Optimized Diagnosability of Distributed Discrete Event Systems Through Abstraction

Lina YE
Doctoral Dissertation
LRI, Univ. Paris-Sud, CNRS / INRIA Saclay Ile-de-France
Advisor: Prof. Philippe DAGUE

July 6, 2011
Outline

1. Introduction

2. Diag_Abstraction
   - Preliminaries
   - Original diagnosability information
   - Global consistency checking

3. Diag_Pattern
   - Preliminaries
   - Pattern recognition
   - Pattern diagnosability checking
   - Preliminary Implementation

4. Diag_Cooperation
   - Preliminaries
   - Cooperative diagnosability

5. Conclusion
Diagnosis and diagnosability

Model-based diagnosis

- Goal: detect possible faults to explain observations of the system
- System knowledge, e.g. system model including normal behavior and fault behavior
- Reasoning: deduce the exact fault that is consistent with system knowledge and observations

Diagnosability

- Perform on system model at the design stage
- Determine the effectiveness of diagnosis algorithm
- Help in designing diagnosis algorithm
Diagnosis and diagnosability

Model-based diagnosis

- Goal: detect possible faults to explain observations of the system
- System knowledge, e.g. system model including normal behavior and fault behavior
- Reasoning: deduce the exact fault that is consistent with system knowledge and observations

Diagnosability

- Perform on system model at the design stage
- Determine the effectiveness of diagnosis algorithm
- Help in designing diagnosis algorithm
Why distributed?

- Systems: a set of components, distributed nature
- Components: produced by different providers
- Knowledge of component: sometimes black box for other components
- Centralized approach: monolithic model, state space explosion, issue of privacy
- Distributed approach: local models, exploit modular structure of the system
Related works

Diagnosability of DES: State of the art

- Sampath et al. (1995): deterministic diagnoser method, exponential complexity
  Jiang et al. (2001), Yoo et al. (2002), etc.: twin plant method (Verifier),
  polynomial complexity but in a centralized way
- Pencolé (2004), Schumann et al. (2007): twin plant method in a distributed way
  (synchronization of local twin plants)
- Jéron et al. (2006): extension from fault event case to pattern case but in a
  centralized way
- Fabre et al. (2002), Qiu et al. (2006), Wang et al. (2007), etc.: several observers,
  check the existence of observer that can make non-ambiguous diagnosis
  decision by itself.
System model

- System model: a Finite State Machine (FSM) $G = (Q, \Sigma, \delta, q^0)$
- $\Sigma = \Sigma_o \cup \Sigma_u \cup \Sigma_f$
- $L(G) = \{s \in \Sigma^* | \exists q \in Q, (q^0, s, q) \in \delta\}$
- $L(G)/s = \{t \in \Sigma^* | s.t \in L(G)\}$
- Synchronization: any shared event occurs simultaneously
- Assumption: there is no unobservable cycle in the system and language is live.

Example
Diagnosability

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

A fault $F$ is diagnosable in a system $G$ iff

$$\exists k \in \mathbb{N}, \forall s^F \in \mathcal{L}(G), \forall t \in \mathcal{L}(G)/s^F, |t| \geq k \Rightarrow
(\forall p \in \mathcal{L}(G), P(p) = P(s^F.t) \Rightarrow F \in p).$$

**Critical pair**

A pair of trajectories $p \in \mathcal{L}(G), pt \in \mathcal{L}(G)$ satisfying the following conditions is a critical pair:

- $p$ contains $F$ and $pt$ does not
- $p$ has arbitrarily long observations after the occurrence of $F$
- $P(p) = P(pt)$

**Theorem**

A fault $F$ is diagnosable in a system $G$ iff there is no critical pair in $G$. 
Diagnosability

The same observations from the two red trajectories: \( \text{obs}=O1.(O2.O3) \)

The existence of critical pair violates diagnosability property
The same observations from the two red trajectories: obs = O1.(O2.O3)*

The existence of critical pair violates diagnosability property
The same observations from the two red trajectories: obs = O1.(O2.O3)*

