

Fault Diagnosis with Dynamic Observers

Franck Cassez

[Joint work with K. Altisen & S. Tripakis]

National ICT Australia & CNRS
Sydney, Australia

October 1st, 2009
NICTA Canberra Research Lab



Outline of the talk

1 Fault Diagnosis for Finite State Systems

- Fault Diagnosis Problem
- Results for Fault Diagnosis
- Algorithms for Fault Diagnosis

2 Static Sensor Minimization Problems

- Results for Static Observers
- Static vs Dynamic Observers

3 Fault Diagnosis with Dynamic Observers

- Dynamic Observers
- Checking Diagnosability with Dynamic Observer
- Cost of a Dynamic Observer
- Minimization Problems with Dynamic Observers
- Synthesis of the Most Permissive Dynamic Observer
- Synthesis of the Optimal Dynamic Observer

Outline

1 Fault Diagnosis for Finite State Systems

- Fault Diagnosis Problem
- Results for Fault Diagnosis
- Algorithms for Fault Diagnosis

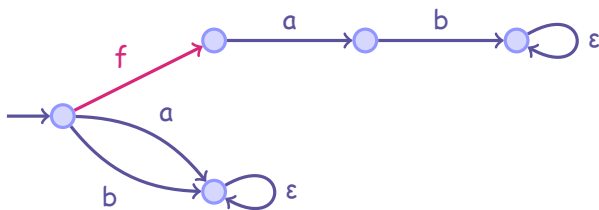
2 Static Sensor Minimization Problems

- Results for Static Observers
- Static vs Dynamic Observers

3 Fault Diagnosis with Dynamic Observers

- Dynamic Observers
- Checking Diagnosability with Dynamic Observer
- Cost of a Dynamic Observer
- Minimization Problems with Dynamic Observers
- Synthesis of the Most Permissive Dynamic Observer
- Synthesis of the Optimal Dynamic Observer

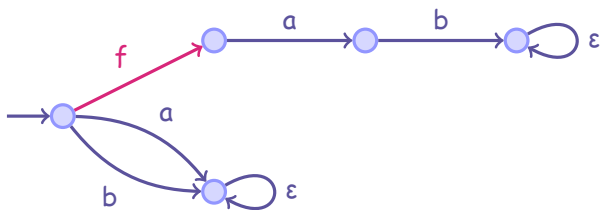
Fault Diagnosis



Given:

- A finite automaton A over $\Sigma^{\epsilon, f} = \Sigma \cup \{\epsilon, f\}$
- f is the fault action, Σ is the set of observable events

Fault Diagnosis



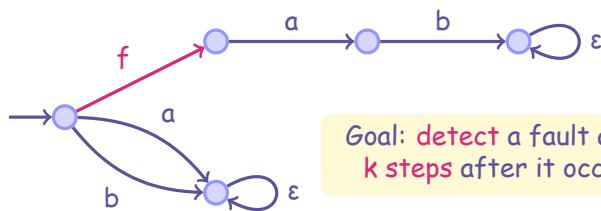
Given:

- A finite automaton A over $\Sigma^{\epsilon, f} = \Sigma \cup \{\epsilon, f\}$
- f is the fault action, Σ is the set of observable events

Define:

- **Faulty_{≥k}(A)**: k -faulty runs that contain f followed by $\geq k$ actions
- **NonFaulty(A)**: Non faulty run that contain no f

Fault Diagnosis



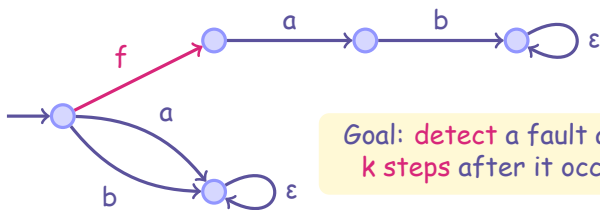
Given:

- A finite automaton A over $\Sigma^{\epsilon, f} = \Sigma \cup \{\epsilon, f\}$
- f is the fault action, Σ is the set of observable events

Define:

- **Faulty _{$\geq k$} (A)**: k -faulty runs that contain f followed by $\geq k$ actions
- **NonFaulty(A)**: Non faulty run that contain no f

Fault Diagnosis



Given:

- A finite automaton A over $\Sigma^{\epsilon, f} = \Sigma \cup \{\epsilon, f\}$
- f is the fault action, Σ is the set of observable events

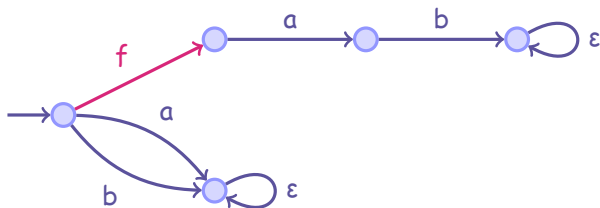
Define:

- **Faulty_{≥k}(A)**: k -faulty runs that contain f followed by $\geq k$ actions
- **NonFaulty(A)**: Non faulty run that contain no f

Purpose of fault diagnosis: given k ,

- you **never** raise an alarm on non-faulty runs
- you **must** raise an alarm on k -faulty runs

Diagnosis Problem



$\text{trace}(\rho)$ = trace of the run ρ (a word in $(\Sigma \cup \{\epsilon, f\})^*$)

$\pi_{\Sigma}(\text{trace}(\rho))$ = projection of the trace on **observable events**

Definition (k-diagnoser)

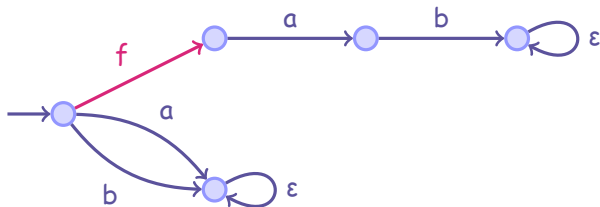
A mapping $D : \Sigma^* \rightarrow \{0, 1\}$ is a **k-diagnoser** for A if:

- for each run $\rho \in \text{NonFaulty}(A)$, $D(\pi_{\Sigma}(\text{trace}(\rho))) = 0$;
- for each run $\rho \in \text{Faulty}_k(A)$, $D(\pi_{\Sigma}(\text{trace}(\rho))) = 1$.

(Σ, k) -Diagnosis Problem

Given $A, \Sigma, k \in \mathbb{N}$, is there a **k-diagnoser** for A ?

Diagnosis Problem



$\text{trace}(\rho)$ = trace of the run ρ (a word in $(\Sigma \cup \{\epsilon, f\})^*$)

$\pi_{\Sigma}(\text{trace}(\rho))$ = projection of the trace on **observable events**

Definition (k-diagnoser)

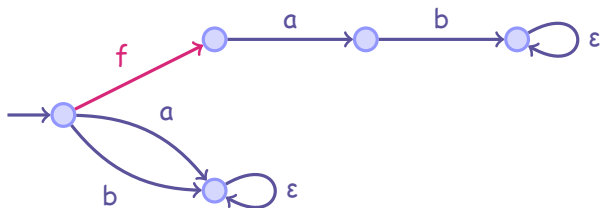
A mapping $D : \Sigma^* \rightarrow \{0, 1\}$ is a **k-diagnoser** for A if:

- for each run $\rho \in \mathbf{NonFaulty}(A)$, $D(\pi_{\Sigma}(\text{trace}(\rho))) = 0$;
- for each run $\rho \in \mathbf{Faulty}_{\geq k}(A)$, $D(\pi_{\Sigma}(\text{trace}(\rho))) = 1$.

(Σ, k) -Diagnosis Problem

Given $A, \Sigma, k \in \mathbb{N}$, is there a **k-diagnoser** for A ?

Diagnosis Problem



$\text{trace}(\rho)$ = trace of the run ρ (a word in $(\Sigma \cup \{\varepsilon, f\})^*$)

$\pi_{\Sigma}(\text{trace}(\rho))$ = projection of the trace on **observable events**

Definition (k-diagnoser)

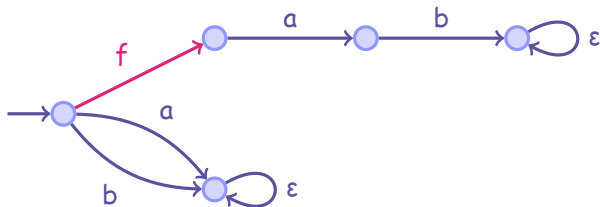
A mapping $D : \Sigma^* \rightarrow \{0, 1\}$ is a **k-diagnoser** for A if:

- for each run $\rho \in \text{NonFaulty}(A)$, $D(\pi_{\Sigma}(\text{trace}(\rho))) = 0$;
- for each run $\rho \in \text{Faulty}_{\geq k}(A)$, $D(\pi_{\Sigma}(\text{trace}(\rho))) = 1$.

(Σ, k) -Diagnosis Problem

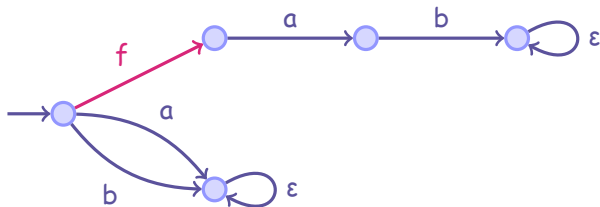
Given $A, \Sigma, k \in \mathbb{N}$, is there a **k-diagnoser** for A ?

Example



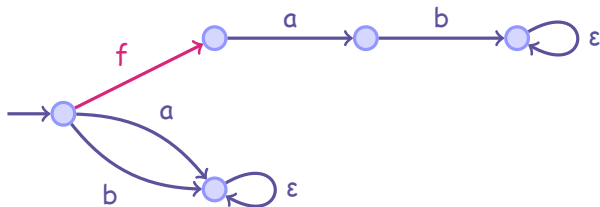
- $\Sigma = \{a, b\}$. Is the plant 1 or 2-diagnosable?
- $\Sigma = \{b\}$. Is the plant 1, 2, k -diagnosable?

Example



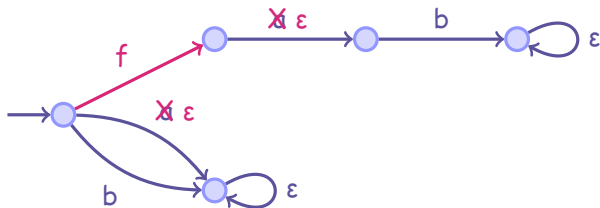
- $\Sigma = \{a, b\}$. Is the plant 1 or 2-diagnosable?
- $\Sigma = \{b\}$. Is the plant 1, 2, k-diagnosable?

Example



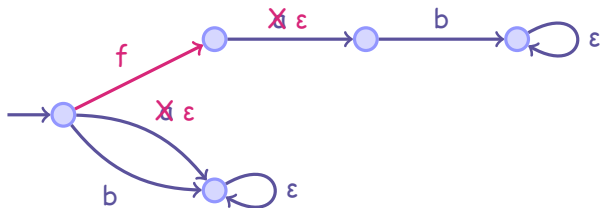
- $\Sigma = \{a, b\}$. Is the plant 1 or 2-diagnosable? **2: Yes**
- $\Sigma = \{b\}$. Is the plant 1, 2, k-diagnosable?

Example



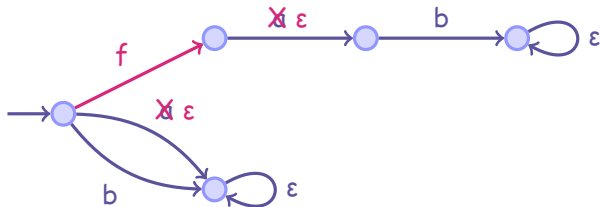
- $\Sigma = \{a, b\}$. Is the plant 1 or 2-diagnosable? 2: Yes
- $\Sigma = \{b\}$. Is the plant 1, 2, k -diagnosable?

Example

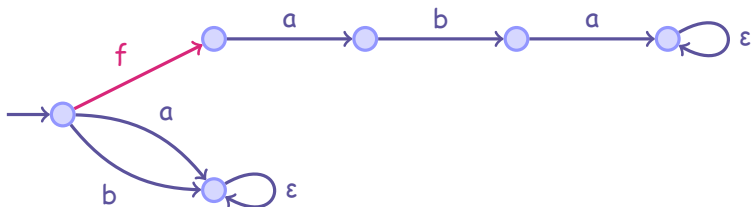


- $\Sigma = \{a, b\}$. Is the plant 1 or 2-diagnosable? 2: Yes
- $\Sigma = \{b\}$. Is the plant 1, 2, k -diagnosable? No

Example

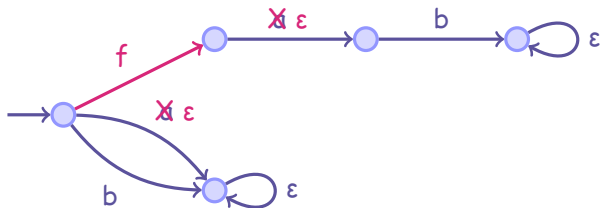


- $\Sigma = \{a, b\}$. Is the plant 1 or 2-diagnosable? 2: Yes
- $\Sigma = \{b\}$. Is the plant 1, 2, k -diagnosable? No

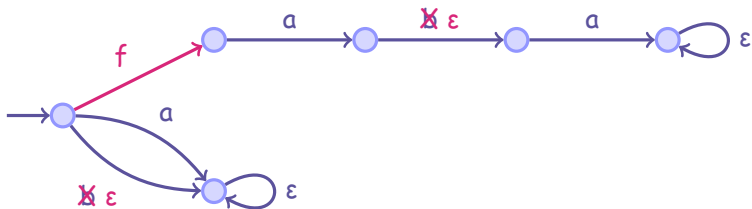


- $\Sigma = \{a\}$. Is the plant 1, 2-diagnosable?

Example

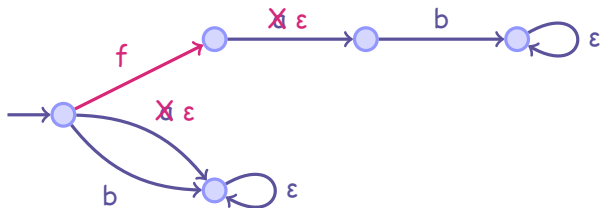


- $\Sigma = \{a, b\}$. Is the plant 1 or 2-diagnosable? 2: Yes
- $\Sigma = \{b\}$. Is the plant 1, 2, k -diagnosable? No

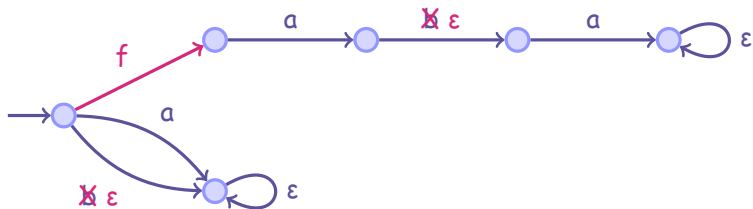


- $\Sigma = \{a\}$. Is the plant 1, 2-diagnosable?

