Event-based Fault Diagnosis under Different Levels of Uncertainty

Ali Karimoddini, PhD
Director, CR^{2}C^2 Regional University Transportation Center
Director, NC-CAV Center of Excellence in Advanced Transportation Technology
Director, ACCESS Laboratory
Department of Electrical and Computer Engineering
North Carolina A&T State University
Website: http://akarimod.info

ACCESS Laboratory

NC A&T State University

http://accesslab.net http://www.ncat.edu

Motivation

ACCESS Laboratory

Fault - a malfunction in system component(s) (actuators, sensors,…etc.) that results in unacceptable system performance, and/or system instability

Motivation

Facts:

- Despite all our efforts, faults in a system cannot be avoided.
- Faults may occur every where in a system.

F-35 crashes for the first time in the jet's 17-year history

There is a need for **automatically diagnosing faults** so that if a fault occurs, the system can be recovered to accomplish the originally assigned task or at least can be brought to a safe mode!

Traditional fault diagnosis

Automatic fault diagnosis

Motivation

Failure to properly diagnose faults, leads to incorrect recovery actions

"Wow, pulled back the wrong side throttle"…… (captain of TransAsia Flight GE235)

American Airlines Flight 191 (1979)

- **Left Engine separated from wing**
- **Pilot only 15s to react**
- **Subsequent analysis shows consequence of faults avoidable**

Motivation

ACCESS Laboratory

Antares Failure during Orb-3 Launch (Oct 28, 2014)

- On October 28, 2014, 6:22 p.m. (EST), *Orbital ATK* launched its *Orb-3 cargo* from NASA's Flight Facility in Virginia.
- Just over 15 seconds into flight, an explosion in the Main Engine System (MES) occurred, causing the vehicle to lose thrust and fall back toward the ground.

Motivation

ACCESS Laboratory

Orb–3 Accident Investigation Report (October 9, 2015)

Technical Findings	
TF-1	T se o
TF-2	G te b
TF-3	T a
TF-4	A a
TF-5	A th
TF-6	B

ne

- TRC-
pump

Technical Recommendations		Finding(s) Addressed by Recommendation
TR-1	NASA should not rely on the AJ26 for further missions without undertaking a more thorough inspection, qualification and acceptance test, and certification program.	TF-1, TF-2, TF-4, and TF-6
TR-2	For the new RD-181 engine that Orbital ATK has identified as a replacement to the AJ26 engine, Orbital ATK should ensure a thorough qualification program and acceptance test program is implemented specific to planned Antares operations.	TF-2 and TF-4 (Also, PF-7, PF-8)
TR-3	For future Antares missions, additional MES sensors and sensor filtering should be provided by Orbital ATK.	TF-3
TR-4	For future engine ATPs, sensors more suitable to the test environment should be utilized, and sensors should be better placed to understand and characterize engine performance.	TF-3

➢ The IR
Recom

Question:
Are we able to place sensor for every possible fault?

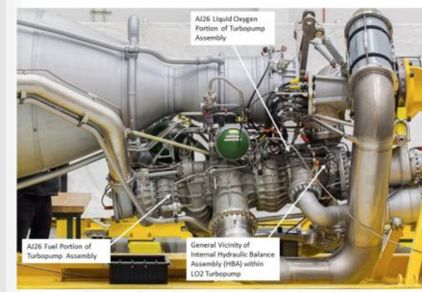


Figure 3. AJ26 Engine on Transportation and Processing Skid

Figure 4. Antares First-Stage Core with Two AJ26 Engines Installed

National Aeronautics and Space Administration (NASA). "Nasa Independent Review Team Orb – 3 Accident Investigation Report (Executive Summary)." http://www.nasa.gov/sites/default/files/atoms/files/orb3_irt_execsumm_0.pdf.

Motivation

ACCESS Laboratory

Problem: In case of occurring a fault in the system, how to automatically diagnose the occurred fault from the external observations of the systems?

Failure detection:	• Is a failure happened in the system?
Failure identification:	• What is the type of failure?
Failure location:	• Where is the place of failure in the system?

Challenges

ACCESS Laboratory

Faults may happen **any time** and **any where** in the system causing despondent situations.

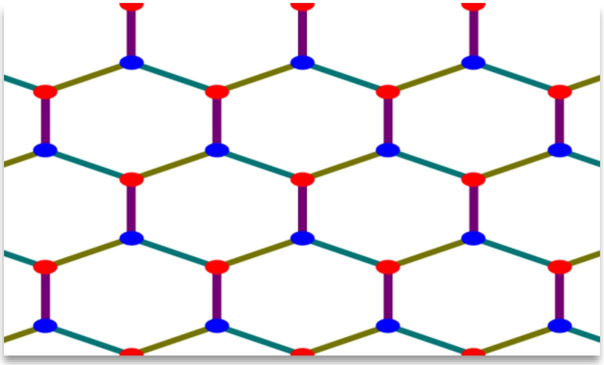
Though we may use sensors for important possible failures, but practically **we cannot have a dedicated sensor for every possible failure** as failures may happen everywhere anytime.

Usually systems are partially or completely unknown. So, we might **not always have access to the model of the system and its failures.**

Faults should **be diagnosed in the shortest possible time** to make it possible to be accommodated.

DES Framework

ACCESS Laboratory



Behavior Model

The arbitrary nature of unobservable fault occurrences leads to the non-deterministic, non-linear behavior of highly complex systems; inherently making a discrete-event modelled system representation quite suitable for diagnosing fault occurrences.

Cause & Effect

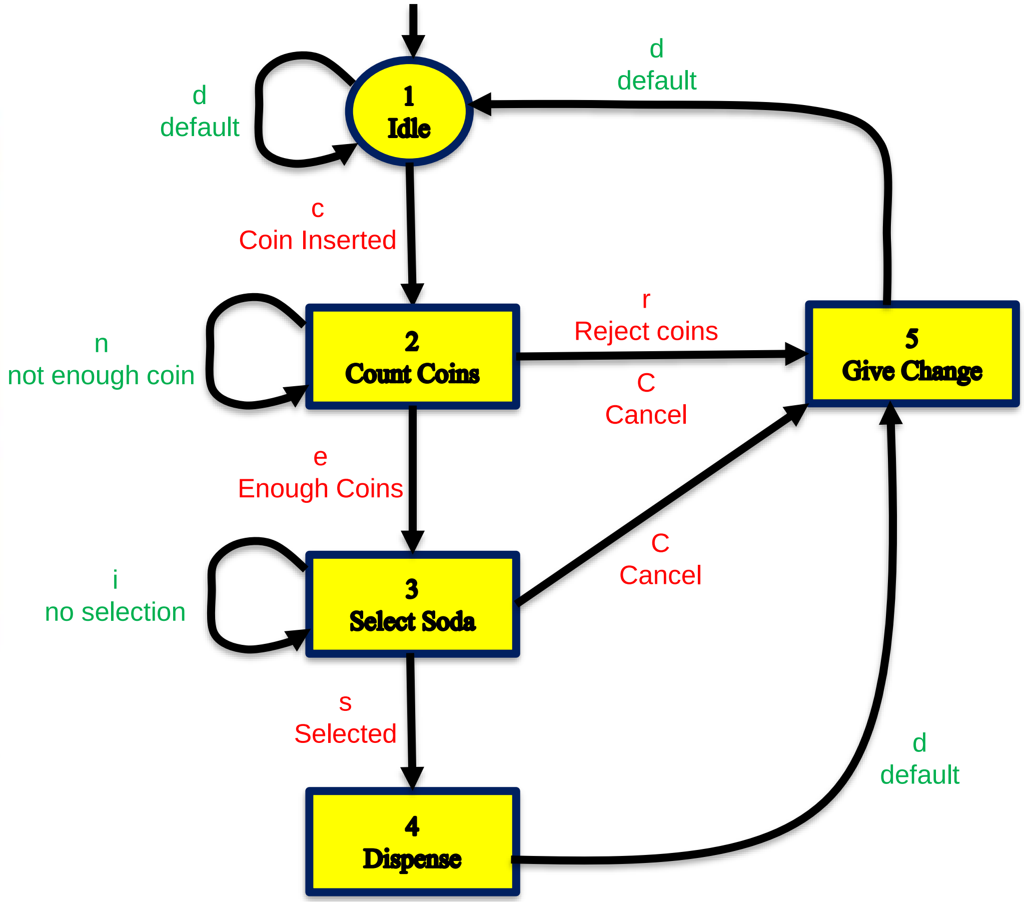
The common structure of DES consists of various sequences of events/actions leading to various system states. This structure matches a human's instinctive perception of cause and effect, thus providing for more natural intuitive system analytics.