The existence of critical pair violates diagnosability property
The same observations from the two red trajectories: obs=O1.(O2.O3)*

The existence of critical pair violates diagnosability property
The pre-diagnoser of the system $G$ is a FSM, $D = (Q_D, \Sigma_D, \delta_D, q^0_D)$ where $\Sigma_D = \Sigma_o$ and $q^0_D = (q^0, \emptyset)$. The transitions are those $((q, \ell), e, (q_f, \ell_f))$ with $(q, \ell)$ reachable from $q^0_D$ and satisfying: $\exists$ a path $p = (q \xrightarrow{u_1} q_1 \ldots \xrightarrow{u_m} q_m \xrightarrow{e} q_f)$ in $G$, with $u_k \in \Sigma_u, \forall k \in \{1, \ldots, m\}, e \in \Sigma_o$ and $\ell_f = \ell \cup (\{u_1, \ldots, u_m\} \cap \{F\})$. 

**System $G$**

**Pre-diagnoser**
Twin plant [Jiang et al. 2001]

The twin plant of the system $G$, denoted by $T$, is the FSM: $T = D \parallel \Sigma_o D$, where $D$ is the pre-diagnoser of the system $G$.

- ambiguous state: only one fault label in the twin plant state
- critical path: ambiguous state cycle with at least one observable event
- critical path $\iff$ critical pair
- fault diagnosable $\iff$ non existence of critical path

Pre-diagnoser

Part of twin plant
Outline

1. Introduction

2. **Diag_Abstraction**
   - Preliminaries
   - Original diagnosability information
   - Global consistency checking

3. **Diag_Pattern**
   - Preliminaries
   - Pattern recognition
   - Pattern diagnosability checking
   - Preliminary Implementation

4. **Diag_Cooperation**
   - Preliminaries
   - Cooperative diagnosability

5. Conclusion
Distributed system model

- System model: a set of Finite State Machines (FSM) $G_i = (Q_i, \Sigma_i, \delta_i, q_i^0)$
- $\Sigma_i = \Sigma_{io} \cup \Sigma_{iu} \cup \Sigma_{if} \cup \Sigma_{ic}$
- Assumption: any communication event is unobservable and is not fault event; in each component, no unobservable cycle and language is live.
Delay closure

Delay Closure

Given a FSM $G (Q, \Sigma, \delta, q^0)$, its delay closure with respect to $\Sigma_d$, where $\Sigma_d \subseteq \Sigma$, is $\mathcal{C}_{\Sigma_d}(G) = (Q, \Sigma_d, \delta_d, q^0)$, where $\delta_d(q, \sigma) = q'$ with $\sigma \in \Sigma_d$ if $\exists s \in (\Sigma \setminus \Sigma_d)^*$, $\delta(q, s\sigma) = q'$ in $G$.

Example

Delay closure with respect to $\Sigma_d = \{O1, O2, O3\}$

Delay closure with respect to $\Sigma_d = \{O1, O2\}$
Delay Closure

Given a FSM $G (Q, \Sigma, \delta, q^0)$, its delay closure with respect to $\Sigma_d$, where $\Sigma_d \subseteq \Sigma$, is $\mathbb{C}_{\Sigma_d}(G) = (Q, \Sigma_d, \delta_d, q^0)$, where $\delta_d(q, \sigma) = q'$ with $\sigma \in \Sigma_d$ if $\exists s \in (\Sigma \setminus \Sigma_d)^*$, $\delta(q, s\sigma) = q'$ in $G$.

Example

Delay closure with respect to $\Sigma_d = \{O1, O2, O3\}$

Delay closure with respect to $\Sigma_d = \{O1, O2\}$
**Regional Diagnosability** [YE et al. WODES-2010]

### Definition

A fault $F$ is regionally diagnosable in a system $G$ with respect to a subsystem $G_S$, where $F \in \Sigma_S$, iff

$$\exists k \in \mathbb{N}, \forall s^F \in L(G), \forall t \in L(G) \setminus s^F, |P_S(t)| \geq k \Rightarrow$$

$$(\forall p \in L(G), P_S(p) = P_S(s^F \cdot t) \Rightarrow F \in p).$$

Then $G_S$ is called a diagnosable subsystem.