Example



- $\Sigma = \{a, b\}$. Is the plant 1 or 2-diagnosable? 2: Yes
- $\Sigma = \{b\}$. Is the plant 1, 2, k -diagnosable? No



- $\Sigma = \{a\}$. Is the plant 1, 2-diagnosable? **3-diagnosable**

Results for the Fault Diagnosis Problem

[Sampath et al., 1995, Jiang et al., 2001, Yoo et al., 2002]

A is (Σ, k) -diagnosable if there is a k -diagnoser for A .

A is Σ -diagnosable if $\exists k \in \mathbb{N}$ s.t. A is (Σ, k) -diagnosable.

Diagnosis Problems

- (A) Is A Σ -diagnosable?
- (B) If "yes" to (A), compute the minimum k and
- (C) Compute a witness diagnoser.

Results for the Fault Diagnosis Problem

- (A) is in PTIME
- (B) is in PTIME
- (C) is in EXPTIME

A witness diagnoser is an automaton with at most $2^{O(A)}$ states

► Skip algorithms

Results for the Fault Diagnosis Problem

[Sampath et al., 1995, Jiang et al., 2001, Yoo et al., 2002]

A is (Σ, k) -diagnosable if there is a k -diagnoser for A .

A is Σ -diagnosable if $\exists k \in \mathbb{N}$ s.t. A is (Σ, k) -diagnosable.

Diagnosis Problems

- (A) Is A Σ -diagnosable ?
- (B) If "yes" to (A), compute the minimum k and
- (C) Compute a witness diagnoser.

Results for the Fault Diagnosis Problem

- (A) is in PTIME
- (B) is in PTIME
- (C) is in EXPTIME

A witness diagnoser is an automaton with at most $2^{O(A)}$ states

► Skip algorithms

Results for the Fault Diagnosis Problem

[Sampath et al., 1995, Jiang et al., 2001, Yoo et al., 2002]

A is (Σ, k) -diagnosable if there is a k -diagnoser for A .

A is Σ -diagnosable if $\exists k \in \mathbb{N}$ s.t. A is (Σ, k) -diagnosable.

Diagnosis Problems

- (A) Is A Σ -diagnosable?
- (B) If "yes" to (A), compute the minimum k and
- (C) Compute a witness diagnoser.

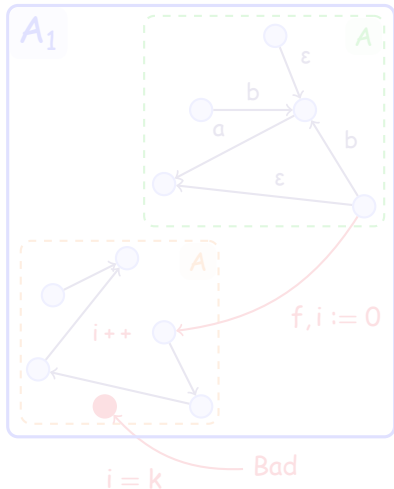
Results for the Fault Diagnosis Problem

- (A) is in PTIME
- (B) is in PTIME
- (C) is in EXPTIME

A witness diagnoser is an automaton with at most $2^{O(A)}$ states

► Skip algorithms

Algorithm for Checking k-Diagnosability



X

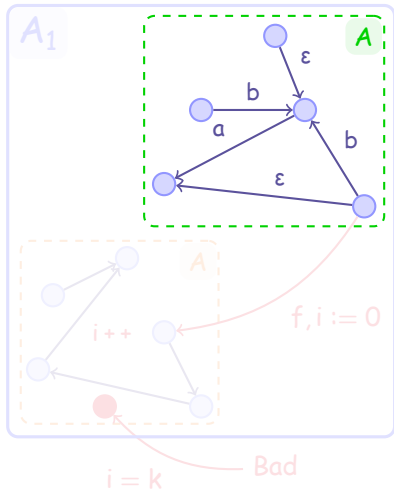


Synchronize* on Σ

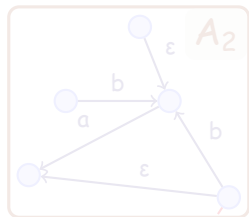
* ϵ does not synchronize

A is k -diagnosable \iff Bad not reachable in $A_1 \times A_2$

Algorithm for Checking k-Diagnosability



\times



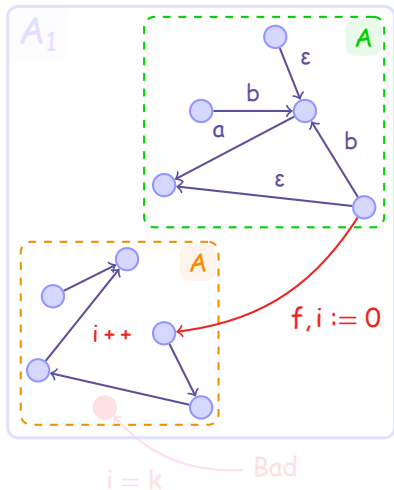
$\not\rightarrow$

Synchronize* on Σ

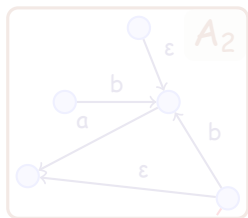
* ϵ does not synchronize

A is k -diagnosable \iff Bad not reachable in $A_1 \times A_2$

Algorithm for Checking k-Diagnosability



\times

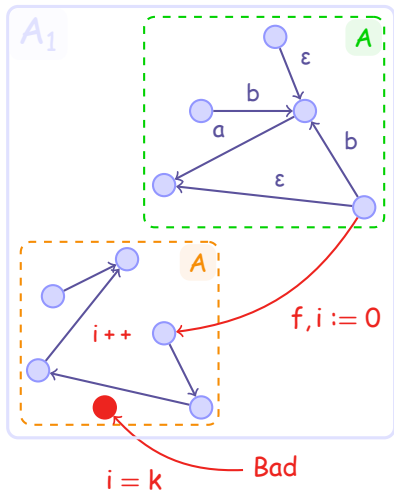


Synchronize* on Σ

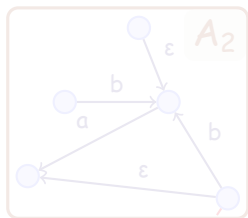
* ϵ does not synchronize

A is k -diagnosable \iff Bad not reachable in $A_1 \times A_2$

Algorithm for Checking k-Diagnosability



\times

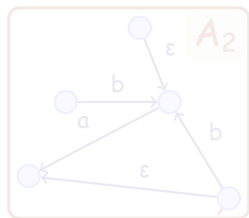
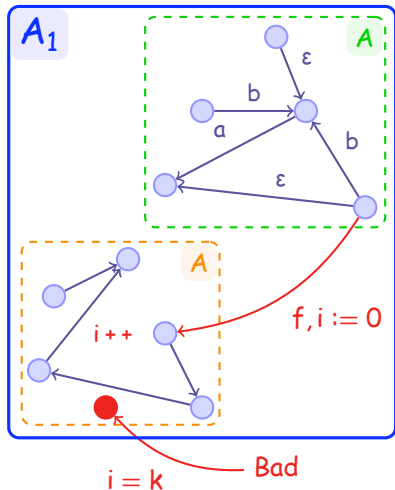


Synchronize* on Σ

* ϵ does not synchronize

A is k -diagnosable \iff Bad not reachable in $A_1 \times A_2$

Algorithm for Checking k-Diagnosability



\times

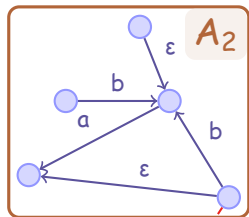
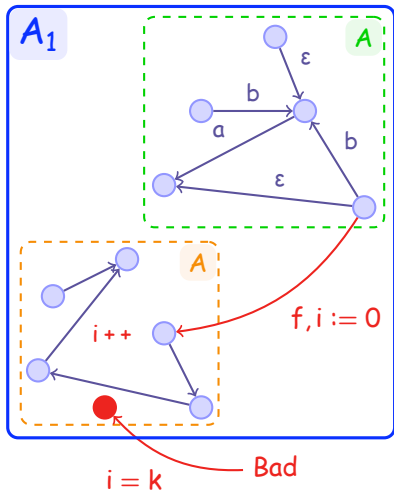


Synchronize* on Σ

* ϵ does not synchronize

A is k -diagnosable \iff Bad not reachable in $A_1 \times A_2$

Algorithm for Checking k-Diagnosability



\times

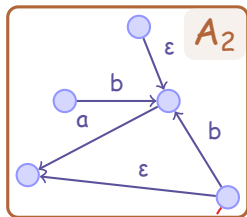
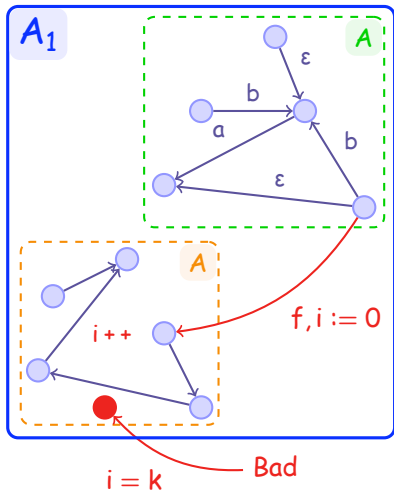


Synchronize* on Σ

* ϵ does not synchronize

A is k -diagnosable \iff Bad not reachable in $A_1 \times A_2$

Algorithm for Checking k-Diagnosability

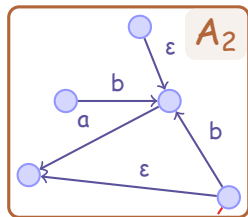
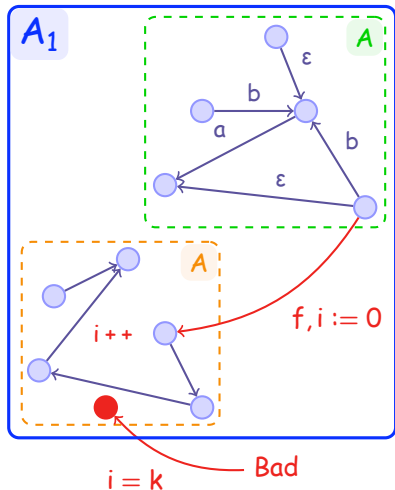


Synchronize* on Σ

* ϵ does not synchronize

A is k -diagnosable \iff Bad not reachable in $A_1 \times A_2$

Algorithm for Checking k-Diagnosability



\times



Synchronize* on Σ

* ϵ does not synchronize

A is k -diagnosable \iff Bad not reachable in $A_1 \times A_2$

Algorithm for Checking Diagnosability

Necessary and Sufficient Condition for Diagnosability

A is **not** Σ -diagnosable $\iff \forall k \in \mathbb{N}^*, A$ is not (Σ, k) -diagnosable

$$\iff \forall k \in \mathbb{N}^*, \begin{cases} \exists \rho \in \text{NonFaulty}(A) \\ \exists \rho' \in \text{Faulty}_{\geq k}(A) \\ \text{and } \pi_{/\Sigma}(\rho) = \pi_{/\Sigma}(\rho') \end{cases}$$

Let $A_1 = (Q \times \{0, 1\}, (q_0, 0), \Sigma^\varepsilon, \rightarrow_1)$ s.t.

- $(q, k) \xrightarrow{1}_1 (q', k')$ iff $q \xrightarrow{1} q'$ and $l \in \Sigma \cup \{\varepsilon\}$ and $k = k'$;
- $(q, k) \xrightarrow{\varepsilon}_1 (q', 1)$ iff $q \xrightarrow{f} q'$

Let $A_2 = (Q, q_0, \Sigma^\varepsilon, \rightarrow_2)$ with $q \xrightarrow{1}_2 q'$ iff $q \xrightarrow{1} q'$ and $l \in \Sigma \cup \{\varepsilon\}$

Büchi condition on $A_1 \times A_2$: infinitely many faulty states and A_1 -actions

Algorithm for Checking Diagnosability

Necessary and Sufficient Condition for Diagnosability

A is **not** Σ -diagnosable $\iff \forall k \in \mathbb{N}^*$, A is not (Σ, k) -diagnosable

$$\iff \forall k \in \mathbb{N}^*, \begin{cases} \exists \rho \in \mathbf{NonFaulty}(A) \\ \exists \rho' \in \mathbf{Faulty}_{\geq k}(A) \\ \text{and } \pi_{/\Sigma}(\rho) = \pi_{/\Sigma}(\rho') \end{cases}$$

Let $A_1 = (Q \times \{0, 1\}, (q_0, 0), \Sigma^\varepsilon, \rightarrow_1)$ s.t.

- $(q, k) \xrightarrow{1}_1 (q', k')$ iff $q \xrightarrow{l} q'$ and $l \in \Sigma \cup \{\varepsilon\}$ and $k = k'$;
- $(q, k) \xrightarrow{\varepsilon}_1 (q', 1)$ iff $q \xrightarrow{f} q'$

Let $A_2 = (Q, q_0, \Sigma^\varepsilon, \rightarrow_2)$ with $q \xrightarrow{1}_2 q'$ iff $q \xrightarrow{l} q'$ and $l \in \Sigma \cup \{\varepsilon\}$

Büchi condition on $A_1 \times A_2$: infinitely many faulty states and A_1 -actions

Algorithm for Checking Diagnosability

Necessary and Sufficient Condition for Diagnosability

A is **not** Σ -diagnosable $\iff \forall k \in \mathbb{N}^*, A$ is not (Σ, k) -diagnosable

$$\iff \forall k \in \mathbb{N}^*, \begin{cases} \exists \rho \in \mathbf{NonFaulty}(A) \\ \exists \rho' \in \mathbf{Faulty}_{\geq k}(A) \\ \text{and } \pi_{/\Sigma}(\rho) = \pi_{/\Sigma}(\rho') \end{cases}$$

Let $A_1 = (Q \times \{0, 1\}, (q_0, 0), \Sigma^\varepsilon, \rightarrow_1)$ s.t.

