

Applying the cycle test to G'_d:

- **Acyclic Topology:** The diagnoser contains **no directed cycles** among uncertain states.
- **Zero Indeterminate Cycles:** Since no uncertain loop exists, the necessary and sufficient condition is strictly satisfied.
- **Conclusion:** Plant G' is **diagnosable** with respect to failure event f.

Finite Detection Delay (K)

Evaluating the bound on sensor events after fault f:

- If f occurs from state 0, the plant enters state 2 silently.
- The only enabled continuation in G' from state 2 is event b.
- As soon as b is observed, G'_d moves to state $\{(3, F)\}$:

  Detection Delay $K = 1$ observable event

- The fault is unambiguously confirmed at the very first sensor record following f.

Section 4.2 Summary & The Dual Problem: Opacity

Summary: Fault Diagnosis (G_d)

- **State Pairs** (x, ℓ): Augments physical state x with health status $\ell \in \{N, F\}$.
- **Constructive Tool:** Sampath diagnoser built via subset construction and label propagation.
- **Diagnosability:** Verified by inspecting cycles for persistent F-uncertainty.

Paradigm Shift to Section 4.3: Opacity

- **In Diagnosis:** The supervisor **wants** to discover the truth (detect faults with zero ambiguity).
- **In Security & Opacity:** An external **adversary** eavesdrops on telemetry.
- **The Core Goal Flips:** We want the adversary to **remain uncertain** about critical secrets ($S \subset X$ or secret language L_S).
- Opacity is the formal dual of observability and diagnosability!

Section 4.3
System Opacity and Privacy Guarantees

Core Focus:
Formalizing security and information-flow privacy against passive adversarial observers

Information Privacy: The Adversarial Threat Model

The Passive Intruder Model

Consider a cyber-physical system interacting with an external adversary:

- **Full Model Knowledge:** The intruder has complete access to plant structure G and initial state x_0.
- **Passive Interception:** The intruder eavesdrops on field communication, observing filtered traces $t = P(s) \in \Sigma_o^*$.
- The intruder cannot tamper with control commands (pure observer).

The Security Objective: Plausible Deniability

- The system possesses confidential behaviors, defined as a **secret** ($S \subset X$ or $L_S \subset L(G)$).
- **Core Security Goal:** Prevent the intruder from ever determining with certainty that the secret has been accessed.
- There must always exist a non-secret explanation consistent with the observation record.

Current-State Opacity: Formal Definition

Secret States ($S \subset X$)

Let $S \subset X$ denote the designated set of confidential physical states:

- Cryptographic key generation modes,
- Restricted operational modes,
- Vulnerable system configurations.

The intruder attempts to deduce whether the current physical state $x \in S$.

Current-State Opacity Condition

System G is **current-state opaque** with respect to S and projection P iff:

$$\forall s \in L(G) \text{ with } f(x_0, s) \in S, \quad \exists s' \in L(G) :$$

$$P(s') = P(s) \quad \wedge \quad f(x_0, s') \notin S$$

- **Observer Equivalent**: For every reachable state $q \in X_{obs}$, the uncertainty set is never completely contained in S:

$$\forall q \in X_{obs} : q \not\subseteq S$$

Verification of Current-State Opacity via G_{obs}

Algorithmic Verification Procedure

Verification reduces directly to inspecting the standard observer automaton G_{obs}:

1. Construct $G_{obs} = (X_{obs}, \Sigma_o, f_{obs}, \Gamma_{obs}, x_{0,obs}, X_{m,obs})$ via subset construction.
2. Identify the secret state set $S \subset X$.
3. **Safety Inspection:** Check every accessible macro-state $q \in X_{obs}$:

Test: Is $q \subseteq S$?

Secret Leak Condition

- **Opacity Violation:** If there exists an accessible macro-state $q \in X_{obs}$ such that:

$$q \subseteq S$$

then the adversary deduces with certainty that the current physical state is secret!

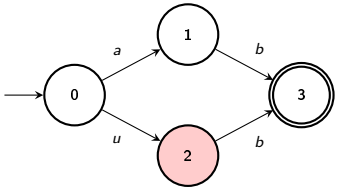
- **Plausible Deniability Condition:**

$$\forall q \in X_{obs} : q \setminus S \neq \emptyset$$

Every reachable estimate must retain at least one non-secret physical state.