### Diagnosis improvement

- Without a diagnosable subsystem: diagnosis decision requires all observations of the system.
- With a diagnosable subsystem: observations of the subsystem are sufficient to decide diagnosis

### Lemma

- If a fault $F$ is regionally diagnosable for $\Sigma_S$, then it is also regionally diagnosable for $G_{S'}$, where $G_S \subseteq G_{S'}$.
- The existence of a diagnosable subsystem $\Rightarrow$ diagnosability of the fault in the system.
Regional Diagnosability [YE et al. WODES-2010]

### Definition

A fault $F$ is regionally diagnosable in a system $G$ with respect to a subsystem $G_S$, where $F \in \Sigma_S$, iff

$$\exists k \in \mathbb{N}, \forall s^F \in L(G), \forall t \in L(G) \setminus s^F, |P_S(t)| \geq k \Rightarrow (\forall p \in L(G), P_S(p) = P_S(s^F.t) \Rightarrow F \in p).$$

Then $G_S$ is called a diagnosable subsystem.

### Diagnosis improvement

- Without a diagnosable subsystem: diagnosis decision requires all observations of the system.
- With a diagnosable subsystem: observations of the subsystem are sufficient to decide diagnosis

### Lemma

- If a fault $F$ is regionally diagnosable for $\Sigma_S$, then it is also regionally diagnosable for $G_{S'}$, where $G_S \subseteq G_{S'}$.
- The existence of a diagnosable subsystem $\Rightarrow$ diagnosability of the fault in the system
Local diagnoser construction

Local pre-diagnoser

The local pre-diagnoser of the component $G_i$ is a FSM, denoted by $D_i = (Q_D, \Sigma_D, \delta_D, q_D^0)$ where $\Sigma_D = \Sigma_i \cup \Sigma_{ic}$, and the transitions are those $((q, q_f), e, (q', q_{f'}))$ with $(q, q_f)$ reachable from $q_D^0$ and satisfying the following condition: $\exists$ a path $p = (q \xrightarrow{u_0} q_1 \cdots \xrightarrow{u_m} q_m \xrightarrow{e} q')$ in $G_i$ with $u_0 \in \Sigma_i \cup \Sigma_{ic}$, $\forall k \in \{1, \ldots, m\}$, $e \in \Sigma_{io} \cup \Sigma_{ic}$ and $q' = q_f \cup (\{u_0, \ldots, u_m\} \cap \{F\})$. 

Component $G_1$

Local pre-diagnoser of $G_1$
Local twin plant construction

Local twin plant (LTP) [Pencoé 2004]

The local twin plant (LTP) of the component $G_i$ is a FSM, denoted by $T_i = D_l^i \parallel \Sigma_{io} D_r^i$, where $D_l^i$ and $D_r^i$ are the left instance and right instance of the local pre-diagnoser of $G_i$. 

Left instance of the local pre-diagnoser

Right instance of the local pre-diagnoser
Part of LTP for $G_1$
Local possible critical path (LPCP)

Definition

In a LTP, a local path containing an ambiguous state cycle is called a local possible critical path (LPCP).

Relationship

The relations between critical pairs and LPCPs can be concluded as follows ($T_f$ is related to the component where $f$ occurs):

1. If there exists a critical pair in the system, its corresponding path in a LTP $T_f$ must be a LPCP
2. If there is no LPCP in $T_f$, then there is no critical pair in the system
3. If there is a LPCP in $T_f$, then it may or may not correspond to a critical pair in the system
LTP optimization [YE et al. ICTAI-2009]

**Redundant part in LTP**

- all paths without ambiguous state cycles
- equivalent LPCP (the same pair of local trajectories)