- $(q, k) \xrightarrow{l}_1 (q', k')$ iff $q \xrightarrow{l} q'$ and $l \in \Sigma \cup \{\varepsilon\}$ and $k = k'$;
- $(q, k) \xrightarrow{\varepsilon}_1 (q', 1)$ iff $q \xrightarrow{f} q'$

Let $A_2 = (Q, q_0, \Sigma^\varepsilon, \rightarrow_2)$ with $q \xrightarrow{l}_2 q'$ iff $q \xrightarrow{l} q'$ and $l \in \Sigma \cup \{\varepsilon\}$

Büchi condition on $A_1 \times A_2$: infinitely many faulty states and A_1 -actions

Algorithm for Checking Diagnosability

Necessary and Sufficient Condition for Diagnosability

A is **not** Σ -diagnosable $\iff \forall k \in \mathbb{N}^*$, A is not (Σ, k) -diagnosable

$$\iff \forall k \in \mathbb{N}^*, \begin{cases} \exists \rho \in \mathbf{NonFaulty}(A) \\ \exists \rho' \in \mathbf{Faulty}_{\geq k}(A) \\ \text{and } \pi_{/\Sigma}(\rho) = \pi_{/\Sigma}(\rho') \end{cases}$$

Let $A_1 = (Q \times \{0, 1\}, (q_0, 0), \Sigma^\varepsilon, \rightarrow_1)$ s.t.

- $(q, k) \xrightarrow{l}_1 (q', k')$ iff $q \xrightarrow{l} q'$ and $l \in \Sigma \cup \{\varepsilon\}$ and $k = k'$;
- $(q, k) \xrightarrow{\varepsilon}_1 (q', 1)$ iff $q \xrightarrow{f} q'$

Let $A_2 = (Q, q_0, \Sigma^\varepsilon, \rightarrow_2)$ with $q \xrightarrow{l}_2 q'$ iff $q \xrightarrow{l} q'$ and $l \in \Sigma \cup \{\varepsilon\}$

Büchi condition on $A_1 \times A_2$: infinitely many faulty states and A_1 -actions

Algorithm for Checking Diagnosability

Necessary and Sufficient Condition for Diagnosability

A is **not** Σ -diagnosable $\iff \forall k \in \mathbb{N}^*, A$ is not (Σ, k) -diagnosable

$$\iff \forall k \in \mathbb{N}^*, \begin{cases} \exists \rho \in \mathbf{NonFaulty}(A) \\ \exists \rho' \in \mathbf{Faulty}_{\geq k}(A) \\ \text{and } \pi_{/\Sigma}(\rho) = \pi_{/\Sigma}(\rho') \end{cases}$$

Let $A_1 = (Q \times \{0, 1\}, (q_0, 0), \Sigma^\varepsilon, \rightarrow_1)$ s.t.

- $(q, k) \xrightarrow{l}_1 (q', k')$ iff $q \xrightarrow{l} q'$ and $l \in \Sigma \cup \{\varepsilon\}$ and $k = k'$;
- $(q, k) \xrightarrow{\varepsilon}_1 (q', 1)$ iff $q \xrightarrow{f} q'$

Let $A_2 = (Q, q_0, \Sigma^\varepsilon, \rightarrow_2)$ with $q \xrightarrow{l}_2 q'$ iff $q \xrightarrow{l} q'$ and $l \in \Sigma \cup \{\varepsilon\}$

Büchi condition on $A_1 \times A_2$: infinitely many **faulty** states and **A_1 -actions**

Algorithm for Checking Diagnosability

Necessary and Sufficient Condition for Diagnosability

A is **not** Σ -diagnosable $\iff \forall k \in \mathbb{N}^*$, A is not (Σ, k) -diagnosable

$$\iff \forall k \in \mathbb{N}^*, \begin{cases} \exists \rho \in \text{NonFaulty}(A) \\ \exists \rho' \in \text{Faulty}_{\geq k}(A) \\ \text{and } \pi_{/\Sigma}(\rho) = \pi_{/\Sigma}(\rho') \end{cases}$$

Let $A_1 = (Q \times \{0, 1\}, (q_0, 0), \Sigma^\varepsilon, \rightarrow_1)$ s.t.

- $(q, k) \xrightarrow{1}_1 (q', k')$ iff $q \xrightarrow{1} q'$ and $l \in \Sigma \cup \{\varepsilon\}$ and $k = k'$;
- $(q, k) \xrightarrow{\varepsilon}_1 (q', 1)$ iff $q \xrightarrow{f} q'$

Let $A_2 = (Q, q_0, \Sigma^\varepsilon, \rightarrow_2)$ with $q \xrightarrow{1}_2 q'$ iff $q \xrightarrow{1} q'$ and $l \in \Sigma \cup \{\varepsilon\}$

Büchi condition on $A_1 \times A_2$: infinitely many faulty states and A_1 -actions

Theorem ([FI'08])

$\mathcal{L}^\omega(A_1 \times A_2) \neq \emptyset \iff A$ is not diagnosable. [Can be checked in PTIME]

The minimum k s.t. A is k -diagnosable can be computed in PTIME.

Outline

1 Fault Diagnosis for Finite State Systems

- Fault Diagnosis Problem
- Results for Fault Diagnosis
- Algorithms for Fault Diagnosis

2 Static Sensor Minimization Problems

- Results for Static Observers
- Static vs Dynamic Observers

3 Fault Diagnosis with Dynamic Observers

- Dynamic Observers
- Checking Diagnosability with Dynamic Observer
- Cost of a Dynamic Observer
- Minimization Problems with Dynamic Observers
- Synthesis of the Most Permissive Dynamic Observer
- Synthesis of the Optimal Dynamic Observer

Minimization of the set of Observable Events

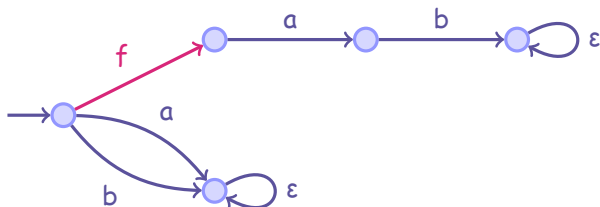
Goal: find $\Sigma_o \subseteq \Sigma$ s.t. A is Σ_o -diagnosable and **minimize** $|\Sigma_o|$



- minimum $\Sigma_o = \{a, b\}$ and $(\Sigma_o, 2)$ -diagnosable

Minimization of the set of Observable Events

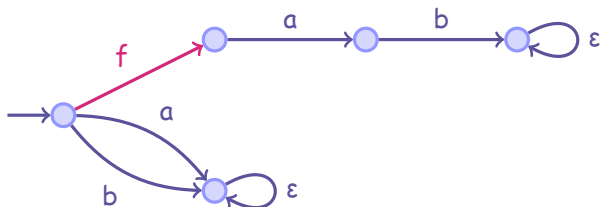
Goal: find $\Sigma_o \subseteq \Sigma$ s.t. A is Σ_o -diagnosable and **minimize** $|\Sigma_o|$



- minimum $\Sigma_o = \{a, b\}$ and $(\Sigma_o, 2)$ -diagnosable

Minimization of the set of Observable Events

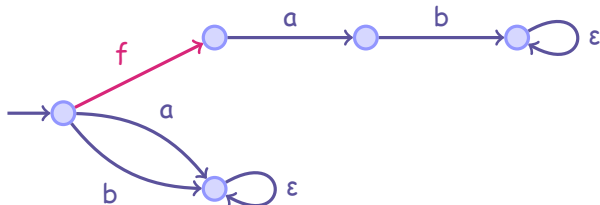
Goal: find $\Sigma_o \subseteq \Sigma$ s.t. A is Σ_o -diagnosable and **minimize** $|\Sigma_o|$



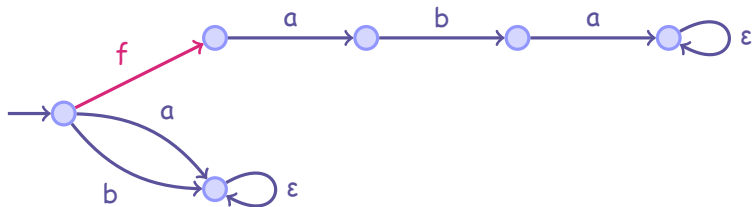
- minimum $\Sigma_o = \{a, b\}$ and $(\Sigma_o, 2)$ -diagnosable

Minimization of the set of Observable Events

Goal: find $\Sigma_o \subseteq \Sigma$ s.t. A is Σ_o -diagnosable and **minimize** $|\Sigma_o|$



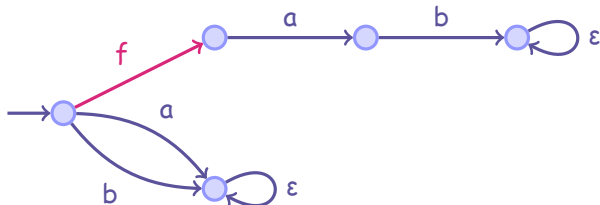
- minimum $\Sigma_o = \{a, b\}$ and $(\Sigma_o, 2)$ -diagnosable



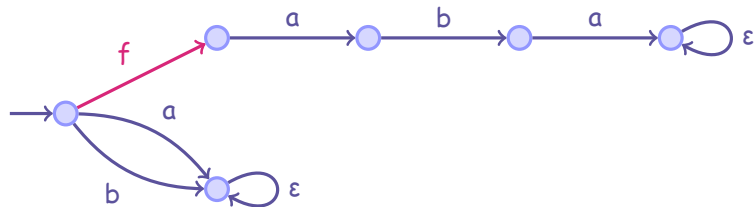
- $\Sigma_o = \{a\}$ is minimal and $(\Sigma_o, 3)$ -diagnosable

Minimization of the set of Observable Events

Goal: find $\Sigma_o \subseteq \Sigma$ s.t. A is Σ_o -diagnosable and **minimize** $|\Sigma_o|$



- minimum $\Sigma_o = \{a, b\}$ and $(\Sigma_o, 2)$ -diagnosable



- $\Sigma_o = \{a\}$ is minimal and $(\Sigma_o, 3)$ -diagnosable

Results for Sensor Minimization Problems

Problem 1: Static Minimization Problem

Input: $A, n \in \mathbb{N}^*$ s.t. $n \leq |\Sigma|$.

Problem: Is there any $\Sigma_o \subseteq \Sigma, |\Sigma_o| \leq n$, s.t. A is Σ_o -diagnosable ?

Results:

- Static Minimization Problem is NP-complete

[Yoo et al., 2001, ACSD'07]

- Mask Version of Static Minimization Problem is also NP-complete

[ACSD'07]

Results for Sensor Minimization Problems

Problem 1: Static Minimization Problem

Input: $A, n \in \mathbb{N}^*$ s.t. $n \leq |\Sigma|$.

Problem: Is there any $\Sigma_o \subseteq \Sigma, |\Sigma_o| \leq n$, s.t. A is Σ_o -diagnosable ?

Results:

- **Static Minimization Problem is NP-complete**

[Yoo et al., 2001, ACSD'07]

- **Mask Version of Static Minimization Problem is also NP-complete**

[ACSD'07]

Results for Sensor Minimization Problems

Problem 1: Static Minimization Problem

Input: $A, n \in \mathbb{N}^*$ s.t. $n \leq |\Sigma|$.

Problem: Is there any $\Sigma_o \subseteq \Sigma, |\Sigma_o| \leq n$, s.t. A is Σ_o -diagnosable ?

Results:

- **Static Minimization Problem is NP-complete**
[Yoo et al., 2001, ACSD'07]
- **Mask Version of Static Minimization Problem is also NP-complete**
[ACSD'07]

Results for Sensor Minimization Problems

Problem 1: Static Minimization Problem

Input: $A, n \in \mathbb{N}^*$ s.t. $n \leq |\Sigma|$.

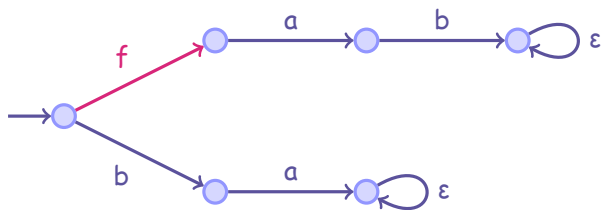
Problem: Is there any $\Sigma_o \subseteq \Sigma, |\Sigma_o| \leq n$, s.t. A is Σ_o -diagnosable ?

Results:

- **Static Minimization Problem is NP-complete** [Yoo et al., 2001, ACSD'07]
- **Mask Version of Static Minimization Problem is also NP-complete** [ACSD'07]

Can we do any better/cheaper?

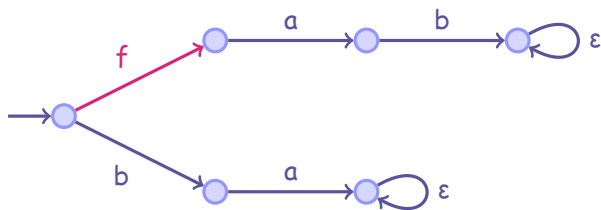
Why Dynamic Observations ?



- **Static** observer: **fixed set** of observable events
- Static observation: $|\Sigma_o| \geq 2$, $\Sigma_o = \{a, b\}$; **$(\{a, b\}, 1)$ -diagnosable**

Dynamic observer: chooses **dynamically** the set of observable events

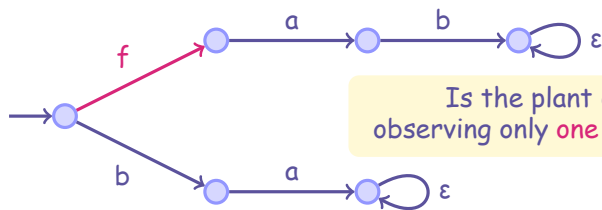
Why Dynamic Observations ?



- **Static** observer: **fixed set** of observable events
- Static observation: $|\Sigma_o| \geq 2$, $\Sigma_o = \{a, b\}$; $(\{a, b\}, 1)$ -diagnosable

Dynamic observer: chooses **dynamically** the set of observable events

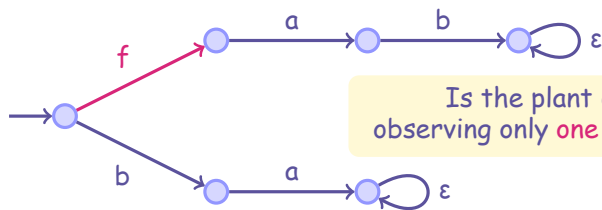
Why Dynamic Observations ?



- **Static** observer: **fixed set** of observable events
- Static observation: $|\Sigma_o| \geq 2$, $\Sigma_o = \{a, b\}$; **($\{a, b\}, 1$)-diagnosable**

Dynamic observer: chooses **dynamically** the set of observable events

Why Dynamic Observations ?