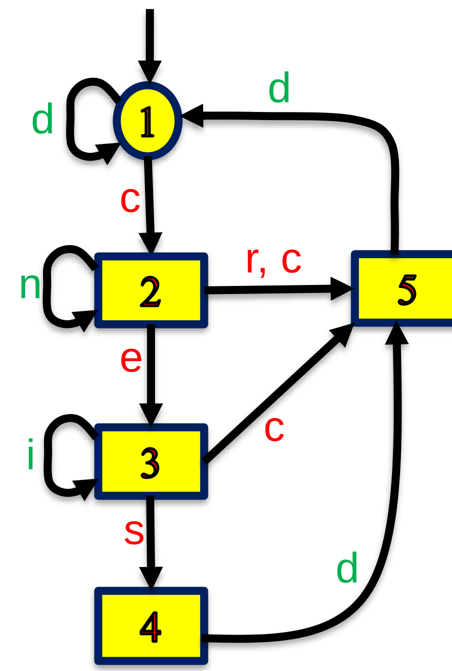
Topology

The topology of a DES, represents a system's behavior as sequences of discrete events. This allows for the capturing of disruptive changes in a system's operation; in turn highlighting faulty behaviors of the system.

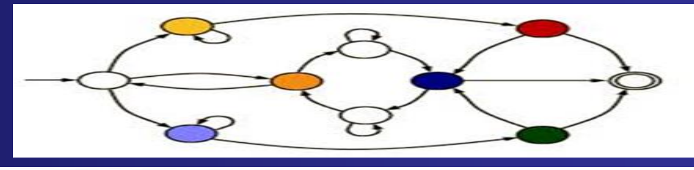
Preliminaries and Background

Automaton


1
Idle
2
Count Coins
3
Select Soda
4
Dispense
5
Give Change
d
default
c
Coin Inserted
n
not enough coin
e
Enough Coins
i
no selection
s
Selected
r
Reject coins
C
Cancel
C
Cancel
d
default
d
default


1
2
3
4
5
d
c
n
e
i
s
r, c
c
d
d

Automaton



ACCESS Laboratory

❖ Definition: a non-deterministic finite-state Discrete-Event System (DES) can be represented by a four-tuple: $\mathrm{G} = (\mathrm{X}, \Sigma, \delta, x_0)$

❖ **State space (X): a discrete set of system states** $X = \{1, 2, 3, 4, 5\}$

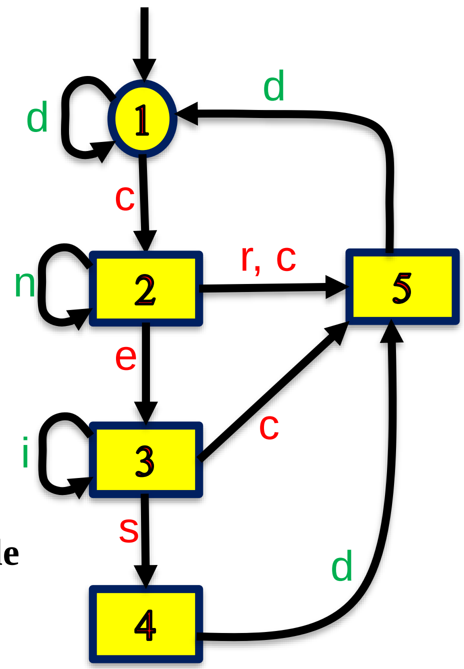
❖ **Event set ($\Sigma = \Sigma_o \cup \Sigma_u$): notable occurrences of asynchronous discrete changes in a system** $\Sigma = \{c, e, s, r, d, n, i\}$

- ❖ **Observable events (Σ_o): events observed by a sensor (e.g., flowing of water)**
  $\Sigma_o = \{c, e, s, r\}$

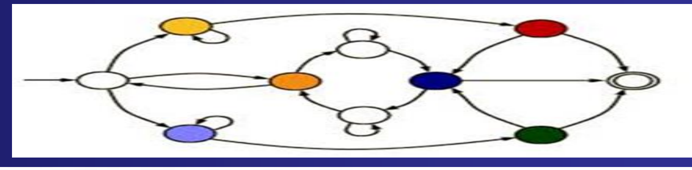
- ❖ **Unobservable events (Σ_u): events that are unable to be detected by sensors; possibly due to sensor absence/damage (e.g., failure event)**
  $\Sigma_u = \{d, n, i\}$

❖ **State-transition relation ($\delta: X \times \Sigma \rightarrow 2^X$): a partial relation that determines all feasible system state transitions caused by system events (2^X is the set of all possible combinations of states)**
$\delta(1, c) = 2,\ \delta(2, e) = 3,\ \delta(3, i) = 3$

❖ **Initial state (x_0): indicated by an input arrow connected to a single state**
$x_0 = \{1\}$

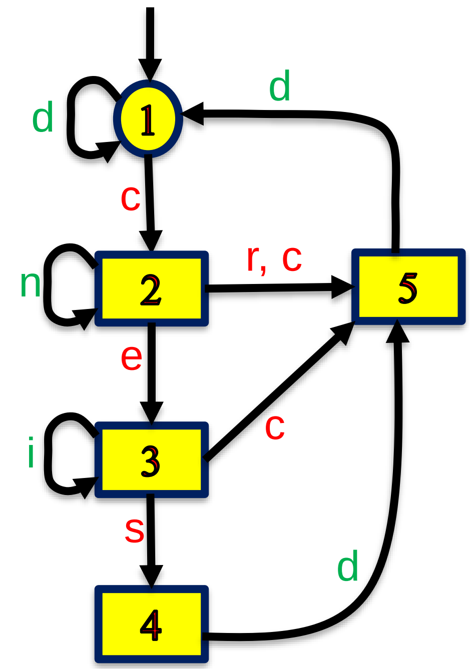


Automaton



The system language is a discrete representation of the system's behaviors (normal and faulty) in the form of sequences of events

- ❖ **Trace (string): a sequence of one or more events, allowable by the system's behavior**
  - ❖ **e.g., $s = e_1 e_2 \ldots e_n$ where $e_i \in \Sigma$**

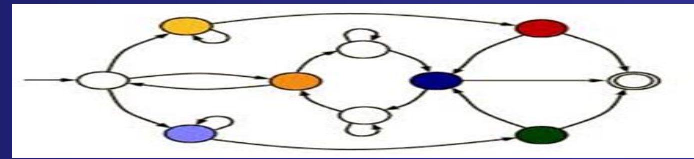


- ❖ **Language ($\mathcal{L}_G(x_0)$): the set of all system <u>traces</u> which originate at the system's initial state x_0**
  - ❖ $\mathcal{L}_G(x_0) = \{s \in \Sigma^* | \delta(x_0, s) \text{ is defined}\}$

(Σ^* is the Kleene Closure of Σ)

❖ $\mathcal{L}_G(x_0) = \{\varepsilon, d^*, ce, cesdd, cei^*s, \ldots\}$

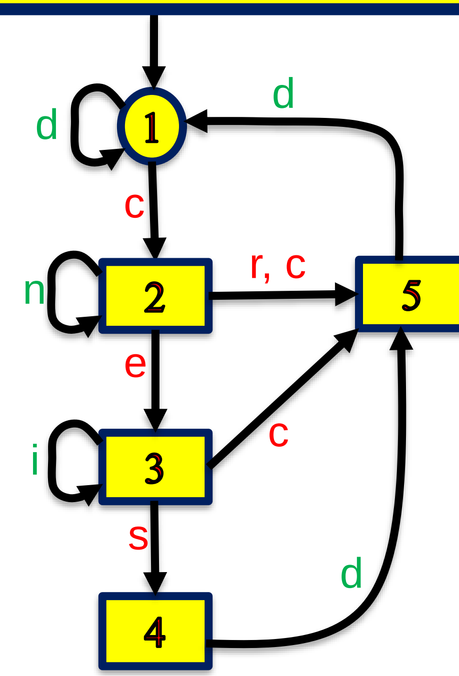
Automaton



- Our purpose is to diagnose unobservable faults from the observable behavior of the system.
- The system's observable behavior can be described by the natural projection (***P***) of the system's language to the observable language set of the system.

$$P: \Sigma^* \rightarrow \Sigma_o^*$$

$$\begin{cases} P(e) = e & \text{if } e \in \Sigma_0 \\ P(e) = \varepsilon & \text{if } e \notin \Sigma_o \\ P(se) = P(s)P(e) & \text{for } s \in \Sigma^* \text{ and } e \in \Sigma \end{cases}$$



Extension of the natural projection to the languages:

$$P(\mathcal{L}_{\mathbf{G}}(\mathbf{x_0})) = \{P(s) \mid s \in \mathcal{L}_{\mathbf{G}}(\mathbf{x_0})\}$$

Inverse of Natural Projection

$$P^{-1}_{\mathcal{L}_G(x_0)}(w) = \{s \in \mathcal{L}_{\mathbf{g}}(\mathbf{x_0}) \mid P(s) = w\}$$

- $\mathcal{L}_G(x_0) = \{\varepsilon, ce, cesdd, cei^*s, \dots\}$
- $\mathbf{P}(\mathcal{L}_G(x_0)) = \{\varepsilon, ce, ces, ces, \dots\}$
- $P^{-1}_{\mathcal{L}_G(x_0)}(c) = \{d^*cn^*\}$

Proposed Framework

Fault Diagnosis

ACCESS Laboratory

Observations

Plant
$G = (X, \Sigma, \delta, x_0)$

$\mathcal{L}_G(x_0)$ →

Natural Projection
$P: \Sigma^* \rightarrow \Sigma_o^*$

$P(\mathcal{L}_G(x_0))$ →

Diagnoser

→

Fault Detection
Fault Isolation
Fault Identification

- **Fault diagnostics is provided by the diagnoser.**
- **The diagnoser extracts information from the original system's observable behaviors, in order to estimate the original system's current state and current condition (faulty or non-faulty).**
- **The diagnoser's transitions are only defined over the original system's observable event occurrences.**
- **Upon observance of the original system's behavior, the diagnoser updates its estimation of the original system's state and condition.**

Scope of Work

ACCESS Laboratory

We consider different levels of uncertainties:

1- Diagnosis of an **Unknown System**

2- Diagnosis under **unknown initial condition** of the system and with **unknown past history**

3- Diagnosis under **partially unknown initial condition** of the system and with **partially unknown past history .**

Developed approaches:

1- **Active-learning** for knowing the system and diagnosing the occurred failures at the same time.