**Optimized LTP**

- reduce the left instance of local pre-diagnoser by retaining only paths with at least one fault state cycle
- reduce the right instance of local pre-diagnoser by retaining only paths with at least one cycle without fault state
- synchronize the reduced left one and the reduced right one

**Reduced left instance**

\[
\begin{aligned}
X_0 \xrightarrow{L:C_1} X_1 \xrightarrow{O_1} X_4 (F) \xrightarrow{O_2} X_5 (F) \xrightarrow{O_3} X_6 (F) \\
\end{aligned}
\]

**Reduced right instance**

\[
\begin{aligned}
X_0 \xrightarrow{Q_1} X_3 \xrightarrow{R:C_2} X_4 \xrightarrow{Q_2} X_5 \xrightarrow{Q_3} X_6 \xrightarrow{R:C_3} X_7 \\
\end{aligned}
\]
LTP optimization [YE et al. ICTAI-2009]

Redundant part in LTP
- all paths without ambiguous state cycles
- equivalent LPCP (the same pair of local trajectories)

Optimized LTP
- reduce the left instance of local pre-diagnoser by retaining only paths with at least one fault state cycle
- reduce the right instance of local pre-diagnoser by retaining only paths with at least one cycle without fault state
- synchronize the reduced left one and the reduced right one

Reduced left instance

Reduced right instance
From LTP to optimized LTP
From LTP to optimized LTP
Abstracted local twin plant (ALTP) [YE et al. WODES-2010]

ALTP construction

The abstracted local twin plant (ALTP) from a LTP $T_i$, denoted by $T_i^d$, is obtained by the following steps:

1. Delay Closure with respect to the set of communication events.
2. Recuperate the observable information for ambiguous state cycles (with communication).
3. Recuperate the observable ambiguous state cycles (without communication).

Optimized LTP
Abstracted local twin plant (ALTP) [YE et al. WODES-2010]

ALTP construction

The abstracted local twin plant (ALTP) from a LTP $T_i$, denoted by $T_i^a$, is obtained by the following steps:

1. Delay Closure with respect to the set of communication events.
2. Recuperate the observable information for ambiguous state cycles (with communication).
3. Recuperate the observable ambiguous state cycles (without communication).

Optimized LTP
Abstracted local twin plant (ALTP) [YE et al. WODES-2010]

**ALTP construction**

The abstracted local twin plant (ALTP) from a LTP $T_i$, denoted by $T_i^a$, is obtained by the following steps:

1. Delay Closure with respect to the set of communication events.
2. Recuperate the observable information for ambiguous state cycles (with communication).
3. Recuperate the observable ambiguous state cycles (without communication).

---

**Optimized LTP**

---

**ALTP**
Abstracted local twin checker (ALTC)

**Local twin checker (LTC)**

Given a component $G_i$, its local twin checker (LTC) is constructed as follows (similar to LTP except without fault information):

- operate delay closure with respect to the set of communication events and observable events
- synchronize its left instance and its right instance based on the set of observable events

**Component $G_2$**

**One path in the LTC of $G_2$**
Abstracted local twin checker (ALTC)

Given a LTC $C_i$, the ALTC $C_i^a$ is constructed as the same as ALTP from LTP:
- Delay Closure with respect to the set of communication events.
- Recuperate the existence of observable information for the cycles (with communication)
- Recuperate the observable cycles (without communication)

From LTC to ALTC
Abstracted local twin checker (ALTC)

Given a LTC $C_i$, the ALTC $C_i^a$ is constructed as the same as ALTP from LTP:

- Delay Closure with respect to the set of communication events.
- Recuperate the existence of observable information for the cycles (with communication)
- Recuperate the observable cycles (without communication)

From LTC to ALTC
Definition: globally consistent LPCP

Given a LPCP $\rho$ in $T_f^a$, if it does not disappear and contains an ambiguous state cycle with at least one observable event of all involved components in the FSM obtained by synchronizing $T_f^a$ with all connected ALTCs, it is said to be a globally consistent LPCP.