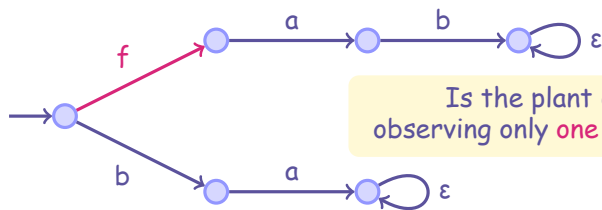


- **Static** observer: **fixed set** of observable events
- Static observation: $|\Sigma_o| \geq 2$, $\Sigma_o = \{a, b\}$; $(\{a, b\}, 1)$ -diagnosable

Dynamic observer: chooses **dynamically** the set of observable events

- 1 observe only **a**

Why Dynamic Observations ?

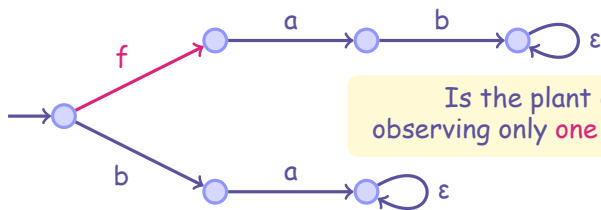


- **Static** observer: **fixed set** of observable events
- Static observation: $|\Sigma_o| \geq 2$, $\Sigma_o = \{a, b\}$; $(\{a, b\}, 1)$ -diagnosable

Dynamic observer: chooses **dynamically** the set of observable events

- 1 observe only **a**
- 2 once an **a** has been observed, observe only **b**

Why Dynamic Observations ?



- **Static** observer: **fixed set** of observable events
- Static observation: $|\Sigma_o| \geq 2$, $\Sigma_o = \{a, b\}$; $(\{a, b\}, 1)$ -diagnosable

Dynamic observer: chooses **dynamically** the set of observable events

- 1 observe only **a**
- 2 once an **a** has been observed, observe only **b**
- 3 if **a.b** occurs **diagnose** a fault \implies dynamically 2-diagnosable

Outline

1 Fault Diagnosis for Finite State Systems

- Fault Diagnosis Problem
- Results for Fault Diagnosis
- Algorithms for Fault Diagnosis

2 Static Sensor Minimization Problems

- Results for Static Observers
- Static vs Dynamic Observers

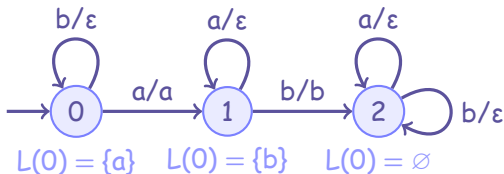
3 Fault Diagnosis with Dynamic Observers

- Dynamic Observers
- Checking Diagnosability with Dynamic Observer
- Cost of a Dynamic Observer
- Minimization Problems with Dynamic Observers
- Synthesis of the Most Permissive Dynamic Observer
- Synthesis of the Optimal Dynamic Observer

Dynamic Observers

Definition (Dynamic Observer)

A **dynamic observer** $Obs = (S, s_0, \Sigma, \delta, L)$ is a complete and deterministic labeled **transducer** s.t. $\forall s \in S, \forall a \in \Sigma$, if $a \notin L(s)$ then $\delta(s, a) = s$.



Definition ((Obs, k)-diagnoser)

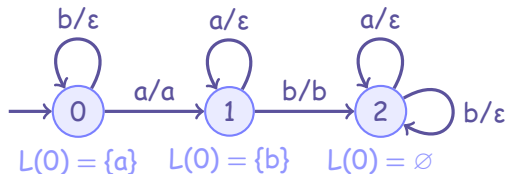
$D : \Sigma^* \rightarrow \{0, 1\}$ is an **(Obs, k)-diagnoser** for A if:

- for each run $\rho \in \text{NonFaulty}(A)$, $D(\text{Obs}(\pi_{\Sigma}(\text{trace}(\rho)))) = 0$ and
- for each run $\rho \in \text{Faulty}_k(A)$, $D(\text{Obs}(\pi_{\Sigma}(\text{trace}(\rho)))) = 1$.

Dynamic Observers

Definition (Dynamic Observer)

A **dynamic observer** $Obs = (S, s_0, \Sigma, \delta, L)$ is a complete and deterministic labeled **transducer** s.t. $\forall s \in S, \forall a \in \Sigma$, if $a \notin L(s)$ then $\delta(s, a) = s$.



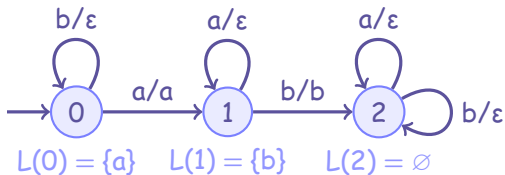
Definition ((Obs, k)-diagnoser)

$D : \Sigma^* \rightarrow \{0, 1\}$ is an **(Obs, k)-diagnoser** for A if:

- for each run $\rho \in \mathbf{NonFaulty}(A)$, $D(Obs(\pi_{/\Sigma}(\text{trace}(\rho)))) = 0$ and
- for each run $\rho \in \mathbf{Faulty}_{\geq k}(A)$, $D(Obs(\pi_{/\Sigma}(\text{trace}(\rho)))) = 1$.

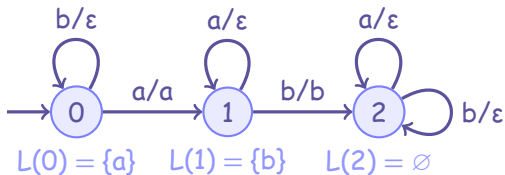
Dynamic Observer and Diagnosability

Obs

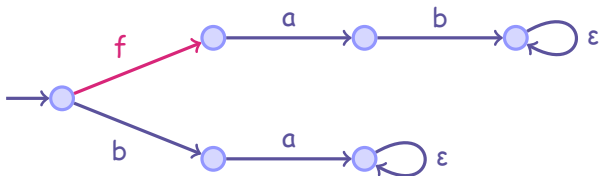


Dynamic Observer and Diagnosability

Obs

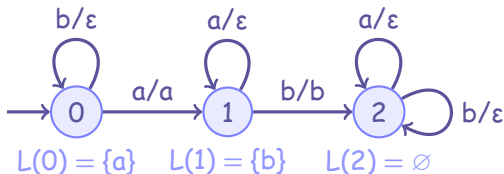


\mathcal{B}

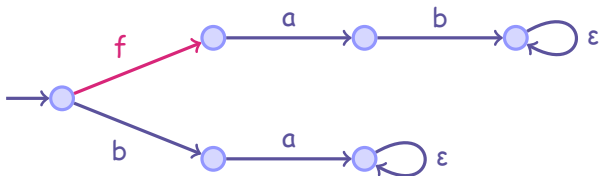


Dynamic Observer and Diagnosability

Obs



\mathcal{B}

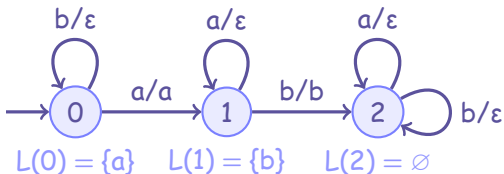


\mathcal{B} is $(\text{Obs}, 2)$ -diagnosable.

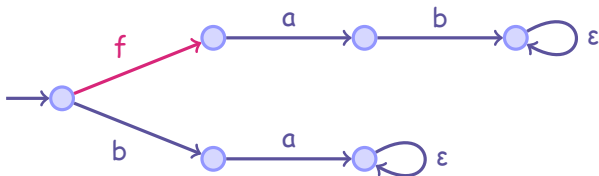
Define $D(a.b.p) = 1$ and $D(p) = 0$ otherwise. D is 2-diagnoser.

Dynamic Observer and Diagnosability

Obs



\mathcal{B}



\mathcal{B} is $(\text{Obs}, 2)$ -diagnosable.

Define $D(a.b.p) = 1$ and $D(p) = 0$ otherwise. D is 2-diagnoser.

Obs observes only **one event** at each point in time.

Checking Obs-diagnosability

[ACSD'07, FI'08]

(Obs, k)-diagnosability: A is (Obs, k)-diagnosable if there is an (Obs, k)-diagnoser for A

Obs-diagnosability: A is Obs-diagnosable if $\exists k \in \mathbb{N}$ s.t. A is (Obs, k)-diagnosable

Finite Obs-diagnosability Problem

Input: A, a finite state observer Obs.

Problem: Is A Obs-diagnosable ?

To check **Obs-diagnosability**, build a product $A \otimes \text{Obs}$:

- initial state: (q_0, s_0)
- for $\lambda \in \Sigma$, $(q, s) \xrightarrow{\beta} (q', s')$ iff $q \xrightarrow{\lambda} q'$, $s \xrightarrow{\lambda/\beta} s'$,
- for $\lambda \in \{\varepsilon, f\}$, $(q, s) \xrightarrow{\lambda} (q', s)$ iff $q \xrightarrow{\lambda} q'$ (Obs cannot see $\{\varepsilon, f\}$).

A is Obs-diagnosable iff $A \otimes \text{Obs}$ is Σ -diagnosable

Checking Obs-diagnosability

[ACSD'07, FI'08]

(Obs, k)-diagnosability: A is (Obs, k)-diagnosable if there is an (Obs, k)-diagnoser for A

Obs-diagnosability: A is Obs-diagnosable if $\exists k \in \mathbb{N}$ s.t. A is (Obs, k)-diagnosable

Finite Obs-diagnosability Problem

Input: A, a finite state observer Obs.

Problem: Is A Obs-diagnosable ?

To check Obs-diagnosability, build a product $A \otimes \text{Obs}$:

- initial state: (q_0, s_0)
- for $\lambda \in \Sigma$, $(q, s) \xrightarrow{\lambda} (q', s')$ iff $q \xrightarrow{\lambda} q'$, $s \xrightarrow{\lambda/\beta} s'$,
- for $\lambda \in \{\varepsilon, f\}$, $(q, s) \xrightarrow{\lambda} (q', s)$ iff $q \xrightarrow{\lambda} q'$ (Obs cannot see $\{\varepsilon, f\}$).

A is Obs-diagnosable iff $A \otimes \text{Obs}$ is Σ -diagnosable

Checking Obs-diagnosability

[ACSD'07, FI'08]

(Obs, k)-diagnosability: A is (Obs, k)-diagnosable if there is an (Obs, k)-diagnoser for A

Obs-diagnosability: A is Obs-diagnosable if $\exists k \in \mathbb{N}$ s.t. A is (Obs, k)-diagnosable

Finite Obs-diagnosability Problem

Input: A, a finite state observer Obs.

Problem: Is A Obs-diagnosable ?

To check **Obs-diagnosability**, build a product $A \otimes \text{Obs}$:

- initial state: (q_0, s_0)
- for $\lambda \in \Sigma$, $(q, s) \xrightarrow{\beta} (q', s')$ iff $q \xrightarrow{\lambda} q'$, $s \xrightarrow{\lambda/\beta} s'$,
- for $\lambda \in \{\varepsilon, f\}$, $(q, s) \xrightarrow{\lambda} (q', s)$ iff $q \xrightarrow{\lambda} q'$ (Obs cannot see $\{\varepsilon, f\}$).

A is Obs-diagnosable iff $A \otimes \text{Obs}$ is Σ -diagnosable

Checking Obs-diagnosability

[ACSD'07, FI'08]

(Obs, k)-diagnosability: A is (Obs, k)-diagnosable if there is an (Obs, k)-diagnoser for A

Obs-diagnosability: A is Obs-diagnosable if $\exists k \in \mathbb{N}$ s.t. A is (Obs, k)-diagnosable

Finite Obs-diagnosability Problem

Input: A, a finite state observer Obs.

Problem: Is A Obs-diagnosable ?

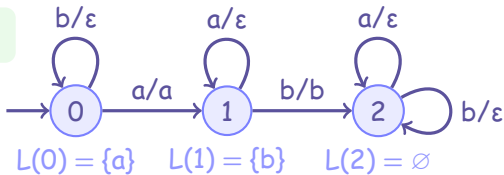
To check **Obs-diagnosability**, build a product $A \otimes \text{Obs}$:

- initial state: (q_0, s_0)
- for $\lambda \in \Sigma$, $(q, s) \xrightarrow{\lambda} (q', s')$ iff $q \xrightarrow{\lambda} q'$, $s \xrightarrow{\lambda/\beta} s'$,
- for $\lambda \in \{\varepsilon, f\}$, $(q, s) \xrightarrow{\lambda} (q', s)$ iff $q \xrightarrow{\lambda} q'$ (Obs cannot see $\{\varepsilon, f\}$).

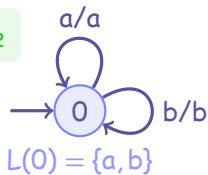
A is Obs-diagnosable iff $A \otimes \text{Obs}$ is Σ -diagnosable

How to Compare Dynamic Observers ?

O_1



O_2

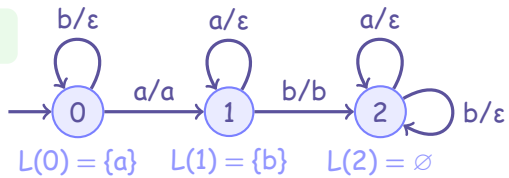


- **Switching** a-sensor on for $a \in \Sigma$ costs **1** per time unit
- On input w , maximum average cost is
 - ▶ 1 for O_1
 - ▶ 2 for O_2
- O_1 observes **less events** than O_2 in the long run
- O_1 is **less expensive** than O_2

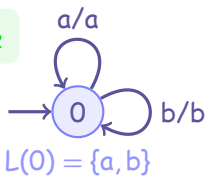
Need to define a **cost** measure for dynamic observers

How to Compare Dynamic Observers ?

O_1



O_2

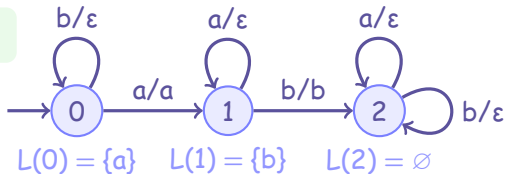


- **Switching** a-sensor on for $a \in \Sigma$ **costs 1** per time unit
- On input w , **maximum average cost** is
 - ▶ 1 for O_1
 - ▶ 2 for O_2
- O_1 observes **less events** than O_2 in the long run
- O_1 is **less expensive** than O_2

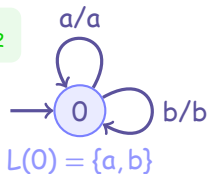
Need to define a **cost** measure for dynamic observers

How to Compare Dynamic Observers ?