2- **Asynchronous diagnosis** for a system with unknown initial condition and with unknown past history

3- **Semi-Asynchronous diagnosis** for a system with partially unknown initial condition of the system and with partially unknown past history.

Assumptions

ACCESS Laboratory

Consider that in the plant G, failures $f_1, f_2, \ldots, f_n$ can happen:

1. Faults are unobservable:

We assume that these events are unobservable events in the automaton G, i.e. $\Sigma F = \{f_1, f_2, \ldots, f_n\} \subseteq \Sigma uo$, otherwise, if failures are observable events, then they can be trivially and immediately diagnosed.

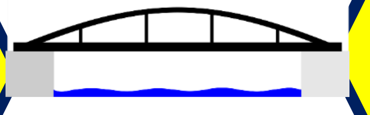
2. We consider those faults that are abrupt changes in the system, and can be modelled as "events" making a distinct change in the system:

These changes (transition $x \xrightarrow{f_i} x'$) can be captured by the transition function $\delta(x, f_i) = x'$ in automaton G

3. Failures do not bring the system to a halt mode:

Therefore, there will be enough time to diagnose failures from the observable behavior of the system $P_{\Sigma o}\left(\mathcal{L}(G)\right)$.

In many practical situations, the system under diagnosis is not completely known.



Active-learning diagnosis

In all of existing methods, the perfect and complete information about the system under diagnosis is needed.

Proposed Approach

Plant
$G = (X, \Sigma, \delta, x_0)$

$\mathcal{L}_G(x_0)$ →

Natural Projection
$P: \Sigma^* \to \Sigma_o^*$

$P(\mathcal{L}_G(x_0))$ →

Diagnoser →

Fault Detection
Fault Isolation
Fault Identification

Approach

We use the theory of Discrete Event Systems to model the failures.

We develop a “diagnoser” as a diagnosis tool.

- The diagnoser extracts information from the original system’s observable behaviors, in order to estimate the original system’s current state and current condition (faulty or non-faulty).
- The diagnoser’s transitions are only defined over the original system’s observable event occurrences.
- Upon observance of the original system’s behavior, the diagnoser updates its estimation of the original system’s state and condition.

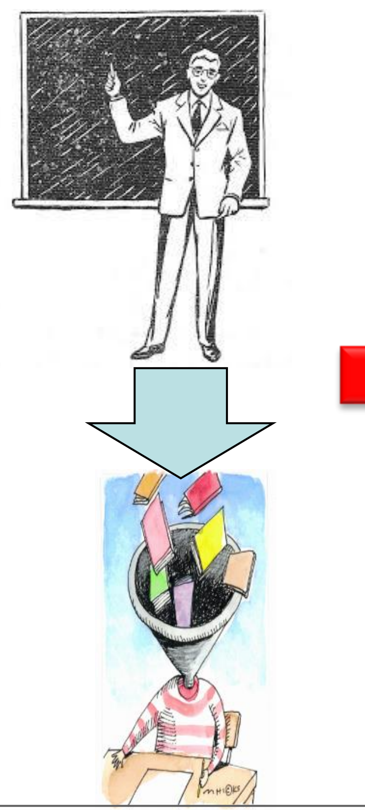
In the absence of complete information about the system, we develop an **active learning** technique to adaptively build-up a diagnosis tool for the system.

Passive Learning vs. Active Learning

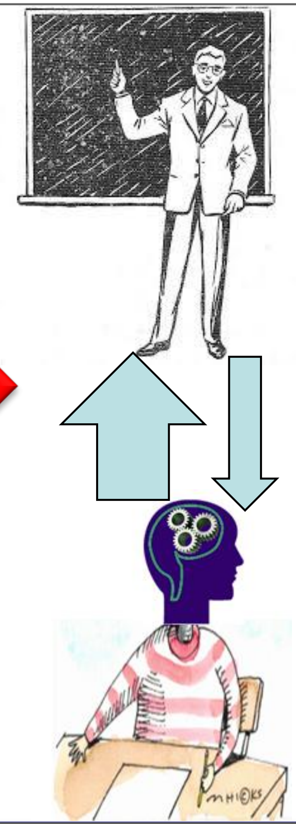
ACCESS Laboratory

Passive Learning

- Teacher provides all available information about the system.
- Learner fits a model for the provided information.
- The learner passively learns the trained information and only can work over the training range.



Active Learning



- The learner asks basic questions about missing information about the system.
- The teacher answers the questions about the system.
- The learner actively learns the enquired information and gradually builds a model for the system.

Question:
How about the case that a new situation happens and the learner is not trained for it?

The active learner can **gradually and adaptively** construct a model for a system.

Contributions

Developed a discrete event system framework for fault diagnosing for a completely unknown system.

Developed a systematic active learning strategy to construct the diagnoser to provide diagnosis for an unknown system.

Actively asking basic minimum queries from an oracle, the proposed algorithm will come up with a labeled deterministic finite state automaton as the system fault diagnoser.

An independent label propagation technique is designed to make the algorithm more efficient to construct the diagnoser.

General Structure of the Developed Diagnoser

ACCESS Laboratory

Diagnoser G_d can be described by a labeled automaton using the following tuple:

$$G_d = (Q_d, \Sigma_d, \Delta, \delta_d, h, q_0)$$

Q_d is the set of diagnoser states	δ_d is the transition rule
$\Sigma_d = \Sigma_o$ is the event set	$h : Q_d \longrightarrow \Delta$ is the output function
Δ is the output label set	q_0 is the initial state

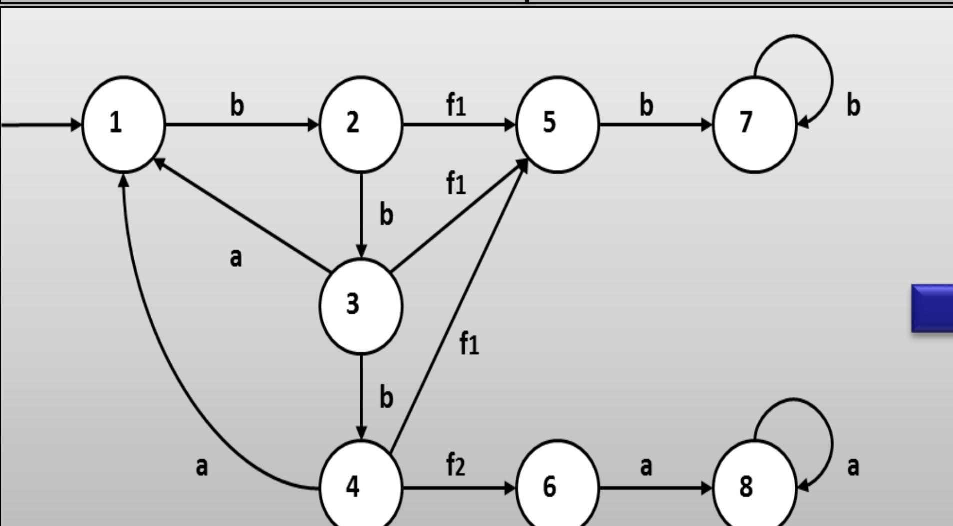
$$\Delta = \{N\} \cup \{L_1, L_2, \dots, L_m\}, L_i \in \{F_i, A_i\}$$