Lemma

A LPCP in $T_f^a$ is a globally consistent LPCP iff it corresponds to a critical pair.

Theorem

A fault $f$ is diagnosable in a system $G$ iff there is no globally consistent LPCP.
Definition: globally consistent LPCP

Given a LPCP $\rho$ in $T_f^a$, if it does not disappear and contains an ambiguous state cycle with at least one observable event of all involved components in the FSM obtained by synchronizing $T_f^a$ with all connected ALTCs, it is said to be a globally consistent LPCP.

Lemma

A LPCP in $T_f^a$ is a globally consistent LPCP iff it corresponds to a critical pair.

Theorem

A fault $f$ is diagnosable in a system $G$ iff there is no globally consistent LPCP.
Outline

1 Introduction

2 \textit{Diag\_Abstraction}
   \begin{itemize}
   \item Preliminaries
   \item Original diagnosability information
   \item Global consistency checking
   \end{itemize}

3 \textit{Diag\_Pattern}
   \begin{itemize}
   \item Preliminaries
   \item Pattern recognition
   \item Pattern diagnosability checking
   \item Preliminary Implementation
   \end{itemize}

4 \textit{Diag\_Cooperation}
   \begin{itemize}
   \item Preliminaries
   \item Cooperative diagnosability
   \end{itemize}

5 Conclusion
Distributed system model

Notation

- System model: a set of Finite State Machines (FSM) $G_i = (Q_i, \Sigma_i, \delta_i, q_i^0)$, $\Sigma_i = \Sigma_{ip} \cup \Sigma_{iu} \cup \Sigma_{ic}$
- A pattern is a deterministic complete FSM with stable final states set $F_\Omega$. [Jéron et al. 2006]
- Significant event: events changing pattern state.
- A simple pattern: only one sequence of significant events.
- Marked language: $L_m(\Omega) = \{ s \in L(\Omega) | \exists q \in F_\Omega, (q_\Omega^0, s, q) \in \delta_\Omega \}$

Components $G_1, G_2, G_3$

Pattern $\Omega$
**Distributed system model**

**Notation**
- System model: a set of Finite State Machines (FSM) \( G_i = (Q_i, \Sigma_i, \delta_i, q_i^0) \), \( \Sigma_i = \Sigma_{io} \cup \Sigma_{iu} \cup \Sigma_{ic} \)
- A pattern is a deterministic complete FSM with stable final states set \( F_\Omega \). [Jéron et al. 2006]
- Significant event: events changing pattern state.
- A simple pattern: only one sequence of significant events.
- Marked language: \( L_m(\Omega) = \{ s \in L(\Omega) | \exists q \in F_\Omega, (q_0^\Omega, s, q) \in \delta_\Omega \} \)

**Components \( G_1, G_2, G_3 \)**

**Pattern \( \Omega \)**
Distributed system model

Product

Given two FSMs $G_1$ and $G_2$, their product is $G_1 \times G_2 = (Q_1 \times Q_2, \Sigma_1 \cup \Sigma_2, \delta_1 \times \delta_2, (q^0_1, q^0_2))$, where $\delta_1 \times \delta_2((q_1, q_2), \sigma) = (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma))$ if both $\delta_1(q_1, \sigma)$ and $\delta_2(q_2, \sigma)$ are defined in $G_1, G_2$ respectively. Otherwise, $\delta_1 \times \delta_2((q_1, q_2), \sigma)$ is undefined in $G_1 \times G_2$.
Distributed system model

**Product**

Given two FSMs $G_1$ and $G_2$, their product is $G_1 \times G_2 = (Q_1 \times Q_2, \Sigma_1 \cup \Sigma_2, \delta_{1\times2}, (q_1^0, q_2^0))$, where $\delta_{1\times2}((q_1, q_2), \sigma) = (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma))$ if both $\delta_1(q_1, \sigma)$ and $\delta_2(q_2, \sigma)$ are defined in $G_1, G_2$ respectively. Otherwise, $\delta_{1\times2}((q_1, q_2), \sigma)$ is undefined in $G_1 \times G_2$.