O_1



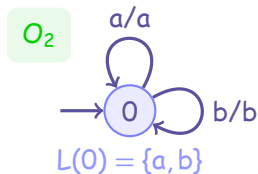
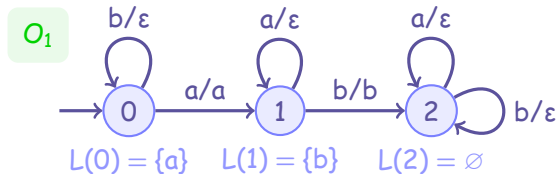
O_2



- **Switching** a-sensor on for $a \in \Sigma$ **costs 1** per time unit
- On input w , **maximum average cost** is
 - ▶ 1 for O_1
 - ▶ 2 for O_2
- O_1 observes **less events** than O_2 in the long run
- O_1 is **less expensive** than O_2

Need to define a **cost** measure for dynamic observers

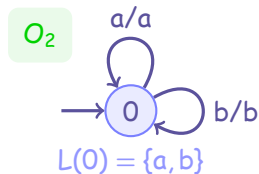
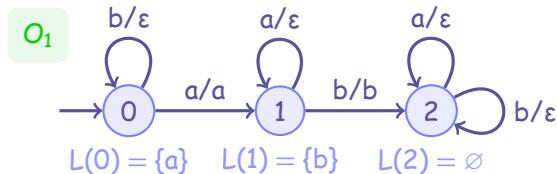
How to Compare Dynamic Observers ?



- **Switching** a-sensor on for $a \in \Sigma$ **costs 1** per time unit
- On input w , **maximum average cost** is
 - ▶ 1 for O_1
 - ▶ 2 for O_2
- O_1 observes **less events** than O_2 in the long run
- O_1 is **less expensive** than O_2

Need to define a **cost** measure for dynamic observers

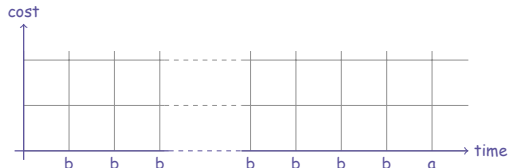
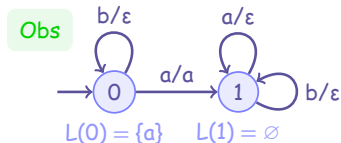
How to Compare Dynamic Observers ?



- **Switching** a-sensor on for $a \in \Sigma$ **costs 1** per time unit
- On input w , **maximum average cost** is
 - ▶ 1 for O_1
 - ▶ 2 for O_2
- O_1 observes **less events** than O_2 in the long run
- O_1 is **less expensive** than O_2

Need to define a **cost** measure for dynamic observers

Cost of an Observer



- **Cost₁**: time basis determined by the observer

- After each observable event, sum up the cost of the state of Obs
- $Obs(b^n) = \epsilon$, $Obs(b^n.a) = a$: two values

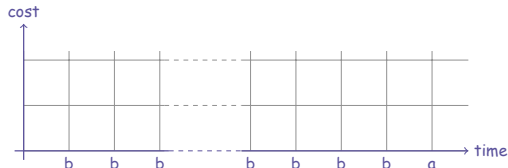
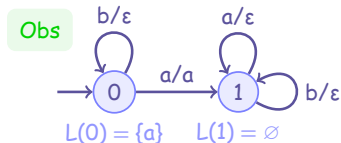
$$Cost_1(b^n.a) = \frac{1+0}{2}$$

- **Cost₂**: time basis determined by the plant being observed

- After each event's occurrence, sum up the cost of the state of Obs
- after b^n , state of Obs is 0, and after $b^n.a$, it is 1

$$Cost_2(b^n.a) = \frac{n+1+0}{n+2}$$

Cost of an Observer



- **Cost₁**: time basis determined by the observer

- 1 After each **observable event**, sum up the cost of the state of Obs
- 2 $\text{Obs}(b^n) = \varepsilon$, $\text{Obs}(b^n.a) = a$: two values

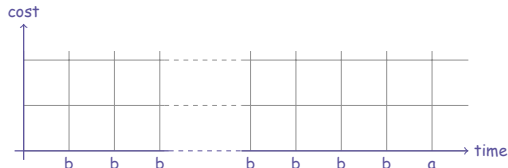
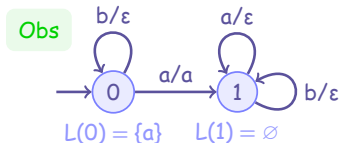
$$\text{Cost}_1(b^n.a) = \frac{1+0}{2}$$

- **Cost₂**: time basis determined by the plant being observed

- 1 After each event's occurrence, sum up the cost of the state of Obs
- 2 after b^n , state of Obs is 0, and after $b^n.a$, it is 1

$$\text{Cost}_2(b^n.a) = \frac{n+1+0}{n+2}$$

Cost of an Observer



- **Cost₁**: time basis determined by the observer

- 1 After each **observable event**, sum up the cost of the state of Obs
- 2 $Obs(b^n) = \varepsilon$, $Obs(b^n.a) = a$: two values

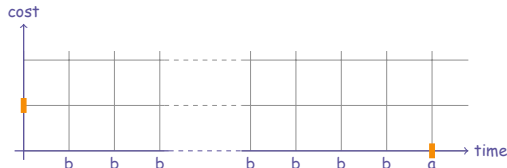
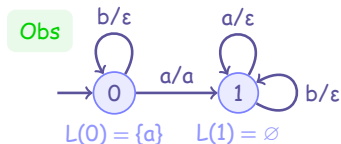
$$Cost_1(b^n.a) = \frac{1+0}{2}$$

- **Cost₂**: time basis determined by the plant being observed

- 1 After each event's occurrence, sum up the cost of the state of Obs
- 2 after b^n , state of Obs is 0, and after $b^n.a$, it is 1

$$Cost_2(b^n.a) = \frac{n+1+0}{n+2}$$

Cost of an Observer



- **Cost₁**: time basis determined by the observer

- 1 After each **observable event**, sum up the cost of the state of Obs
- 2 $\text{Obs}(b^n) = \varepsilon$, $\text{Obs}(b^n.a) = a$: two values

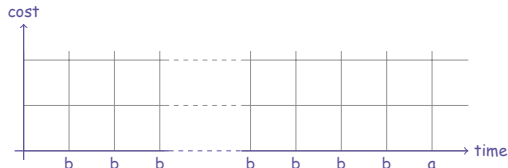
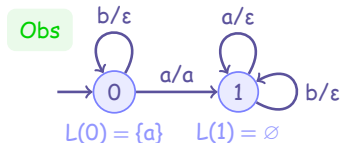
$$\text{Cost}_1(b^n.a) = \frac{1+0}{2}$$

- **Cost₂**: time basis determined by the plant being observed

- 1 After each event's occurrence, sum up the cost of the state of Obs
- 2 after b^n , state of Obs is 0, and after $b^n.a$, it is 1

$$\text{Cost}_2(b^n.a) = \frac{n+1+0}{n+2}$$

Cost of an Observer



- **Cost₁**: **time basis** determined by the observer

- 1 After each **observable event**, sum up the cost of the state of Obs
- 2 $\text{Obs}(b^n) = \varepsilon$, $\text{Obs}(b^n.a) = a$: two values

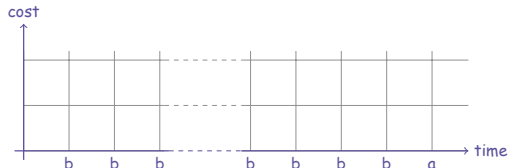
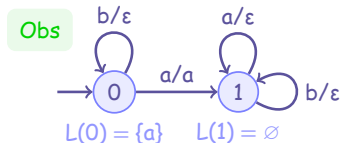
$$\text{Cost}_1(b^n.a) = \frac{1+0}{2}$$

- **Cost₂**: **time basis** determined by the plant being observed

- 1 After each **event's occurrence**, sum up the cost of the state of Obs
- 2 after b^n , state of Obs is 0, and after $b^n.a$, it is 1

$$\text{Cost}_2(b^n.a) = \frac{n+1+0}{n+2}$$

Cost of an Observer



- **Cost₁**: **time basis** determined by the observer

- 1 After each **observable event**, sum up the cost of the state of Obs
- 2 $\text{Obs}(b^n) = \varepsilon$, $\text{Obs}(b^n.a) = a$: two values

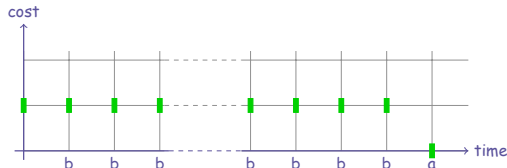
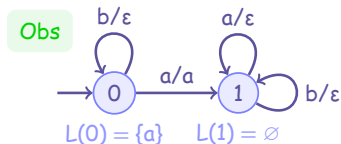
$$\text{Cost}_1(b^n.a) = \frac{1+0}{2}$$

- **Cost₂**: **time basis** determined by the plant being observed

- 1 After each **event's occurrence**, sum up the cost of the state of Obs
- 2 after b^n , state of Obs is 0, and after $b^n.a$, it is 1

$$\text{Cost}_2(b^n.a) = \frac{n+1+0}{n+2}$$

Cost of an Observer



- **Cost₁**: time basis determined by the observer

- 1 After each **observable event**, sum up the cost of the state of Obs
- 2 $\text{Obs}(b^n) = \varepsilon$, $\text{Obs}(b^n.a) = a$: two values

$$\text{Cost}_1(b^n.a) = \frac{1+0}{2}$$

- **Cost₂**: time basis determined by the plant being observed

- 1 After each **event's occurrence**, sum up the cost of the state of Obs
- 2 after b^n , state of Obs is 0, and after $b^n.a$, it is 1

$$\text{Cost}_2(b^n.a) = \frac{n+1+0}{n+2}$$

Computing the Cost of a Given Observer

A run of the plant A is $\rho = q_0 \xrightarrow{a_1} q_1 \cdots q_{n-1} \xrightarrow{a_n} q_n$

Obs is an observer and $w_i = \pi_{/\Sigma}(\text{trace}(q_0 \xrightarrow{a_1} \cdots \xrightarrow{a_i} q_i))$

Cost of a run ρ

$$\text{Cost}_2(\rho, Obs) = \frac{\text{Cost}_2(\rho)}{|\rho| + 1} = \frac{1}{n+1} \cdot \sum_{i=0}^n |L(\delta(s_0, w_i))|$$

Maximal cost of runs of length n

$$\text{Cost}_2(n, A, Obs) = \max\{ \text{Cost}_2(\rho, Obs) \text{ for } \rho \in \text{Runs}^n(A) \}$$

Cost of the pair (Obs, A)

$$\text{Cost}_2(A, Obs) = \limsup_{n \rightarrow \infty} \text{Cost}_2(n, A, Obs)$$

Theorem ([TASE'07])

$\text{Cost}_2(A, Obs)$ can be computed in PTIME.

Computing the Cost of a Given Observer

A run of the plant A is $\rho = q_0 \xrightarrow{a_1} q_1 \cdots q_{n-1} \xrightarrow{a_n} q_n$

Obs is an observer and $w_i = \pi_{/\Sigma}(\text{trace}(q_0 \xrightarrow{a_1} \cdots \xrightarrow{a_i} q_i))$

Cost of a run ρ

$$\mathbf{Cost}_2(\rho, Obs) = \frac{\mathbf{Cost}_2(\rho)}{|\rho| + 1} = \frac{1}{n+1} \cdot \sum_{i=0}^n |L(\delta(s_0, w_i))|$$

Maximal cost of runs of length n

$$\mathbf{Cost}_2(n, A, Obs) = \max\{ \mathbf{Cost}_2(\rho, Obs) \text{ for } \rho \in \mathbf{Runs}^n(A) \}$$

Cost of the pair (Obs, A)

$$\mathbf{Cost}_2(A, Obs) = \limsup_{n \rightarrow \infty} \mathbf{Cost}_2(n, A, Obs)$$

Theorem ([TASE'07])

$\mathbf{Cost}_2(A, Obs)$ can be computed in **PTIME**.

Computing the Cost of a Given Observer

A run of the plant A is $\rho = q_0 \xrightarrow{a_1} q_1 \cdots q_{n-1} \xrightarrow{a_n} q_n$

Obs is an observer and $w_i = \pi_{/\Sigma}(\text{trace}(q_0 \xrightarrow{a_1} \cdots \xrightarrow{a_i} q_i))$

Cost of a run ρ

$$\text{Cost}_2(\rho, Obs) = \frac{\text{Cost}_2(\rho)}{|\rho| + 1} = \frac{1}{n+1} \cdot \sum_{i=0}^n |L(\delta(s_0, w_i))|$$

Maximal cost of runs of length n

$$\text{Cost}_2(n, A, Obs) = \max\{\text{Cost}_2(\rho, Obs) \text{ for } \rho \in \text{Runs}^n(A)\}$$

Cost of the pair (Obs, A)

$$\text{Cost}_2(A, Obs) = \limsup_{n \rightarrow \infty} \text{Cost}_2(n, A, Obs)$$

Theorem ([TASE'07])

$\text{Cost}_2(A, Obs)$ can be computed in PTIME.

Computing the Cost of a Given Observer

A run of the plant A is $\rho = q_0 \xrightarrow{a_1} q_1 \cdots q_{n-1} \xrightarrow{a_n} q_n$

Obs is an observer and $w_i = \pi_{/\Sigma}(\text{trace}(q_0 \xrightarrow{a_1} \cdots \xrightarrow{a_i} q_i))$

Cost of a run ρ

$$\mathbf{Cost}_2(\rho, Obs) = \frac{\mathbf{Cost}_2(\rho)}{|\rho| + 1} = \frac{1}{n+1} \cdot \sum_{i=0}^n |L(\delta(s_0, w_i))|$$

Maximal cost of runs of length n

$$\mathbf{Cost}_2(n, A, Obs) = \max\{ \mathbf{Cost}_2(\rho, Obs) \text{ for } \rho \in \mathbf{Runs}^n(A) \}$$

Cost of the pair (Obs, A)

$$\mathbf{Cost}_2(A, Obs) = \limsup_{n \rightarrow \infty} \mathbf{Cost}_2(n, A, Obs)$$

Theorem ([TASE'07])

$\mathbf{Cost}_2(A, Obs)$ can be computed in PTIME.