N: normal

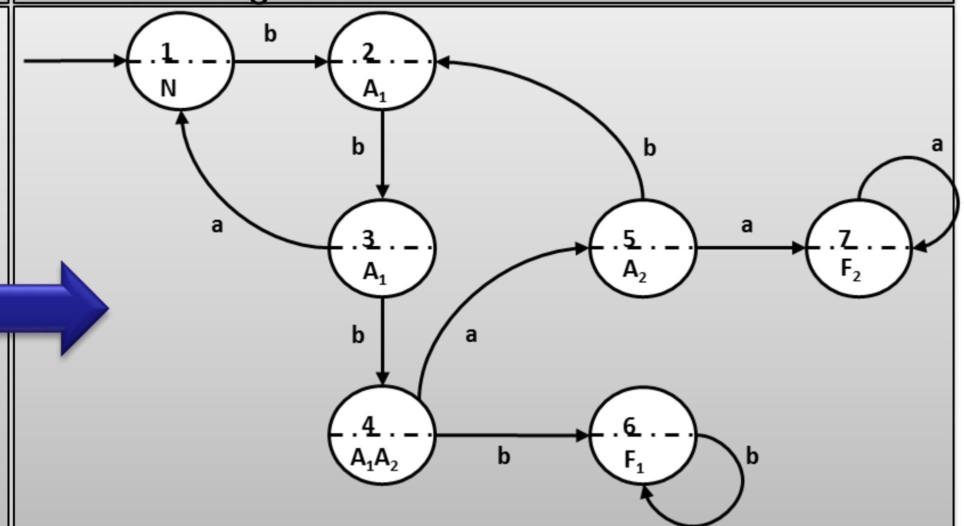
F_i: occurrence of the failure f_i

A_i: ambiguity in the occurrence of the failure f_i

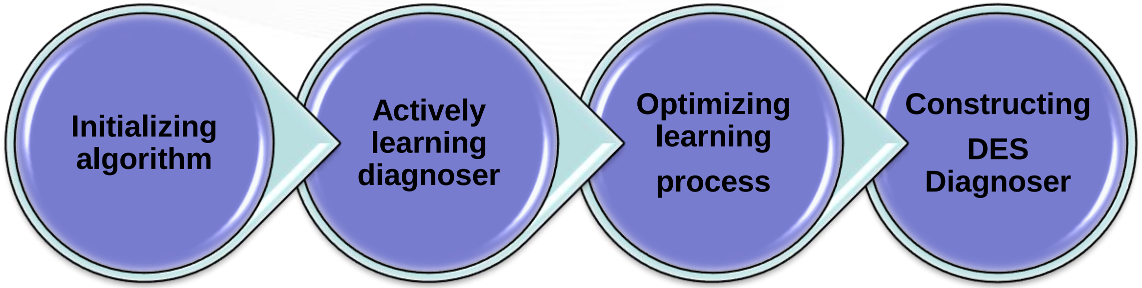
- The DES Model of the plant



- The diagnoser



General Overview of the Proposed Algorithm



1. The proposed algorithm gradually learns the diagnoser starting with an initialized diagnoser, building up a deterministic label transition system for an unknown DES plant.
2. The proposed active-learning mechanism **acquires the required information** through an oracle who answers some basic queries about the system and observable strings.
3. A **label propagation** method is introduced to make the fault diagnosing more efficient.
4. With the acquired information, the algorithm completes a series of observation tables, and eventually conjectures a correct diagnoser.

Construction of the Diagnoser: The Role of Oracle

The algorithm constructs the diagnoser G_d by asking minimum queries from an oracle who correctly answers two types of basic queries:

1. Membership queries:

Whether a newly observed strings belongs to $P_{\Sigma o}(\mathcal{L}(G))$, and if it is faulty.

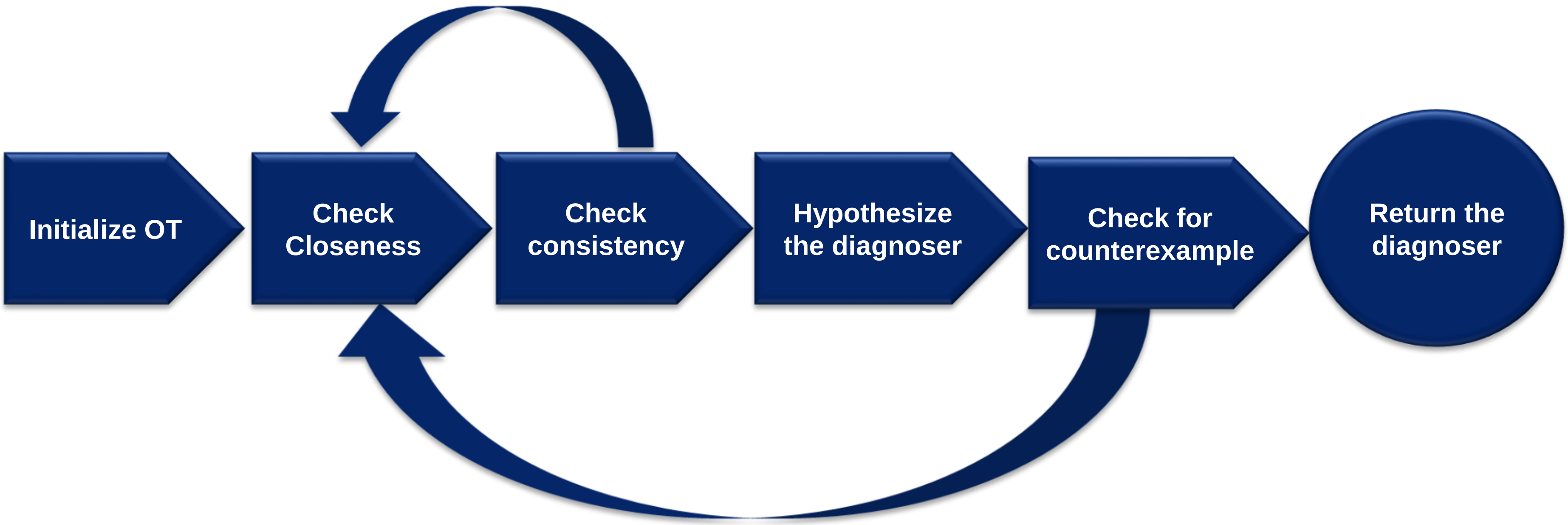
2. Equivalence queries:

- Whether $\mathcal{L}(G_d) = P_{\Sigma o}(\mathcal{L}(G))$
- If not, the oracle returns a counterexample:

$$cex \in \mathcal{L}(\mathrm{G_d}) \backslash P_{\Sigma \mathrm{o}}\left(\mathcal{L}(G)\right) \cup P_{\Sigma \mathrm{o}}\left(\mathcal{L}(G)\right) \backslash \mathcal{L}(\mathrm{G_d})$$

The Proposed Algorithm

ACCESS Laboratory



Initializing algorithm → Actively learning diagnoser → Optimizing learning process → Constructing DES Diagnoser

The acquired information will be used to create a series of observation tables (S,E,T), where

$S \subseteq \Sigma^*$ is a non-empty, prefix closed, finite set of strings

E is a non-empty, suffix closed, finite set of strings

T is the Condition Map:

$$T(s):(S \cup S.\Sigma_o).E \rightarrow \Delta \cup \{0\}$$

T_2		**E**
		ϵ
S	ϵ	N
	a	0
	b	A_1
$S\Sigma_0 - S$	aa	0
	ab	0
	ba	0
	bb	$\boldsymbol{A_1}$

OTs incrementally record and maintain the information about the observed strings.

Construction of the Diagnoser: The Condition Map

Plant
$\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{\Sigma}, \boldsymbol{\delta}, \boldsymbol{x_0})$

$\boldsymbol{\mathcal{L}(G)}$

Natural Projection
$P_{\Sigma o}: \boldsymbol{\Sigma}^* \rightarrow \boldsymbol{\Sigma}_o^*$

$\mathrm{w} \in P_{\Sigma o}(\boldsymbol{\mathcal{L}(G)})$

$T(\mathrm{w}): (S \bigcup S.\Sigma_o).E \rightarrow \Delta \cup \{0\}$

$T(w) =$

- $\{0\}$ if $w \notin P_{\Sigma o}(\mathcal{L}(G))$
- $\{N\}$ if $w \in P_{\Sigma o}(\mathcal{L}(G))$, and for any $u \in P_{\Sigma_o}^{-1}(w) \rightarrow f_i \notin u$, for $\forall i = 1,2,\dots,n$
- $\{L_1, L_2, \dots, L_m\}$, $L_i \in \{F_i, A_i\}$
  - $\{F_i\} \in T(w)$ if any $u \in P_{\Sigma_o}^{-1}(w)$ contains the failure f_i.
  - $\{A_i\} \in T(w)$ if $\exists u, u' \in P_{\Sigma_o}^{-1}(w)$ such that $f_i \in u$ and $f_i \notin u'$.