**Examples**

![Product Examples Diagram]

**Synchronization**

![Synchronization Diagram]

**Product**

![Product Diagram]
Definitions

Pattern diagnosability [Jéron et al. 2006]

A pattern $\Omega$ is diagnosable in a system $G$, iff

$\exists n \in \mathbb{N}, \forall s \in L(G) \cap L_m(\Omega), \forall t \in L(G)/s$, if $|t| \geq n$, then $P^{-1}P(s.t) \subseteq L_m(\Omega)$.

Critical pair

A critical pair $p, p'$ of system $G$ with respect to the pattern $\Omega$ should satisfy three conditions:

- $p \in L_m(\Omega)$ and $p' \notin L_m(\Omega)$;
- $p$ is of arbitrarily long length after pattern recognition;
- $P(p) = P(p')$. 
Definitions

Pattern diagnosability [Jéron et al. 2006]

A pattern $\Omega$ is diagnosable in a system $G$, iff
$\exists n \in N, \forall s \in L(G) \cap L_m(\Omega), \forall t \in L(G)/s$, if $|t| \geq n$, then $P^{-1}P(s.t) \subseteq L_m(\Omega)$.

Critical pair

A critical pair $p, p'$ of system $G$ with respect to the pattern $\Omega$ should satisfy three conditions:

- $p \in L_m(\Omega)$ and $p' \notin L_m(\Omega)$;
- $p$ is of arbitrarily long length after pattern recognition;
- $P(p) = P(p')$. 
Pattern recognition

Pattern recognizer (Local pre-diagnoser)

Given a subsystem $G_S = (Q_S, \Sigma_S, \delta_S, q_0^S)$ and a pattern $\Omega = (Q_\Omega, \Sigma_\Omega, \delta_\Omega, q_0^\Omega, F_\Omega)$, then the pattern recognizer of $G_S$ is $R_{GS} = (Q_{RGS}, \Sigma_{RGS}, \delta_{RGS}, q_{RGS}^0, F_{RGS}) = G_S \times \Omega$, where the initial state is $q_{RGS}^0 = (q_0^S, q_0^\Omega)$, $F_{RGS} = (Q_S \times F_\Omega) \cap Q_{RGS}$ is the set of final states of the pattern recognizer.

- Target suspicious state (TSS): contain pattern state to which from initial pattern state the number of significant transitions is maximal in the pattern.
Pattern recognition

Pattern recognizer (Local pre-diagnoser)

Given a subsystem $G_S = (Q_S, \Sigma_S, \delta_S, q^0_S)$ and a pattern $\Omega = (Q_\Omega, \Sigma_\Omega, \delta_\Omega, q^0_\Omega, F_\Omega)$, then the pattern recognizer of $G_S$ is $R_{GS} = (Q_{RGS}, \Sigma_{RGS}, \delta_{RGS}, q^0_{RGS}, F_{RGS}) = G_S \times \Omega$, where the initial state is $q^0_{RGS} = (q^0_S, q^0_\Omega)$, $F_{RGS} = (Q_S \times F_\Omega) \cap Q_{RGS}$ is the set of final states of the pattern recognizer.

- Target suspicious state (TSS): contain pattern state to which from initial pattern state the number of significant transitions is maximal in the pattern.
Incremental recognition [YE et al. DX-2009]

- Given a $R_{GS}$, if $F_{R_{GS}} = \emptyset$, it is not a complete recognizer $\Rightarrow$ extension of subsystem

- Recognizer reduction
  1. retain the paths with TSS
  2. retain the paths with the same observations as any one retained in step 1

- Subsystem extension: synchronize reduced recognizer with next component, which contain the significant event whose source state in pattern is that in a TSS of this recognizer
Incremental recognition [YE et al. DX-2009]