Current-State Opacity Benchmark: Plant Setup

Plant Automaton G (Secret State $S = \{2\}$)



System Specifications

- State space: $X = \{0, 1, 2, 3\}, \quad x_0 = 0$
- Secret state subset: $S = \{2\}$ (shaded red)
- Alphabet: $\Sigma_o = \{a, b\}, \quad \Sigma_{uo} = \{u\}$

Intruder Observation Channel

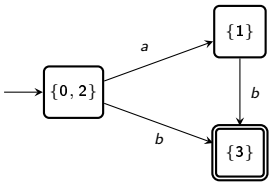
- The intruder observes traces through natural projection $P : \Sigma^* \to \{a, b\}^*$.
- Transition $0 \xrightarrow{u} 2$ occurs silently into the secret state.
- If the plant moves silently to state 2, can the intruder infer it from future observation?

Security Verification Task

Evaluate the observer macro-states for G to determine whether secret $S = \{2\}$ remains opaque.

Solution: Observer Evaluation for Opacity

Intruder Observer G_{obs}



Macro-State Inspection

- Root state: $x_{0,obs} = \mathrm{UR}(\{0\}) = \{0, 2\}$
- On event a: successor is $\{1\}$
- On event b: successor is $\{3\}$

Opacity Verdict: Satisfied!

Testing condition $q \subseteq S$ where $S = \{2\}$:

- $q_0 = \{0, 2\} \not\subseteq \{2\}$ (since $0 \notin S$)
- $q_1 = \{1\} \not\subseteq \{2\}$
- $q_2 = \{3\} \not\subseteq \{2\}$

Conclusion:

- Plant G is **current-state opaque** with respect to $S = \{2\}$.
- Whenever the plant is in secret state 2, the intruder's estimate is $\{0, 2\}$—maintaining plausible deniability with state 0.

Extended Opacity Formulations in DES

Initial-State Opacity (ISO)

- **Setting:** The plant starts from an unknown initial state $x_0 \in X_0$, some of which are confidential ($S_0 \subset X_0$).
- **Condition:** The intruder must never be able to deduce that the execution originated from S_0:

$$\forall s \in L(G, S_0), \exists s' \in L(G, X_0 \backslash S_0) : P(s') = P(s)$$

- Requires backward state estimation or reverse observer construction.

K-Step and Language Opacity

- **K-Step Opacity:** A secret state visit must remain concealed for at least K subsequent observable steps before exposure becomes harmless.
- **Language Opacity:** A confidential execution sequence $L_S \subset L(G)$ cannot be distinguished from non-secret traces:

$$P(L_S) \subseteq P(L(G) \setminus L_S)$$

- Provides language-level privacy guarantees against passive observers.

Language-Based Opacity: Formal Definition

Language-Based Secrecy ($L_S \subset L$)

In many security protocols, secrecy resides in an **execution sequence** rather than a physical state coordinate:

- Accessing a database followed by unlogged extraction: $s \in L_S$.
- The confidential behavior is defined as a regular sublanguage:

$$L_S \subseteq L(G), \quad L_S \neq \emptyset$$

- Non-secret behavior: $L_{NS} := L(G) \setminus L_S$.

Language-Based Opacity Condition

Language L_S is **opaque** with respect to $L(G)$ and projection P iff:

$$P(L_S) \subseteq P(L(G) \setminus L_S)$$

- **Plausible Deniability at Trace Level:** For every secret trace $s \in L_S$, there exists at least one non-secret trace $s' \in L(G) \setminus L_S$ such that:

$$P(s) = P(s')$$

- The intruder observes record $t = P(s)$ and can never be sure whether $s \in L_S$ or $s' \notin L_S$ generated it.

Verification of Language-Based Opacity

Automata-Theoretic Reduction

Let G_S mark the secret language ($L_m(G_S) = L_S$) and G generate the plant behavior:

1. **Secret Projection:** Construct the projected observer automaton for the secret:

$$\text{Obs}(G_S) \implies L_m(\text{Obs}(G_S)) = P(L_S)$$

2. **Non-Secret Projection:** Construct the observer for non-secret traces:

$$\text{Obs}(G_{NS}) \implies L_m(\text{Obs}(G_{NS})) = P(L(G) \setminus L_S)$$

3. **Language Inclusion Check:** Verify if:

$$P(L_S) \setminus P(L_{NS}) = \emptyset$$

Exposure Condition: Revealing Traces

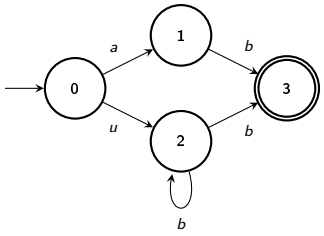
- An observed record $t \in \Sigma_o^*$ is a **revealing trace** if:

$$t \in P(L_S) \quad \text{and} \quad t \notin P(L(G) \setminus L_S)$$

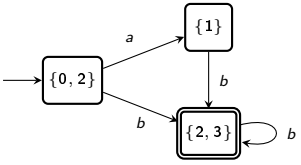
- When the intruder detects t, all possible explanations $s \in P^{-1}(t) \cap L(G)$ lie strictly inside the secret language L_S.
- Secrecy is completely broken at that instant.
- **Decidability:** Solvable via standard DFA intersection and emptiness checks in polynomial time.