Computing the Cost of a Given Observer

A run of the plant A is $\rho = q_0 \xrightarrow{a_1} q_1 \cdots q_{n-1} \xrightarrow{a_n} q_n$

Obs is an observer and $w_i = \pi_{/\Sigma}(\text{trace}(q_0 \xrightarrow{a_1} \cdots \xrightarrow{a_i} q_i))$

Cost of a run ρ

$$\mathbf{Cost}_2(\rho, Obs) = \frac{\mathbf{Cost}_2(\rho)}{|\rho| + 1} = \frac{1}{n+1} \cdot \sum_{i=0}^n |L(\delta(s_0, w_i))|$$

Maximal cost of runs of length n

$$\mathbf{Cost}_2(n, A, Obs) = \max\{ \mathbf{Cost}_2(\rho, Obs) \text{ for } \rho \in \mathbf{Runs}^n(A) \}$$

Cost of the pair (Obs, A)

$$\mathbf{Cost}_2(A, Obs) = \limsup_{n \rightarrow \infty} \mathbf{Cost}_2(n, A, Obs)$$

Theorem ([TASE'07])

$\mathbf{Cost}_2(A, Obs)$ can be computed in **PTIME**.

Computing the Cost of a Given Observer

A run of the plant A is $\rho = q_0 \xrightarrow{a_1} q_1 \cdots q_{n-1} \xrightarrow{a_n} q_n$

Obs is an observer and $w_i = \pi_{/\Sigma}(\text{trace}(q_0 \xrightarrow{a_1} \cdots \xrightarrow{a_i} q_i))$

Cost of a run ρ

$$\mathbf{Cost}_2(\rho, Obs) = \frac{\mathbf{Cost}_2(\rho)}{|\rho| + 1} = \frac{1}{n + 1} \cdot \sum_{i=0}^n |L(\delta(s_0, w_i))|$$

Maximal cost of runs of length n

$$\mathbf{Cost}_2(n, A, Obs) = \max\{\mathbf{Cost}_2(\rho, Obs) \text{ for } \rho \in \mathbf{Runs}^n(A)\}$$

Cost of the pair (Obs, A)

$$\mathbf{Cost}_2(A, Obs) = \limsup_{n \rightarrow \infty} \mathbf{Cost}_2(n, A, Obs)$$

Theorem ([TASE'07])

$\mathbf{Cost}_2(A, Obs)$ can be computed in PTIME.

Use Karp's Maximum Mean-Weight Cycle Algorithm

Computing the Cost of a Given Observer

A run of the plant A is $\rho = q_0 \xrightarrow{a_1} q_1 \cdots q_{n-1} \xrightarrow{a_n} q_n$

Obs is an observer and $w_i = \pi_{/\Sigma}(\text{trace}(q_0 \xrightarrow{a_1} \cdots \xrightarrow{a_i} q_i))$

Cost of a run ρ

$$\mathbf{Cost}_2(\rho, Obs) = \frac{\mathbf{Cost}_2(\rho)}{|\rho| + 1} = \frac{1}{n+1} \cdot \sum_{i=0}^n |L(\delta(s_0, w_i))|$$

Maximal cost of runs of length n

$$\mathbf{Cost}_2(n, A, Obs) = \max\{\mathbf{Cost}_2(\rho, Obs) \text{ for } \rho \in \mathbf{Runs}^n(A)\}$$

Cost of the pair (Obs, A)

$$\mathbf{Cost}_2(A, Obs) = \limsup_{n \rightarrow \infty} \mathbf{Cost}_2(n, A, Obs)$$

Theorem ([TASE'07])

$\mathbf{Cost}_2(A, Obs)$ can be computed in **PTIME**.

How to synthesize a/the **best or optimal** observer ?

Bounded Cost Observer

Problem 2: Bounded Cost Observer

Input: A , $k \in \mathbb{N}$ and $c \in \mathbb{N}$.

Problem: Assume A is (Σ, k) -diagnosable,

(A). Is there an observer Obs s.t. A is (Obs, k) -diagnosable and $\mathbf{Cost}_2(Obs) \leq c$?

(B). If "yes", compute a **witness observer** Obs with $\mathbf{Cost}_2(Obs) \leq c$.

Steps to Solve Problem 2:

- Step 1: Compute the **most permissive** observer O ; see **Problem 3**
- Step 2: Compute an **optimal cost** observer.

Theorem ([ACSD'07])

There is a **finite state most permissive** observer for A .

Theorem ([TASE'07])

An **optimal observer** can be computed in **2EXPTIME** (in size of A and Σ).

Bounded Cost Observer

Problem 2: Bounded Cost Observer

Input: A , $k \in \mathbb{N}$ and $c \in \mathbb{N}$.

Problem: Assume A is (Σ, k) -diagnosable,

(A). Is there an observer Obs s.t. A is (Obs, k) -diagnosable and $\mathbf{Cost}_2(Obs) \leq c$?

(B). If "yes", compute a **witness observer** Obs with $\mathbf{Cost}_2(Obs) \leq c$.

Steps to Solve Problem 2:

- Step 1: Compute the **most permissive** observer O ; see **Problem 3**
- Step 2: Compute an **optimal cost** observer.

Theorem ([ACSD'07])

There is a **finite state most permissive** observer for A .

Theorem ([TASE'07])

An **optimal observer** can be computed in **2EXPTIME** (in size of A and Σ).

Bounded Cost Observer

Problem 2: Bounded Cost Observer

Input: A , $k \in \mathbb{N}$ and $c \in \mathbb{N}$.

Problem: Assume A is (Σ, k) -diagnosable,

(A). Is there an observer Obs s.t. A is (Obs, k) -diagnosable and $\mathbf{Cost}_2(Obs) \leq c$?

(B). If "yes", compute a **witness observer** Obs with $\mathbf{Cost}_2(Obs) \leq c$.

Steps to Solve Problem 2:

- Step 1: Compute the **most permissive** observer O ; see **Problem 3**
- Step 2: Compute an **optimal cost** observer.

Theorem ([ACSD'07])

There is a **finite state most permissive** observer for A .

Theorem ([TASE'07])

An **optimal observer** can be computed in **2EXPTIME** (in size of A and Σ).

Bounded Cost Observer

Problem 2: Bounded Cost Observer

Input: A , $k \in \mathbb{N}$ and $c \in \mathbb{N}$.

Problem: Assume A is (Σ, k) -diagnosable,

(A). Is there an observer Obs s.t. A is (Obs, k) -diagnosable and $\mathbf{Cost}_2(Obs) \leq c$?

(B). If "yes", compute a **witness observer** Obs with $\mathbf{Cost}_2(Obs) \leq c$.

Steps to Solve Problem 2:

- Step 1: Compute the **most permissive** observer O ; see **Problem 3**
- **Step 2**: Compute an **optimal cost** observer.

Theorem ([ACSD'07])

There is a **finite state most permissive** observer for A .

Theorem ([TASE'07])

An **optimal observer** can be computed in **2EXPTIME** (in size of A and Σ).

Bounded Cost Observer

Problem 2: Bounded Cost Observer

Input: A , $k \in \mathbb{N}$ and $c \in \mathbb{N}$.

Problem: Assume A is (Σ, k) -diagnosable,

(A). Is there an observer Obs s.t. A is (Obs, k) -diagnosable and $\mathbf{Cost}_2(Obs) \leq c$?

(B). If "yes", compute a **witness observer** Obs with $\mathbf{Cost}_2(Obs) \leq c$.

Steps to Solve Problem 2:

- Step 1: Compute the **most permissive** observer O ; see **Problem 3**
- Step 2: Compute an **optimal cost** observer.

Theorem ([ACSD'07])

There is a **finite state most permissive observer** for A .

Theorem ([TASE'07])

An **optimal observer** can be computed in **2EXPTIME** (in size of A and Σ).

Bounded Cost Observer

Problem 2: Bounded Cost Observer

Input: A , $k \in \mathbb{N}$ and $c \in \mathbb{N}$.

Problem: Assume A is (Σ, k) -diagnosable,

(A). Is there an observer Obs s.t. A is (Obs, k) -diagnosable and $\mathbf{Cost}_2(Obs) \leq c$?

(B). If "yes", compute a **witness observer** Obs with $\mathbf{Cost}_2(Obs) \leq c$.

Steps to Solve Problem 2:

- Step 1: Compute the **most permissive** observer O ; see **Problem 3**
- Step 2: Compute an **optimal cost** observer.

Theorem ([ACSD'07])

There is a **finite state most permissive observer** for A .

Theorem ([TASE'07])

An **optimal observer** can be computed in **2EXPTIME** (in size of A and Σ).

Most Permissive Observer

Problem 3+: Dynamic Diagnosability Problem

Input: A .

Problem: Compute the set of **all observers** s.t. A is Obs-diagnosable.

Future work ...

Problem 3: Dynamic k -Diagnosability Problem

Input: $A, k \in \mathbb{N}$.

Problem: Compute the set of **all observers** s.t. A is (Obs, k)-diagnosable.

Most Permissive Observer

Problem 3+: Dynamic Diagnosability Problem

Input: A .

Problem: Compute the set of **all observers** s.t. A is Obs-diagnosable.

Future work ...

Problem 3: Dynamic k -Diagnosability Problem

Input: $A, k \in \mathbb{N}$.

Problem: Compute the set of **all observers** s.t. A is (Obs, k)-diagnosable.

Most Permissive Observer

Problem 3+: Dynamic Diagnosability Problem

Input: A .

Problem: Compute the set of **all observers** s.t. A is Obs-diagnosable.

Future work ...

Problem 3: Dynamic k -Diagnosability Problem

Input: $A, k \in \mathbb{N}$.

Problem: Compute the set of **all observers** s.t. A is (Obs, k)-diagnosable.

Problem 3 as a Game Problem

- Reduce Problem 3 to a **safety 2-player game**
- **Player 1** chooses what to observe
- **Player 2** tries to produce a **bad** sequence of events
i.e. traces that are in $\text{NonFaulty}(A) \cap \text{Faulty}_k(A)$
behaves like a synchronized product of two automata

Problem 3 as a Game Problem

- Reduce Problem 3 to a **safety 2-player game**
- **Player 1** chooses what to observe
- **Player 2** tries to produce a **bad** sequence of events
i.e. traces that are in $\text{NonFaulty}(A) \cap \text{Faulty}_{\Delta}(A)$
behaves like a synchronized product of two automata

Problem 3 as a Game Problem

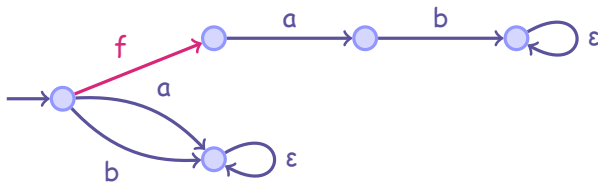
- Reduce Problem 3 to a **safety 2-player game**
- **Player 1** chooses what to observe
- **Player 2** tries to produce a **bad** sequence of events
i.e. traces that are in $\text{NonFaulty}(A) \cap \text{Faulty}_\Delta(A)$
behaves like a synchronized product of two automata

Problem 3 as a Game Problem

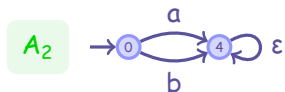
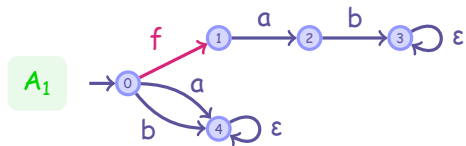
- Reduce Problem 3 to a **safety 2-player game**
- **Player 1** chooses what to observe
- **Player 2** tries to produce a **bad** sequence of events
i.e. traces that are in $\text{NonFaulty}(A) \cap \text{Faulty}_{\geq k}(A)$
behaves like a **synchronized product** of two automata

Problem 3 as a Game Problem

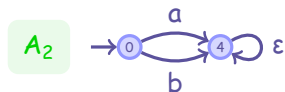
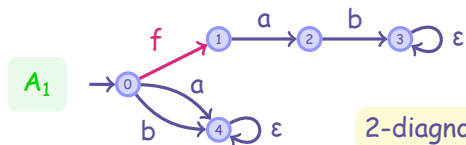
- Reduce Problem 3 to a **safety 2-player game**
- **Player 1** chooses what to observe
- **Player 2** tries to produce a **bad** sequence of events
i.e. traces that are in $\text{NonFaulty}(A) \cap \text{Faulty}_{\geq k}(A)$
behaves like a **synchronized product** of two automata



Problem 3 as a Game Problem

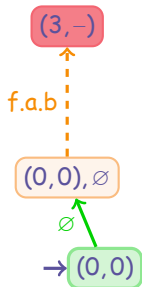
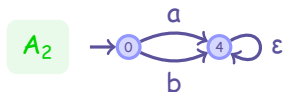
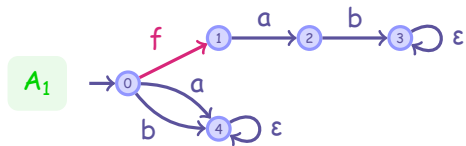


Problem 3 as a Game Problem

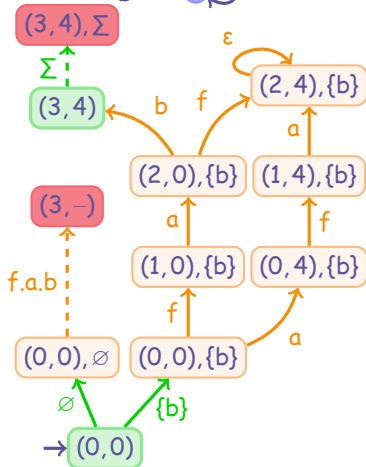
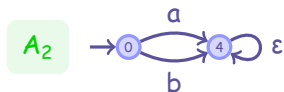
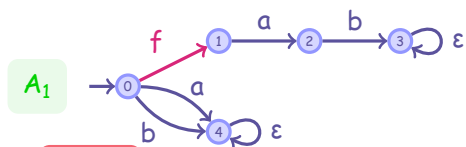