Make it more efficient: Label Propagation

Using this algorithm, some of the queries are possible to be answered using current information in the table:

If a string s is faulty, so are all its possible extensions:

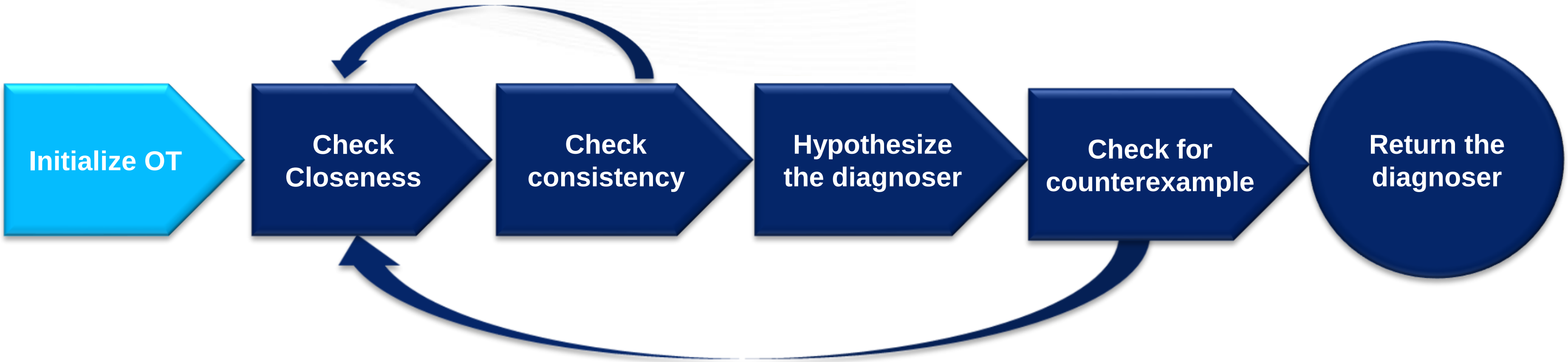
$$[s \in S \cup S.\Sigma\text{: } \mathrm{F_i} \in T(s)] \Rightarrow \left[\forall s' \in ext(s) \cap P_{\Sigma o}\big(\mathcal{L}(G)\big)\text{: } F_i \in T(s')\right]$$

For any string s that is not defined in the system, so are all its extensions:

$$[s \in S \cup S.\Sigma\ \text{:}\ \ T(s) = \{0\}] \Rightarrow [\forall s' \in ext(s)\text{:} T(s') = \{0\}]$$

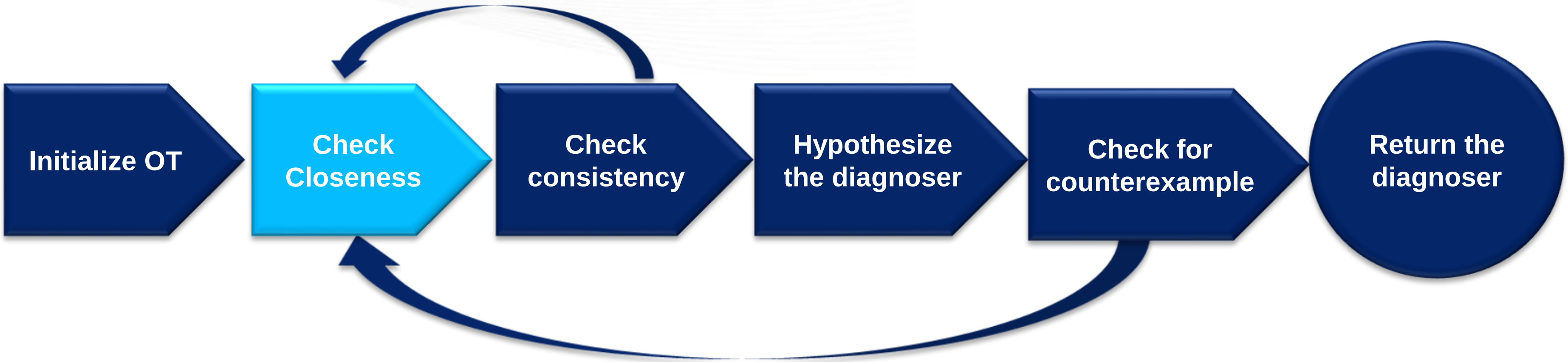
The Proposed Algorithm: Initialization

ACCESS Laboratory



T_1		$\boldsymbol{E}$
		ϵ
S	ϵ	$\boldsymbol{N}$
$S\Sigma_0 - S$	a	$\mathbf{0}$
	b	$\boldsymbol{A_1}$

The Proposed Algorithm: Closeness



An observation table is said to be closed if and only if:

$$\forall t \in S.\Sigma_o, \exists s \in S |\ row(t) = row(s)$$

If the observation table is not closed

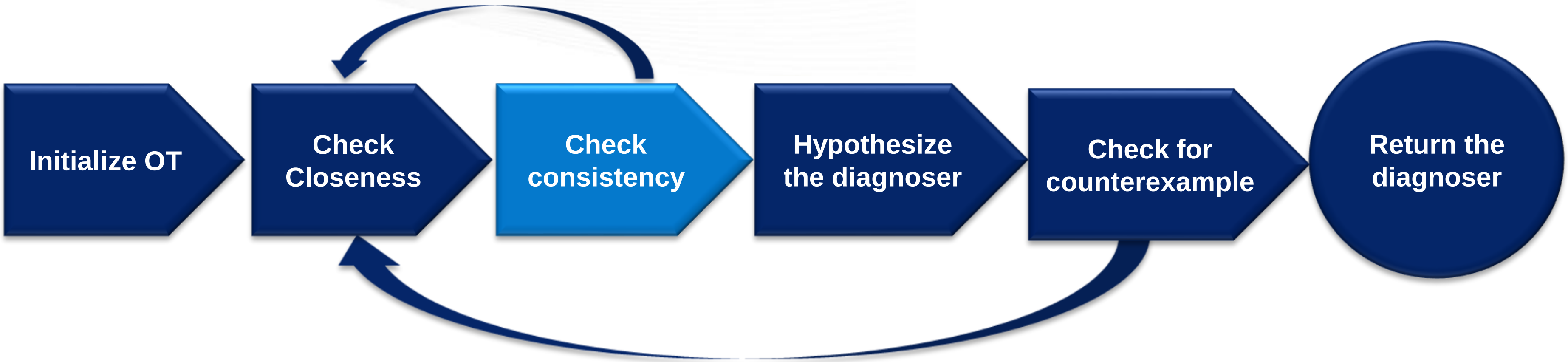
$$\exists s \in S, \exists t \in S.\Sigma_o |\ row(t) \neq row(s)$$

To make the observation table closed, add *t* to *S* and update the table.

T_1		E
		ϵ
S	ϵ	N
$S\Sigma_o - S$	a	0
	b	A_1

T_2		E
		ϵ
S	ϵ	N
	a	0
	b	A_1
$S\Sigma_o - S$	aa	0
	ab	0
	ba	**0**
	bb	$\mathbf{A_1}$

The Proposed Algorithm: Consistency



An observation table is consistent iff:
$\forall s_1, s_2 \in S \; with \; [row(s_1) = row(s_2)]$
$\Rightarrow [row(s_1.\sigma) = row(s_2.\sigma)], \forall \sigma \in \Sigma_o$

If the observation table is not consistent,
$\exists (s_1, s_2) \in S, \exists \sigma \in \Sigma_o, \exists \mathrm{e} \in \mathrm{E} \mid row(s_1) = row(s_2) \; but \; T(s_1 \sigma.\mathrm{e}) \neq T(s_1 \; \sigma.\mathrm{e})$

To make the observation table consistent, add $\sigma.e$ to E and update the table.

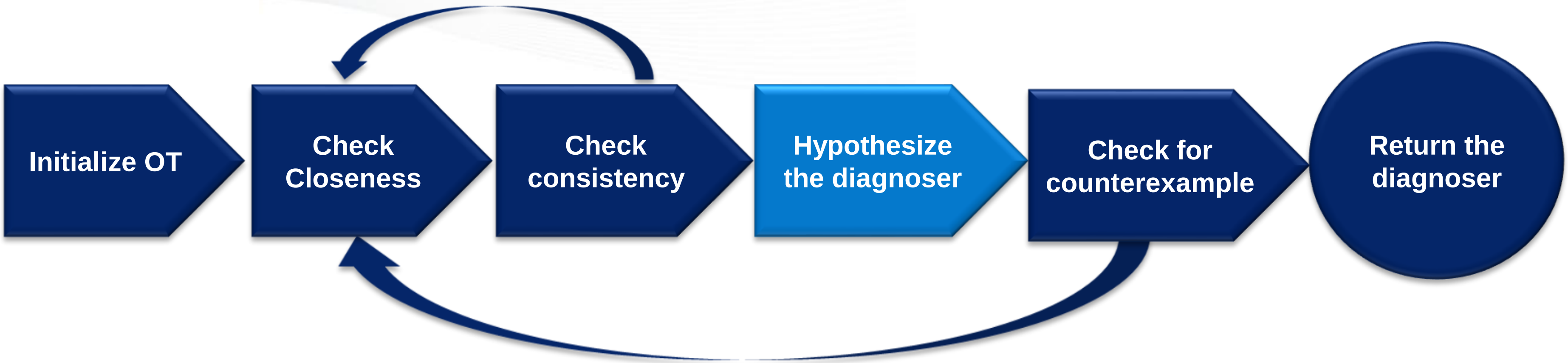
T_6		E: ϵ
S	ϵ	N
	a	0
	b	A_1
	bb	A_1
	bba	N
	bbb	A_1A_2
	$bbba$	A_2
	$bbbb$	F_1
	$bbbaa$	F_2
$S\Sigma_0 - S$	aa	0
	ab	0
	ba	0
	$bbaa$	0
	$bbab$	A_1
	$bbbab$	A_1
	$bbbba$	0
	$bbbbb$	F_1
	$bbbaaa$	$\boldsymbol{F_2}$
	$bbbaab$	**0**