Language Opacity Benchmark: Setup & Violation

Plant Automaton G



Intruder Observer G_{obs}



Secret Language Specification

- $\Sigma_o = \{a, b\}, \quad \Sigma_{uo} = \{u\}$.
- Secret string: $L_S = \{ub\} \subset L(G)$.
- Non-secret traces: $L_{NS} = L(G) \setminus \{ub\}$.

Opacity Analysis of Trace $t = b$

- $P(L_S) = P(\{ub\}) = \{b\}$.
- Can non-secret strings produce record $t = b$?
- In G, record b can only be produced by ub (which is in L_S).
- Thus, $b \notin P(L_{NS}) \implies P(L_S) \not\subseteq P(L_{NS})$.
- **Result:** L_S is **not language-opaque**!

Enforcement of Opacity: Supervisory Control

The Opacity Enforcement Problem

When a plant G violates opacity, a supervisor S must intervene:

- Restrict system behavior by dynamically disabling controllable events Σ_c.
- **Control Goal:** Synthesize supervisor S such that the closed-loop system S/G is current-state opaque with respect to $S \subset X$.
- **Maximally Permissive:** Restrict nominal behavior only when a secret exposure is imminent.

Enforcement Architectures

- **Supervisory Pruning:** Trim all paths leading to revealing observer states ($q \subseteq S$) using backward reachability over G_{obs}.
- **Dynamic Sensor Masking:** Turn physical sensors on/off dynamically to selectively withhold information from the intruder.
- **Insertion Functions:** Causal communication masks that insert dummy observable events into the public channel to confuse the adversary.

Section 4.4: Review & Assessment

Conclusions, Theoretical Questions & Exercises

Core Focus:

Consolidating state estimation, fault diagnosis, and opacity through analytical practice

Part 4 Synthesis: Estimation vs. Diagnosis vs. Opacity

Comparative Architectural Framework

Dimension	State Estimation (4.1)	Fault Diagnosis (4.2)	Opacity & Privacy (4.3)
Observer Role	Friendly Monitor	Internal Health Agent	Hostile External Adversary
Tracked Entity	Current state: $x \in X$	Past failure events: $f \in \Sigma_f$	Confidential states/traces: $S \subset X$
State Space	Power set: $q \in 2^X \setminus \{\emptyset\}$	Extended pairs: $Q \subseteq X \times \Delta$	Estimator subsets: $q \in X_{obs}$
Desired Outcome	Zero Uncertainty: $\lvert C(s)\rvert = 1$	Alarm Certainty: $Q \subseteq X \times \{F\}$	Persistent Ambiguity: $q \not\subseteq S$
Structural Test	Singleton cycles in G_{obs}	No indeterminate cycles in G_d	No pure secret macrostates in G_{obs}

Duality Principle

- **Diagnosis and Opacity are exact conceptual duals**: diagnosis aims to eliminate uncertainty regarding critical events, whereas opacity deliberately preserves uncertainty to protect confidential behavior.

Part 4 Summary: Key Takeaways

Core Theoretical Deliverables

- **Natural Projection (P):** Formal mathematical model of sensor limitations and silent transitions.
- **Observer Automaton (G_{obs}):** Deterministic tracking of current physical state uncertainty via UR.
- **Sampath Diagnoser (G_d):** Extended state-label tracking guaranteeing bounded-delay failure alarms.
- **Opacity Verification:** Ensuring plausible deniability against passive eavesdroppers.

The Complete DES Journey

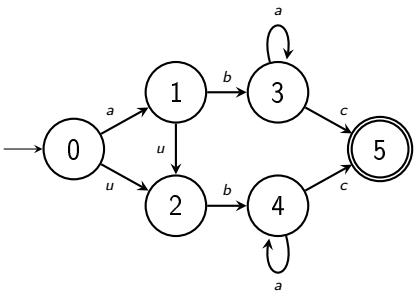
- **Part 1:** Foundation and Linguistic Models.
- **Part 2:** Automata Models and Operations.
- **Part 3:** Supervisory Control Theory.
- **Part 4:** Partial observation: Estimation, Diagnosis, Security).