2-diagnosable: avoid $\text{Bad} = (3, -)$ in $A_1 \times A_2$

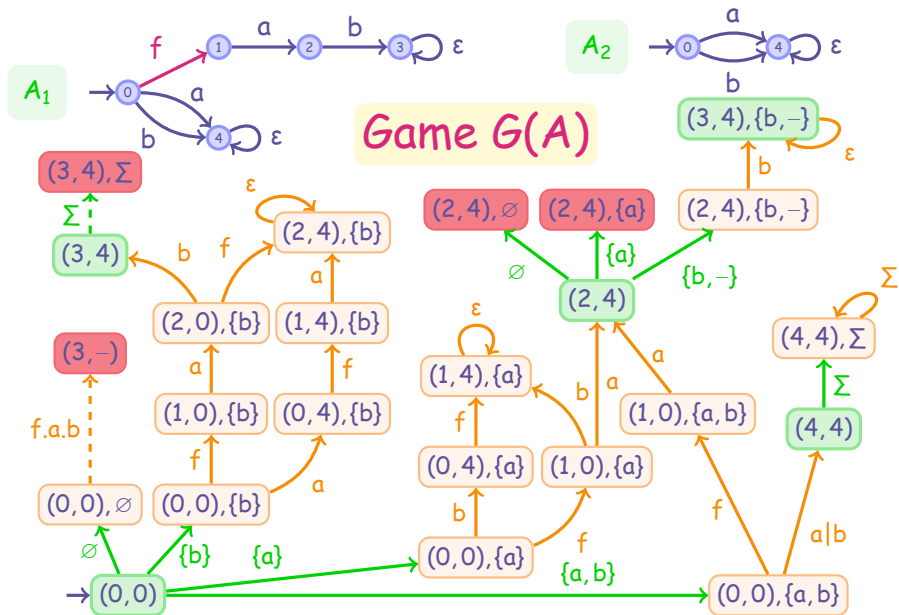
Problem 3 as a Game Problem



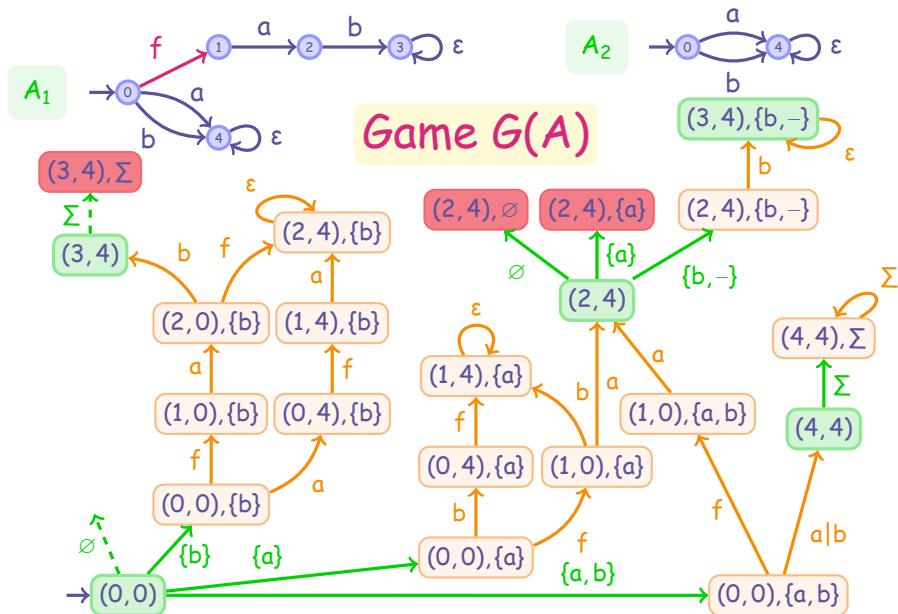
Problem 3 as a Game Problem



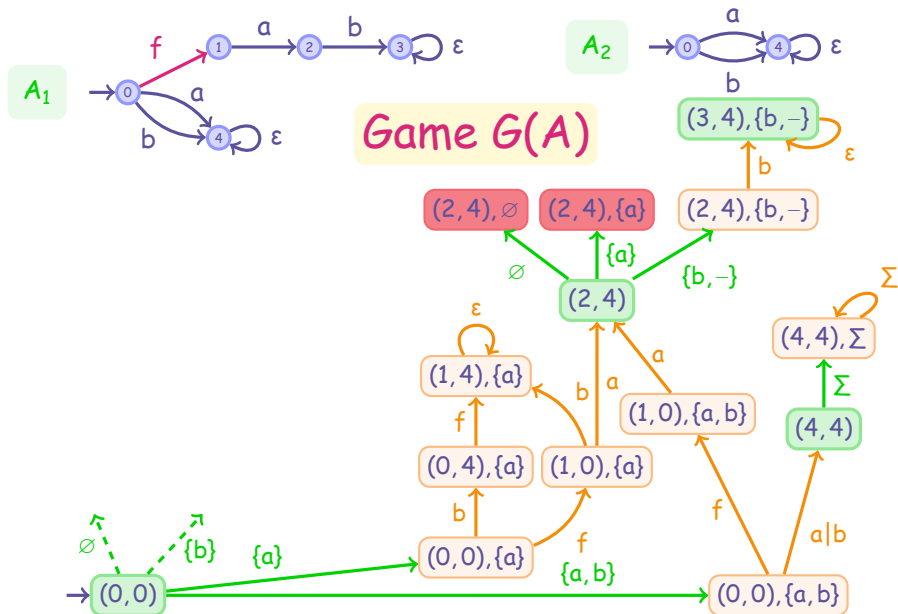
Problem 3 as a Game Problem



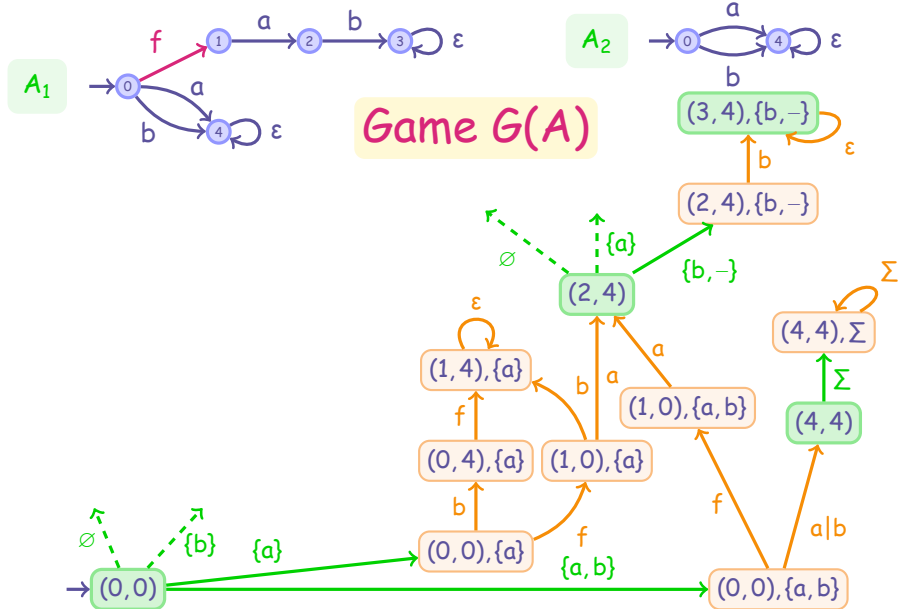
Problem 3 as a Game Problem



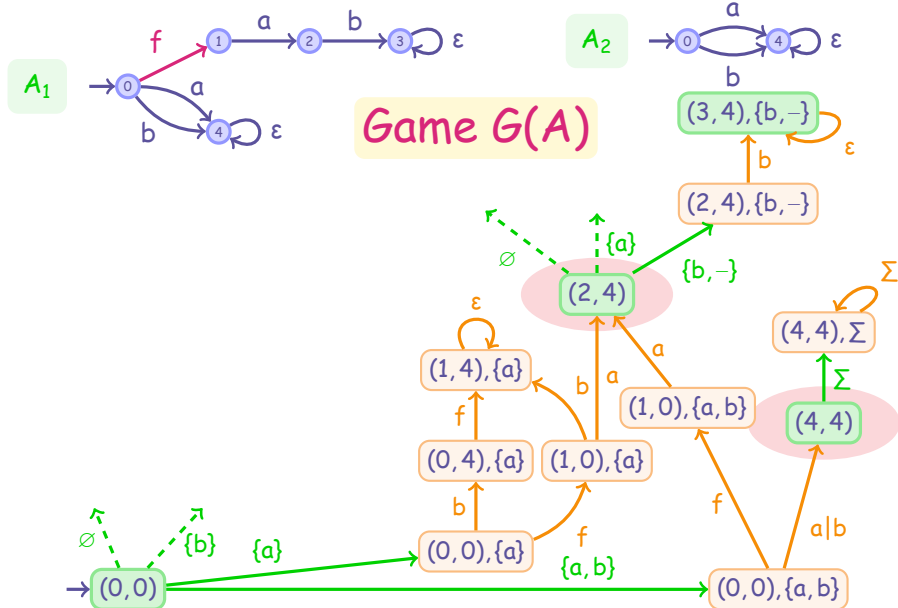
Problem 3 as a Game Problem



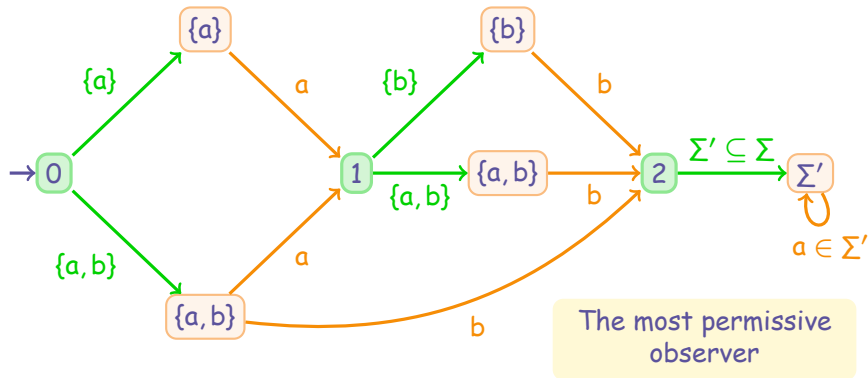
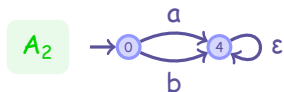
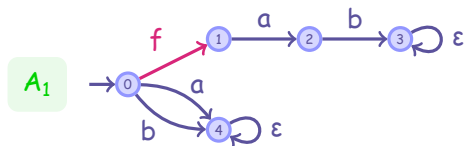
Problem 3 as a Game Problem



Problem 3 as a Game Problem



Problem 3 as a Game Problem



Results about the Most Permissive Observer

Let $G(A)$ be the game associated with A and Σ

$G(A)$ is a game with **partial observation** of the actions

Theorem ([FI08])

- O is an observer s.t. A is (O, k) -diagnosable \implies there is a **corresponding** winning strategy $f(O)$ in $G(A)$.
- There is a winning strategy f in $G(A)$ \implies there is a **corresponding** observer $O(f)$ s.t. A is $(O(f), k)$ -diagnosable.

Theorem (Memoryless Strategies are Sufficient)

There is a **memoryless most permissive strategy** for any (safety) finite game G with **complete information**.

Theorem (Existence of a Most Permissive Observer)

There is a **finite-state (2EXP) most permissive observer** for A .

Results about the Most Permissive Observer

Let $G(A)$ be the game associated with A and Σ

$G(A)$ is a game with **partial observation** of the actions

Theorem ([FI08])

- O is an observer s.t. A is (O, k) -diagnosable \implies there is a **corresponding** winning strategy $f(O)$ in $G(A)$.
- There is a winning strategy f in $G(A)$ \implies there is a **corresponding** observer $O(f)$ s.t. A is $(O(f), k)$ -diagnosable.

Theorem (Memoryless Strategies are Sufficient)

There is a **memoryless most permissive strategy** for any (safety) finite game G with **complete information**.

Theorem (Existence of a Most Permissive Observer)

There is a **finite-state (2EXP) most permissive observer** for A .

Results about the Most Permissive Observer

Let $G(A)$ be the game associated with A and Σ

$G(A)$ is a game with **partial observation** of the actions

Theorem ([FI'08])

- O is an observer s.t. A is (O, k) -diagnosable \implies there is a **corresponding** winning strategy $f(O)$ in $G(A)$.
- There is a winning strategy f in $G(A)$ \implies there is a **corresponding** observer $O(f)$ s.t. A is $(O(f), k)$ -diagnosable.

Theorem (Memoryless Strategies are Sufficient)

There is a **memoryless most permissive strategy** for any (safety) finite game G with **complete information**.

Theorem (Existence of a Most Permissive Observer)

There is a **finite-state (2EXP) most permissive observer** for A .

Results about the Most Permissive Observer

Let $G(A)$ be the game associated with A and Σ

$G(A)$ is a game with **partial observation** of the actions

Theorem ([FI'08])

- O is an observer s.t. A is (O, k) -diagnosable \implies there is a **corresponding** winning strategy $f(O)$ in $G(A)$.
- There is a winning strategy f in $G(A)$ \implies there is a **corresponding** observer $O(f)$ s.t. A is $(O(f), k)$ -diagnosable.

Theorem (Memoryless Strategies are Sufficient)

There is a **memoryless most permissive strategy** for any (safety) finite game G with **complete information**.

Theorem (Existence of a Most Permissive Observer)

There is a **finite-state (2EXP) most permissive observer** for A .

Results about the Most Permissive Observer

Let $G(A)$ be the game associated with A and Σ

$G(A)$ is a game with **partial observation** of the actions

Theorem ([FI'08])

- O is an observer s.t. A is (O, k) -diagnosable \implies there is a **corresponding** winning strategy $f(O)$ in $G(A)$.
- There is a winning strategy f in $G(A)$ \implies there is a **corresponding** observer $O(f)$ s.t. A is $(O(f), k)$ -diagnosable.

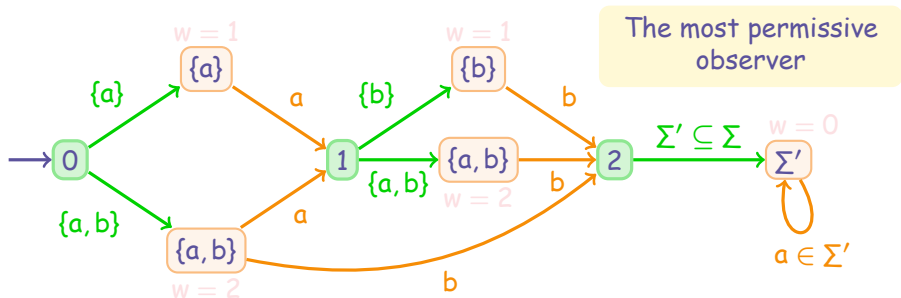
Theorem (Memoryless Strategies are Sufficient)

There is a **memoryless most permissive strategy** for any (safety) finite game G with **complete information**.

Theorem (Existence of a Most Permissive Observer)

There is a **finite-state (2EXP) most permissive observer** for A .