σ

T_7		E: ϵ	E: a
S	ϵ	N	0
	a	0	0
	b	A_1	0
	bb	A_1	N
	bba	N	0
	bbb	A_1A_2	A_2
	$bbba$	A_2	F_2
	$bbbb$	F_1	0
	$bbbaa$	F_2	F_2
$S\Sigma_0 - S$	aa	0	0
	ab	0	0
	ba	0	0
	$bbaa$	0	0
	$bbab$	A_1	**0**
	$bbbab$	A_1	**0**
	$bbbba$	0	0
	$bbbbb$	F_1	**0**
	$bbbaaa$	F_2	$\boldsymbol{F_2}$
	$bbbaab$	0	0

The Proposed Algorithm: Making Hypotheses

ACCESS Laboratory

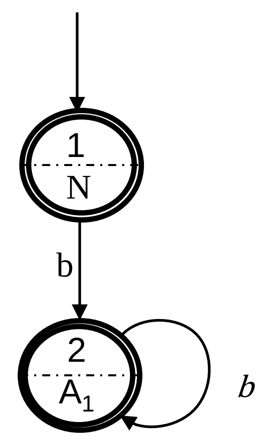


If the observation table is complete (closed and consistent), then we can hypothesize the diagnoser $G_d(T_i)$ based on the observation table OT:

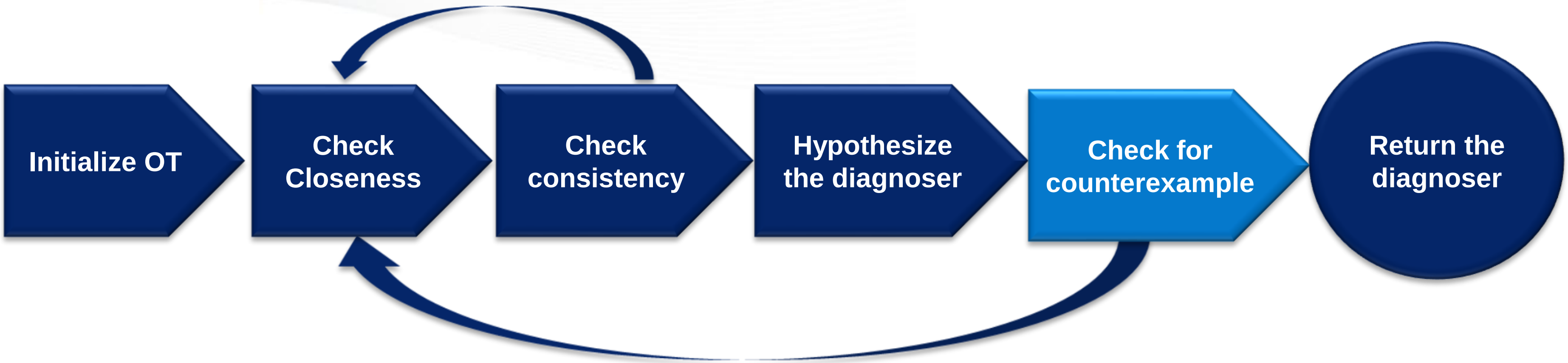
$$G_d(T_i) = (Q_d, \Sigma_d, \Delta_d, \delta_d, h, q_0)$$

- $\Sigma_d = \Sigma_o$
- $\Delta_p = \Delta \cup \{0\}$
- $Q_d = \{row(s) | s \in S\}$
- $q_0 = row(\varepsilon)$
- $h(row(s)) = T(s.\varepsilon)$
- $\delta_d\,(row(s), \sigma) = row(s.\sigma)$

T_2		**E**
		ϵ
S	ϵ	N
	a	0
	b	A_1
$S\Sigma_0 - S$	aa	0
	ab	0
	ba	**0**
	bb	$\mathbf{A_1}$



The Proposed Algorithm: Check for Counterexamples

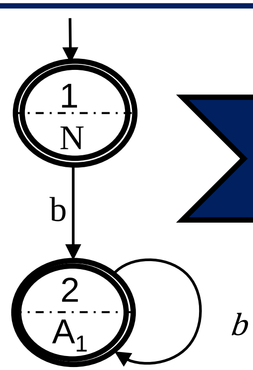


Once the diagnoser was hypothesized, using equivalence queries, the oracle checks for a counter example cex: $cex \in \mathcal{L}(\mathrm{G_d})\backslash P_{\Sigma_o}\left(\mathcal{L}(G)\right) \cup P_{\Sigma_o}\left(\mathcal{L}(G)\right)\backslash\mathcal{L}(\mathrm{G_d})$

If there exist a counterexample cex:

- Add cex and all its prefixes into S, and update table with the new changes.
- This new table again has to be checked for completeness and consistency.

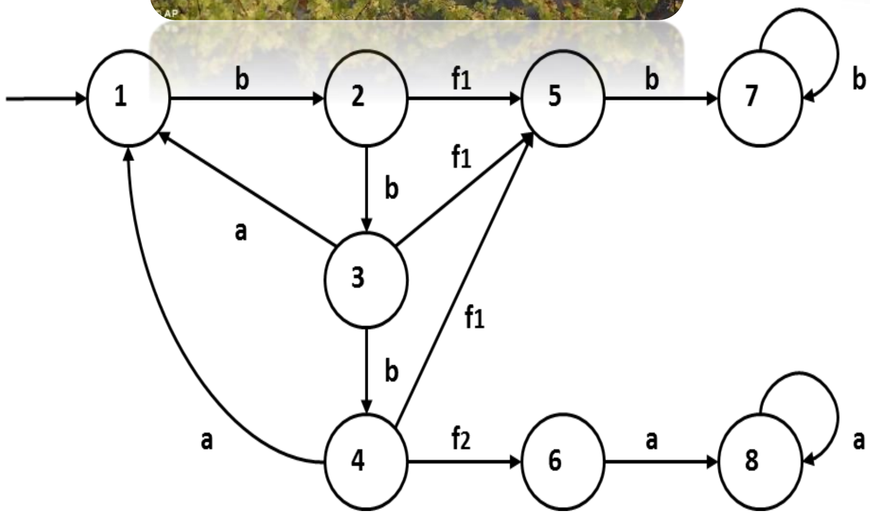
T_2		**E**
		ϵ
S	ϵ	N
	a	0
	b	A_1
$S\Sigma_0 - S$	aa	0
	ab	0
	ba	**0**
	bb	$\mathbf{A_1}$



cex= bba $\in P_{\Sigma_o}\ \left(\mathcal{L}(G)\right)\backslash\mathcal{L}(\mathrm{G_d})$

T_3		**E**
		ϵ
S	ϵ	N
	a	0
	b	A_1
	bb	A_1
	bba	N
$S\Sigma_0 - S$	aa	0
	ab	0
	ba	0
	bbb	$\mathbf{A_1A_2}$
	$bbaa$	**0**
	$bbab$	$\mathbf{A_1}$

Remark: If no counterexample was found, then return the diagnoser.