Part 4: Theoretical Review Questions

1. **Unobservable Reach & Fixed Points**: Why does the computation of $\mathrm{UR}(B)$ always terminate in a finite number of steps on finite-state automata, even when the unobservable sub-alphabet contains directed cycles?
2. **Observer vs. Diagnoser States**: Why is tracking subsets $q \subseteq X$ sufficient for state estimation, but structurally inadequate for fault diagnosis? Explain what information the extended state-label pair $(x, \ell) \in X \times \Delta$ preserves that standard subset construction loses.
3. **Permanence of Diagnostic Health**: In label propagation, why is a transition from label F_i back to label N forbidden along any path in G_d? What physical assumption about faults does this reflect?
4. **Detectability vs. Diagnosability**: Contrast the asymptotic requirement of detectability in G_{obs} with bounded-delay diagnosability in G_d. Why does diagnosability allow the system to visit ambiguous states indefinitely, provided it does not stay trapped in them after a fault?
5. **Indeterminate Cycles**: Define the difference between an F-uncertain cycle in G_d and an indeterminate cycle. Why is the existence of an F-uncertain cycle a necessary, but not always sufficient, condition for non-diagnosability?
6. **Estimation-Opacity Duality**: Explain why the design objectives of fault diagnosis and current-state opacity are exact mathematical duals under partial event observation.

Practice Exercise: State Estimation

Plant Automaton G_1



$$\Sigma_o = \{a, b, c\}, \quad \Sigma_{uo} = \{u\}$$
$$X = \{0, 1, 2, 3, 4, 5\}, \quad X_m = \{5\}$$

Observer Design Task

Construct the deterministic observer automaton:

$$G_{obs} = (X_{obs}, \Sigma_o, f_{obs}, \Gamma_{obs}, x_{0,obs}, X_{m,obs})$$

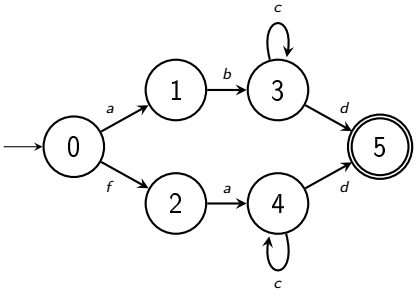
Draw all macro-states using rectangular nodes.

Analytical Questions

1. Determine the root state $x_{0,obs} = \mathrm{UR}(\{0\})$. Does it contain initial ambiguity?
2. Compute the current-state estimate $\mathcal{C}(s)$ for traces $s_1 = b$ and $s_2 = ab$.
3. Is plant G_1 **state-observable**? Justify via X_{obs}.
4. Inspect all directed cycles in G_{obs}. Does the loop driven by event a allow persistent ambiguity? Conclude on **detectability**.

Practice Exercise: Fault Diagnosis

Plant Automaton G_2



$\Sigma_o = \{a, b, c, d\}, \quad \Sigma_{uo} = \Sigma_f = \{f\}$

Fault label alphabet: $\Delta = \{N, F\}$

Diagnoser Synthesis Task

Synthesize the Sampath diagnoser:

$$G_d = (Q_d, \Sigma_o, f_d, \Gamma_d, Q_0, Q_{m,d})$$

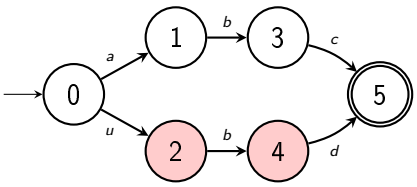
using extended state-label pairs $(x, \ell) \in X \times \{N, F\}$.

Diagnostic Evaluation

1. Compute root $Q_0 = \mathrm{UR}_d(\{(0, N)\})$. What is its diagnostic health status?
2. Partition each reachable macro-state in Q_d into N-certain, F-certain, or F-uncertain.
3. Trace the execution along the cycle driven by event c. Does it form an **indeterminate cycle**?
4. Is plant G_2 **diagnosable**? If not, what sensor modification would guarantee bounded delay?

Practice Exercise: Current-State Opacity

Plant Automaton G_3



$\Sigma_o = \{a, b, c, d\}, \quad \Sigma_{uo} = \{u\}$

Secret state set: $S = \{2, 4\}$ (shaded red)

Intruder Estimation Task

Construct the intruder observer automaton:

$$G_{obs} = (X_{obs}, \Sigma_o, f_{obs}, \Gamma_{obs}, x_{0,obs}, X_{m,obs})$$

assuming natural projection $P : \Sigma^* \to \Sigma_o^*$.

Security Analysis Questions

1. When the plant reaches secret state 2, what is the intruder's estimate? Is plausible deniability preserved?
2. Evaluate the state reached in G_{obs} after observing trace $t = b$. Does it satisfy $q \not\subseteq S$?
3. What happens to secrecy if event d is observed? Does the adversary reveal a past/current visit?
4. Formally conclude whether G_3 is **current-state opaque** with respect to secret set S.