Optimal Dynamic Observer



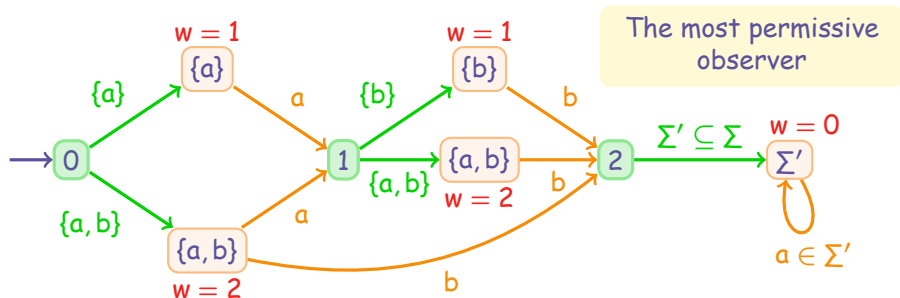
- Player 1 chooses what to observe: X / Player 2 generates $p, l, l \in X$
- Player 1 and 2 produce weighted runs
- Given a strategy for Player 1 and the moves of Player 2

$w(p)$ is the sum of the weights $w_1 w_2 \cdots w_n$ of the run p

$$\text{Cost}_2(p, \text{Obs}) = w(p) / (|\rho| + 1)$$

- Goal for Player 1: minimize $\limsup_{n \rightarrow \infty} \{w(p) / (|\rho| + 1) \mid p \in \text{Runs}^n(A)\}$

Optimal Dynamic Observer



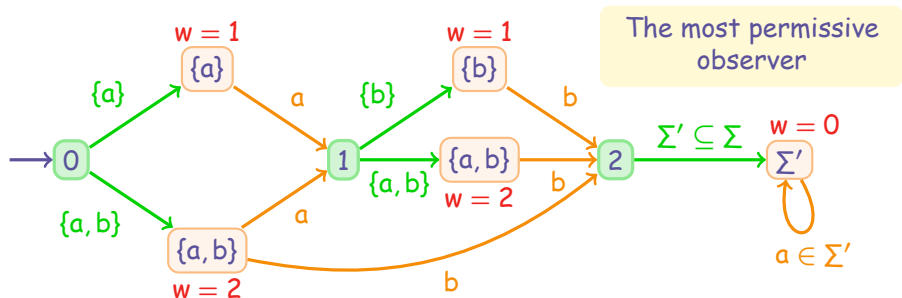
- Player 1 chooses what to observe: X / Player 2 generates $p.l, l \in X$
- Player 1 and 2 produce **weighted** runs
- Given a **strategy** for Player 1 and the moves of Player 2

$w(p)$ is the **sum** of the weights $w_1 w_2 \cdots w_n$ of the run p

$$\text{Cost}_2(p, \text{Obs}) = w(p) / (|\rho| + 1)$$

- Goal for Player 1: minimize $\limsup_{n \rightarrow \infty} \{w(p) / (|\rho| + 1) \mid p \in \text{Runs}^n(A)\}$

Optimal Dynamic Observer



- Player 1 chooses what to observe: X / Player 2 generates $p.l, l \in X$
- Player 1 and 2 produce **weighted** runs
- Given a **strategy** for Player 1 and the moves of Player 2

$w(p)$ is the **sum** of the weights $w_1 w_2 \cdots w_n$ of the run p

$$\text{Cost}_2(p, \text{Obs}) = w(p) / (|\rho| + 1)$$

- Goal for Player 1: minimize $\limsup_{n \rightarrow \infty} \{w(p) / (|\rho| + 1) \mid p \in \text{Runs}^n(A)\}$

Mean Payoff Games

Weighted two-player games

- Each state s has a **weight** $w(s)$
can be done with weight on edges
- **Turn-based** game
- Goal of the Players:
 - ▶ Player 1: **minimize** $l_0 = \limsup w(\rho)/(|\rho| + 1)$
 - ▶ Player 2: **maximize** $l_1 = \liminf w(\rho)/(|\rho| + 1)$

Results for weighted two-player games

[Zwick & Paterson, TCS 1996]

- There is a value $v \in \mathbb{Q}$ s.t. each player has a **memoryless** strategy to ensure $l_0 \leq v$ and $l_1 \geq v$
- v can be **effectively** computed
- **Memoryless** strategies for both players can be **effectively** computed

Winning Strategy for Player 1 = Optimal Observer

Mean Payoff Games

Weighted two-player games

- Each state s has a **weight** $w(s)$
can be done with weight on edges
- **Turn-based** game
- Goal of the Players:
 - ▶ Player 1: **minimize** $l_0 = \limsup w(\rho)/(|\rho| + 1)$
 - ▶ Player 2: **maximize** $l_1 = \liminf w(\rho)/(|\rho| + 1)$

Results for weighted two-player games

[Zwick & Paterson, TCS 1996]

- There is a value $v \in \mathbb{Q}$ s.t. each player has a **memoryless** strategy to ensure $l_0 \leq v$ and $l_1 \geq v$
- v can be **effectively** computed
- **Memoryless** strategies for both players can be **effectively** computed

Winning Strategy for Player 1 = Optimal Observer

Mean Payoff Games

Weighted two-player games

- Each state s has a **weight** $w(s)$
can be done with weight on edges
- **Turn-based** game
- Goal of the Players:
 - ▶ Player 1: **minimize** $l_0 = \limsup w(\rho)/(|\rho| + 1)$
 - ▶ Player 2: **maximize** $l_1 = \liminf w(\rho)/(|\rho| + 1)$

Results for weighted two-player games

[Zwick & Paterson, TCS 1996]

- There is a value $v \in \mathbb{Q}$ s.t. each player has a **memoryless** strategy to ensure $l_0 \leq v$ and $l_1 \geq v$
- v can be **effectively** computed
- **Memoryless** strategies for both players can be **effectively** computed

Winning Strategy for Player 1 = Optimal Observer

Results for Problem 2

Problem 2: Bounded Cost Observer

Input: A , $k \in \mathbb{N}$ and $c \in \mathbb{N}$.

Problem: Assume A is (Σ, k) -diagnosable,

(A). Is there an observer Obs s.t. A is (Obs, k) -diagnosable and $\mathbf{Cost}_2(\text{Obs}) \leq c$?

(B). If "yes", compute a **witness observer** Obs with $\mathbf{Cost}_2(\text{Obs}) \leq c$.

Solution to Problem 2

- 1. Compute the most permissive observer O
- 2. Build a weighted graph game: $O \times A$
- 3. Use Zwick & Paterson's algorithm to compute the value of the game
- 4. Compare c to the value of the game

Theorem ([TASE'07])

Problem 2 can be solved in 2EXPTIME .

An optimal observer can be computed in 2EXPTIME .

Results for Problem 2

Problem 2: Bounded Cost Observer

Input: A , $k \in \mathbb{N}$ and $c \in \mathbb{N}$.

Problem: Assume A is (Σ, k) -diagnosable,

(A). Is there an observer Obs s.t. A is (Obs, k) -diagnosable and $\mathbf{Cost}_2(Obs) \leq c$?

(B). If "yes", compute a **witness observer** Obs with $\mathbf{Cost}_2(Obs) \leq c$.

Solution to Problem 2

- 1 Compute the **most permissive observer** O
- 2 Build a **weighted graph game**: $O \times A$
- 3 Use **Zwick & Paterson's** algorithm to compute the value of the game
- 4 Compare c to the value of the game

Theorem ([TASE'07])

Problem 2 can be solved in 2EXPTIME.

An optimal observer can be computed in 2EXPTIME.

Results for Problem 2

Problem 2: Bounded Cost Observer

Input: A , $k \in \mathbb{N}$ and $c \in \mathbb{N}$.

Problem: Assume A is (Σ, k) -diagnosable,

(A). Is there an observer Obs s.t. A is (Obs, k) -diagnosable and $\mathbf{Cost}_2(Obs) \leq c$?

(B). If "yes", compute a **witness observer** Obs with $\mathbf{Cost}_2(Obs) \leq c$.

Solution to Problem 2

- 1 Compute the **most permissive observer** O
- 2 Build a **weighted graph game**: $O \times A$
- 3 Use **Zwick & Paterson's** algorithm to compute the value of the game
- 4 Compare c to the value of the game

Theorem ([TASE'07])

Problem 2 can be solved in **2EXPTIME**.

An **optimal observer** can be computed in **2EXPTIME**.

Results for Problem 2

Problem 2: Bounded Cost Observer

Input: A , $k \in \mathbb{N}$ and $c \in \mathbb{N}$.

Problem: Assume A is (Σ, k) -diagnosable,

(A). Is there an observer Obs s.t. A is (Obs, k) -diagnosable and $\mathbf{Cost}_2(Obs) \leq c$?

(B). If "yes", compute a **witness observer** Obs with $\mathbf{Cost}_2(Obs) \leq c$.

Solution to Problem 2

- 1 Compute the **most permissive observer** O
- 2 Build a **weighted graph game**: $O \times A$
- 3 Use **Zwick & Paterson's** algorithm to compute the value of the game
- 4 Compare c to the value of the game

Theorem ([TASE'07])

Problem 2 can be solved in **2EXPTIME**.

An **optimal observer** can be computed in **2EXPTIME**.

Conclusion & Future Work

Results:

- Long version in [FI'08], short version in [WODES'08]
- **Dynamic** observers for bounded diagnosis (k-diagnosability)
- **Computation** of the **most permissive** observer [ACSD'07]
- **Cost & computation of the cost** of dynamic observers
- Existence of a **finite optimal dynamic** observer [TASE'07]

Recent Work

- Security problem: **opacity** with dynamic observers [ATVA'09]

Future Work:

- **Exact complexity** of Problem 2
(less than 2EXPTIME, completeness ??)
- **Solution** to Generalised Problem 2 (delay k unknown)
- **Implement** the algorithms
- **Extend** to **timed automata**

Conclusion & Future Work

Results:

- Long version in [FI'08], short version in [WODES'08]
- **Dynamic** observers for bounded diagnosis (k-diagnosability)
- **Computation** of the **most permissive** observer [ACSD'07]
- **Cost & computation of the cost** of dynamic observers
- Existence of a **finite optimal dynamic** observer [TASE'07]

Recent Work

- Security problem: **opacity** with dynamic observers [ATVA'09]

Future Work:

- **Exact complexity** of Problem 2
(less than $2EXPTIME$, completeness ??)
- **Solution** to Generalised Problem 2 (delay k unknown)
- **Implement** the algorithms
- **Extend** to **timed automata**

Conclusion & Future Work

Results:

- Long version in [FI'08], short version in [WODES'08]
- **Dynamic** observers for bounded diagnosis (k-diagnosability)
- **Computation** of the **most permissive** observer [ACSD'07]
- **Cost & computation of the cost** of dynamic observers
- Existence of a **finite optimal dynamic** observer [TASE'07]

Recent Work

- Security problem: **opacity** with dynamic observers [ATVA'09]

Future Work:

- **Exact complexity** of Problem 2
(less than 2EXPTIME, completeness ??)
- **Solution** to Generalised Problem 2 (delay k unknown)
- **Implement** the algorithms
- **Extend to timed automata**

References I

- [ATVA'09] Franck Cassez, Jérémy Dubreil, and Hervé Marchand.
Dynamic Observers for the Synthesis of Opaque Systems.
In 7th Int. Symp. on Automated Technology for Verification and Analysis (ATVA'09), volume 5799 of Lecture Notes in Computer Science, pages 352-367, Macau SAR, China, October 2009.
- [FI'08] Franck Cassez and Stavros Tripakis.
Fault Diagnosis with Static and Dynamic Observers.
Fundamenta Informatica, 88(4), pages 497-540, November 2008.
- [WODES'08] Franck Cassez and Stavros Tripakis.
Fault diagnosis with dynamic diagnosers.
In Proc. of the 9th Workshop on Discrete Event Systems (WODES'08), pages 212-217. IEEE Computer Society, May 2008.
- [ACSD'07] Franck Cassez, Stavros Tripakis, and Karine Altisen.
Sensor minimization problems with static or dynamic observers for fault diagnosis.
In 7th International Conference on Application of Concurrency to System Design (ACSD'07), Bratislava, Slovak Republic, July 2007. IEEE Computer Society.
- [TASE'07] Franck Cassez, Stavros Tripakis, and Karine Altisen.
Synthesis Of Optimal-Cost Dynamic Observers for Fault Diagnosis of Discrete-Event Systems
In 1st IFIP & IEEE International Conference on Theoretical Aspects of Software Engineering (TASE'07), Shanghai, China, June 2007. IEEE Computer Society.
- [Dasdan et al., 1999] Ali Dasdan, Sandy S. Irani, and Rajesh K. Gupta.
Efficient algorithms for optimum cycle mean and optimum cost to time ratio problems.
In Annual ACM IEEE Design Automation Conference, pages 37-42, New Orleans, Louisiana, United States, 1999. ACM Press New York, NY, USA.
ISBN:1-58133-109-7.
- [Debouk et al., 2004] Rami Debouk, Stéphane Lafortune, and Demosthenis Teneketzis.
On an optimization problem in sensor selection.
Discrete Event Dynamic Systems, 4(12), November 2004.
- [Jiang et al., 2001] Shengbing Jiang, Zhongdong Huang, Vigyan Chandra, and Ratnesh Kumar.
A polynomial algorithm for testing diagnosability of discrete event systems.
IEEE Transactions on Automatic Control, 46(8), August 2001.

References II

- [Jiang et al., 2003] Shengbing Jiang, Ratnesh Kumar, and Humberto E. Garcia.
Optimal sensor selection for discrete event systems with partial observation.
IEEE Transactions on Automatic Control, 48(3):369-381, March 2003.
- [Karp, Discrete Math. 1978] Richard M. Karp.
A characterization of the minimum mean cycle in a digraph.
Discrete Mathematics, 23:309-311, 1978.
- [Sampath et al., 1995] Meera Sampath, Raja Sengupta, Stéphane Lafortune, Kasim Sinnamohideen, and Demosthenis C. Teneketzis.
Diagnosability of discrete event systems.
IEEE Transactions on Automatic Control, 40(9), September 1995.
- [Yoo et al., 2001] Tae-Sic Yoo and Stéphane Lafortune.
On the computational complexity of some problems arising in partially-observed discrete event systems.
In American Control Conference (ACC'01), 2001.
Arlington, VA.
- [Yoo et al., 2002] Yoo, T.-S., Lafortune, S.: Polynomial-Time Verification of Diagnosability of Partially-Observed Discrete-Event Systems,
IEEE Transactions on Automatic Control, 47(9), September 2002, 1491-1495.
- [Zwick & Paterson, TCS 1996] Uri Zwick and Michael S. Paterson.
The complexity of mean payoff games on graphs.
Theoretical Computer Science, 158(1-2):343-359, 1996.