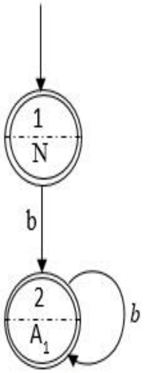


T_1		E
		ϵ
S	ϵ	N
$S\Sigma_0 - S$	a	0
	b	A_1

(a)

T_2		E
		ϵ
S	ϵ	N
	a	0
	b	A_1
$S\Sigma_0 - S$	aa	0
	ab	0
	ba	0
	bb	A_1

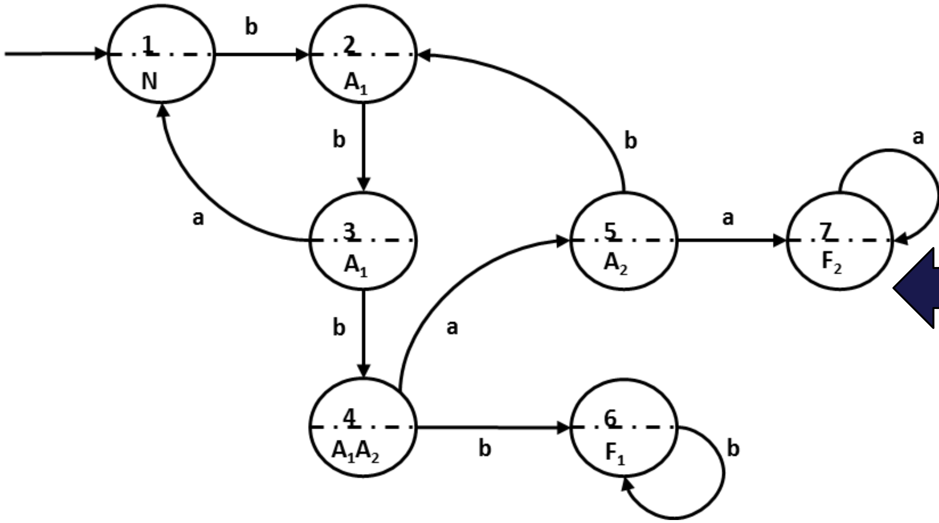
(b)



(c)

T_3		E
		ϵ
S	ϵ	N
	a	0
	b	A_1
	bb	A_1
	bba	N
$S\Sigma_0 - S$	aa	0
	ab	0
	ba	0
	bbb	A_1A_2
	$bbaa$	0
	$bbab$	A_1

(d)



T_4		E
		ϵ
S	ϵ	N
	a	0
	b	A_1
	bb	A_1
	bba	N
	bbb	A_1A_2
$S\Sigma_0 - S$	aa	0
	ab	0
	ba	0
	$bbaa$	0
	$bbab$	A_1
	$bbba$	A_2
	$bbbb$	F_1

(e)

T_5		E
		ϵ
S	ϵ	N
	a	0
	b	A_1
	bb	A_1
	bba	N
	bbb	A_1A_2
	$bbba$	A_2
	$bbbb$	F_1
$S\Sigma_0 - S$	aa	0
	ab	0
	ba	0
	$bbaa$	0
	$bbab$	A_1
	$bbbaa$	F_2
	$bbbab$	A_1
	$bbbba$	0
	$bbbbb$	F_1

(f)

T_6		E
		ϵ
S	ϵ	N
	a	0
	b	A_1
	bb	A_1
	bba	N
	bbb	A_1A_2
	$bbba$	A_2
	$bbbb$	F_1
	$bbbaa$	F_2
$S\Sigma_0 - S$	aa	0
	ab	0
	ba	0
	$bbaa$	0
	$bbab$	A_1
	$bbbab$	A_1
	$bbbba$	0
	$bbbbb$	F_1
	$bbbaaa$	F_2
	$bbbaab$	0

(g)

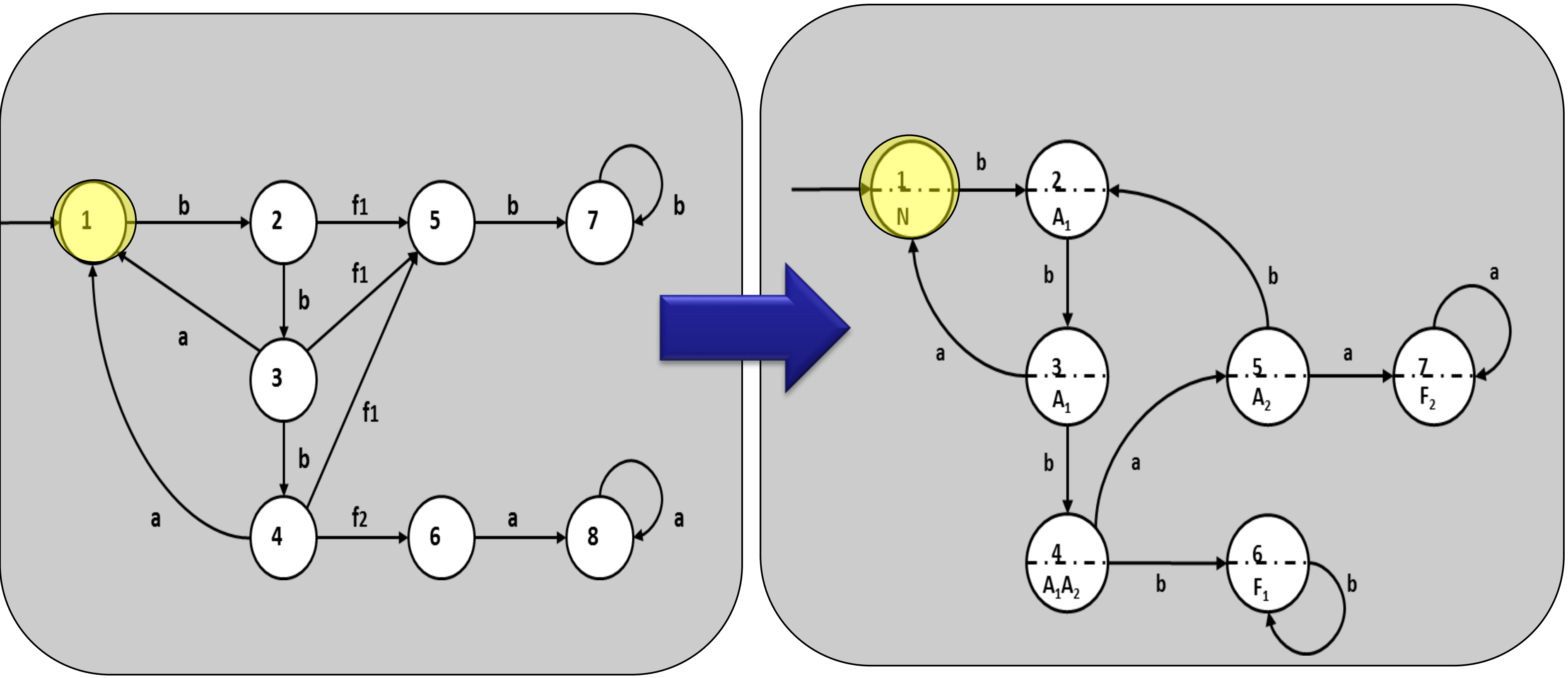
T_7		E	
		ϵ	a
S	ϵ	N	0
	a	0	0
	b	A_1	0
	bb	A_1	N
	bba	N	0
	bbb	A_1A_2	A_2
	$bbba$	A_2	F_2
	$bbbb$	F_1	0
	$bbbaa$	F_2	F_2
$S\Sigma_0 - S$	aa	0	0
	ab	0	0
	ba	0	0
	$bbaa$	0	0
	$bbab$	A_1	0
	$bbbab$	A_1	0
	$bbbba$	0	0
	$bbbbb$	F_1	0
	$bbbaaa$	F_2	F_2
	$bbbaab$	0	0

(h)

The Developed Diagnoser: Online Implementation

Occurred Events: b b a b b f_1 b b

Diagnosis: N A_1 A_1 N A_1 A_1 A_1A_2 F_1



Properties of the Proposed Algorithm

ACCESS Laboratory

Minimality of the Diagnoser

Theorem 1. Let G_d to be the diagnoser constructed by the proposed Algorithm. Then, any other diagnoser, which is consistent with the condition map, T, has more number of states than G_d.

Determinism of the Diagnoser

Theorem 2. The diagnoser G_d, constructed by the proposed Algorithm, is a deterministic finite-state automaton.

Termination of the Algorithm

Theorem 3. The algorithm for constructing the diagnoser G_d, terminates after a finite number of iterations.

[1] W. Bates, A. Karimoddini, M. Karimadini, "***A Learning-based Approach for Diagnosis and Diagnosability of Initially Unknown Discrete Event Systems***," IEEE Transactions on Neural Networks and Learning Systems, 2022, DOI: 10.1109/TNNLS.2022.3204557.

[2] Ira Wendell Bates II, Ali Karimoddini, Mohammad Karimadini, "***Learning a Partially-Known Discrete Event System***," Journal of IEEE ACCESS, Vol. 8, pp. 61806 – 61816, 2020.

[3] M. Karimi, A. Karimoddini, A. White, W. Bates, "**Event-Based Fault Diagnosis for an Unknown Plant**," Proc. of the 55th IEEE Conference on Decision and Control, 2016.