- Given a $R_{GS}$, if $F_{RG_S} = \emptyset$, it is not a complete recognizer $\Rightarrow$ extension of subsystem
- Recognizer reduction
  1. retain the paths with TSS
  2. retain the paths with the same observations as any one retained in step 1
- Subsystem extension: synchronize reduced recognizer with next component, which contain the significant event whose source state in pattern is that in a TSS of this recognizer
Incremental recognition [YE et al. DX-2009]

- Given a $R_{GS}$, if $F_{RGS} = \emptyset$, it is not a complete recognizer $\Rightarrow$ extension of subsystem

Recognizer reduction

1. retain the paths with TSS
2. retain the paths with the same observations as any one retained in step 1

Subsystem extension: synchronize reduced recognizer with next component, which contain the significant event whose source state in pattern is that in a TSS of this recognizer
Incremental recognition [YE et al. DX-2009]

- Given a $R_{GS}$, if $F_{R_{GS}} = \emptyset$, it is not a complete recognizer $\Rightarrow$ extension of subsystem
- Recognizer reduction
  1. retain the paths with TSS
  2. retain the paths with the same observations as any one retained in step 1
- Subsystem extension: synchronize reduced recognizer with next component, which contain the significant event whose source state in pattern is that in a TSS of this recognizer
Incremental recognition [YE et al. DX-2009]

- Given a $R_{GS}$, if $F_{RG_g} = \emptyset$, it is not a complete recognizer ⇒ extension of subsystem

- Recognizer reduction
  1. retain the paths with TSS
  2. retain the paths with the same observations as any one retained in step 1

- Subsystem extension: synchronize reduced recognizer with next component, which contain the significant event whose source state in pattern is that in a TSS of this recognizer
From recognition to diagnosability

Pattern verifier (local twin plant)

The pattern verifier is constructed from the complete recognizer as follows:

- delay closure to keep the set of observable events and communication events
- reduce the left instance by retaining only paths with at least one fault cycle (cycle with at least one observable event for all involved components)
- reduce the right instance by retaining only paths with at least one cycle without fault (cycle with at least one observable event for all involved components)
- synchronize the reduced left one and the reduced right one

Complete recognizer

Pattern verifier

[Diagram of the complete recognizer and pattern verifier]
From recognition to diagnosability

Pattern verifier (local twin plant)

The pattern verifier is constructed from the complete recognizer as follows:

- delay closure to keep the set of observable events and communication events
- reduce the left instance by retaining only paths with at least one fault cycle (cycle with at least one observable event for all involved components)
- reduce the right instance by retaining only paths with at least one cycle without fault (cycle with at least one observable event for all involved components)
- synchronize the reduced left one and the reduced right one

Complete recognizer

Pattern verifier
From recognition to diagnosability

Pattern verifier (local twin plant)

The pattern verifier is constructed from the complete recognizer as follows:

- delay closure to keep the set of observable events and communication events
- reduce the left instance by retaining only paths with at least one fault cycle (cycle with at least one observable event for all involved components)
- reduce the right instance by retaining only paths with at least one cycle without fault (cycle with at least one observable event for all involved components)
- synchronize the reduced left one and the reduced right one
Abstracted pattern verifier (ALTP)

The abstracted pattern verifier (APV) from a pattern verifier is obtained by the following steps:

1. **Delay Closure** with respect to the set of communication events.

2. Recuperate the existence of observable information for ambiguous cycles with communication events containing observations for all involved components.

3. Recuperate the ambiguous cycles (without communication) containing observations for all involved components.

From pattern verifier to abstracted pattern verifier

![Diagram showing the transition from pattern verifier to abstracted pattern verifier]

- **Y0P0** to **Y2P1**
- **Y0P0** to **Y5P0**
- **Y3P2** to **Y6P0**
- **Y3P2** to **Y4P2**
- **Y3P2** to **Y7P0**

- **Y0P0** to **Y2P1**
- **Y0P0** to **Y6P0**
- **Y3P2** to **q2**
- **obs1**, **obs2**
Global consistency checking

Partial critical path

In the (abstracted) pattern verifier, a path containing an ambiguous state cycle with observable information for each involved component is called partial critical path.