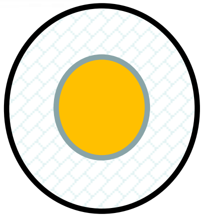
Event-based Fault Diagnosis under Different Levels of Uncertainty

Future Research

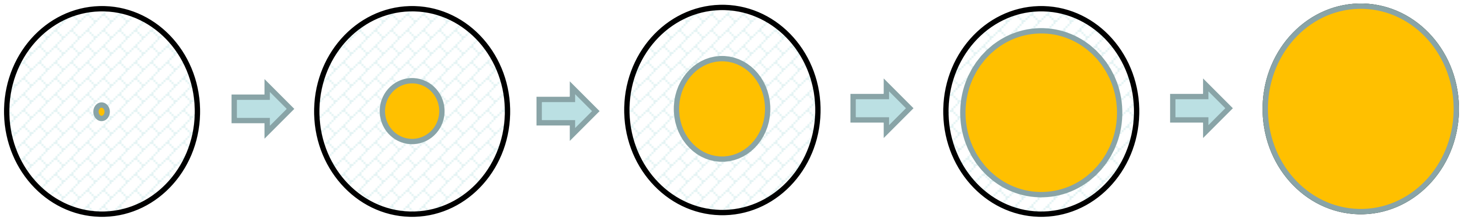
Diagnosis for partially known systems

ACCESS Laboratory

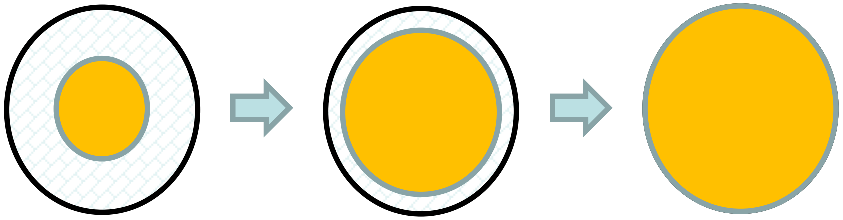
Partially known system



Strategy 1: Ignore the known information and treat the system as an unknown system

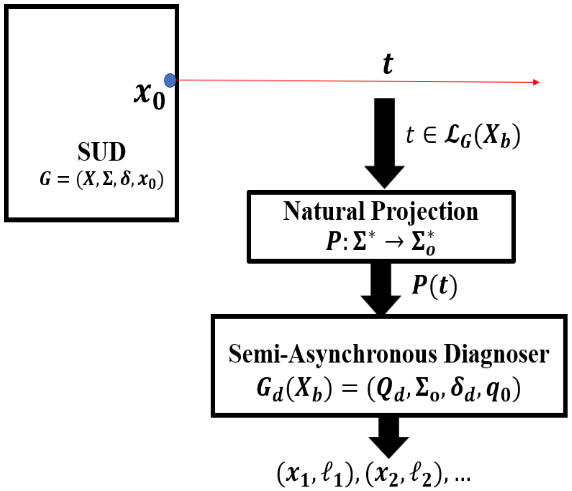


Strategy 2: Take advantage of the information about the known part and actively learn the unknown part



Diagnosis under uncertain initial condition

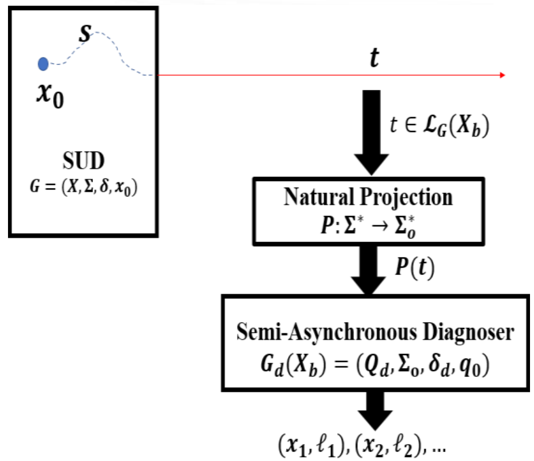
Synchronous Diagnosis



[1] S. Lafortune, "**Diagnosis of discrete event systems**,"Encyclopedia Syst.Control, Springer, pp. 268–275, 2015.

[2] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D.Teneketzis, "**Diagnosability of discrete-event systems**," IEEE Trans. Autom. Control, vol. 40, no. 9, pp. 1555–1575, Sep. 1995.

Asynchronous Diagnosis

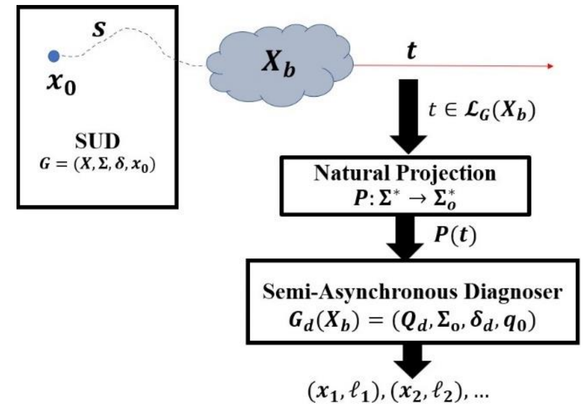


[1] A. White, A. Karimoddini, and R. Su. "**Fault diagnosis of discrete event systems under unknown initial conditions**." *IEEE Transactions on Automatic Control* 64, no. 12 (2019): 5246-5252.

[2] A. White, A. Karimoddini. "**Asynchronous fault diagnosis of discrete event systems**." In *2017 American Control Conference (ACC)*, pp. 3224-3229. IEEE, 2017.

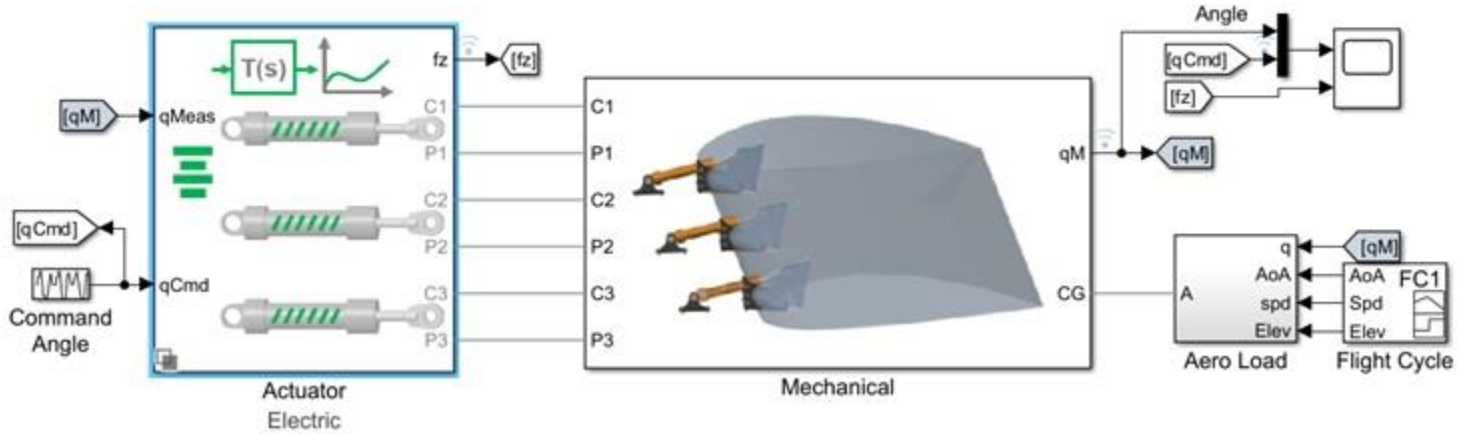
[3] A. White, A. Karimoddini, and M. Karimadini, 2020. **Resilient fault diagnosis under imperfect observations-a need for industry 4.0 era**. *IEEE/CAA Journal of Automatica Sinica*, *7*(5), pp.1279-1288.

Semi-asynchronous Diagnosis

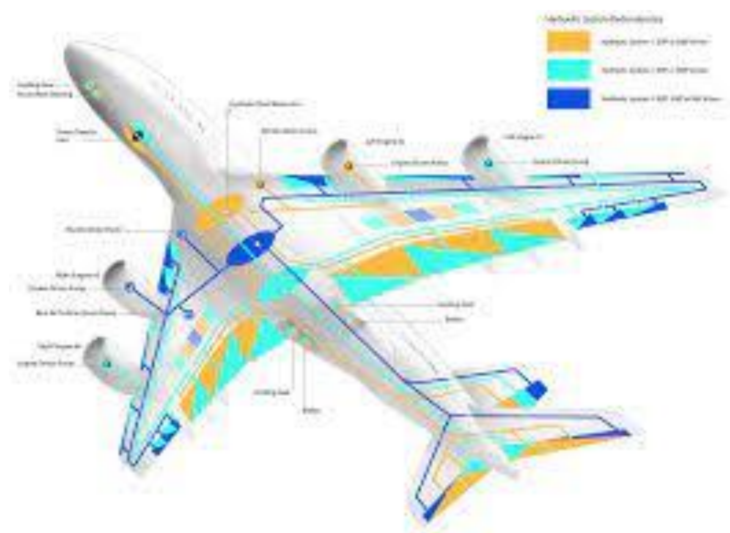


Diagnosis of faults in aircraft actuators

Component-level diagnosis



System-level diagnosis



Acknowledgements

- My collagenous and students at ACCESS Laboratory


Alejandro White, PhD


Wendell Bate, PhD


Jose Matute, PhD


Azmol Fuad

ACCESS Laboratory
http://accesslab.net

- AVIATE Team, particularly TC3 Members

AVIATE CENTER
https://aviate.illinois.edu

- NASA ULI Program

NASA

University Leadership Initiative