Global consistency checking

The global consistency checking of a partial critical path consists in synchronizing it with all connected ALTCs, if after the synchronization, this path contains ambiguous cycles with observable information for all involved components, then it is globally consistent.

Theorem

A pattern is diagnosable in the system iff there is no globally consistent partial critical path.
Partial critical path

In the (abstracted) pattern verifier, a path containing an ambiguous state cycle with observable information for each involved component is called partial critical path.

Global consistency checking

The global consistency checking of a partial critical path consists in synchronizing it with all connected ALTCs, if after the synchronization, this path contains ambiguous cycles with observable information for all involved components, then it is globally consistent.

Theorem

A pattern is diagnosable in the system iff there is no globally consistent partial critical path.
Discussion

Search space reduction

- Incremental pattern recognition normally smaller than global recognition (global model)
- Optimized pattern verifier smaller than global pattern verifier in most of the case
- During global consistency checking, ALTC is used instead of LTC
- Before each synchronization with ALTC, we retain only paths with consistent ambiguous cycles instead of the whole one
- Heuristic selection strategy
Test case

System

Pattern
Search space results

- Number of states
  - Distributed algorithm
  - Centralized algorithm

- Number of transitions
  - Distributed algorithm
  - Centralized algorithm
Test case

Search space results

- Global PR
- Global PV
- Distributed PR
- Distributed PV

Graphs showing the relationship between the number of components and the states/number of transitions.
Results

Number of states

- Distributed with abstraction and reduction
- Distributed without abstraction and reduction
- Centralized

Number of transitions

- Distributed with abstraction and reduction
- Distributed without abstraction and reduction
- Centralized
## Outline

1. **Introduction**

2. **Diag_Abstraction**
   - Preliminaries
   - Original diagnosability information
   - Global consistency checking

3. **Diag_Pattern**
   - Preliminaries
   - Pattern recognition
   - Pattern diagnosability checking
   - Preliminary Implementation

4. **Diag_Cooperation**
   - Preliminaries
   - Cooperative diagnosability

5. **Conclusion**
Distributed discrete event system (distributed observations)

- System model: a set of Finite State Machines (FSM) $G_i = (Q_i, \Sigma_i, \delta_i, q_i^0)$
- $\Sigma_i = \Sigma_{i_0} \cup \Sigma_{i_u} \cup \Sigma_{i_f} \cup \Sigma_{i_c}$
- $P_i(s)$: the projection of a trajectory $s$ to locally observable event set $\Sigma_{i_o}$ of $G_i$

Communication compatibility [YE et al. ECAI-2010]

In a system $G$, given two infinite local trajectories $s \in L(G_i), s' \in L(G_j)$, where $G_i \neq G_j$, they are communication compatible if the following conditions are satisfied:

- $\forall \sigma \in s, \sigma \in \Sigma_{i_c}$, if $\sigma \in \Sigma_{j_c}$, then $\sigma \in s'$;
- $\forall \sigma \in s', \sigma \in \Sigma_{j_c}$, if $\sigma \in \Sigma_{i_c}$, then $\sigma \in s$;
- $\forall (\sigma, \sigma')$, where $\sigma \in s, \sigma' \in s, \sigma' \in s'$ and $(\sigma, \sigma')$ in $\Sigma_{i_c} \times \Sigma_{j_c}$, the whole sequence of $\sigma$ and $\sigma'$ in $s$ is identical to its analog in $s'$. 
Distributed discrete event system (distributed observations)

- System model: a set of Finite State Machines (FSM) $G_i = (Q_i, \Sigma_i, \delta_i, q_i^0)$
- $\Sigma_i = \Sigma_{io} \cup \Sigma_{iu} \cup \Sigma_{if} \cup \Sigma_{ic}$
- $P_i(s)$: the projection of a trajectory $s$ to locally observable event set $\Sigma_{io}$ of $G_i$

Communication compatibility [YE et al. ECAI-2010]

In a system $G$, given two infinite local trajectories $s \in L(G_i), s'l \in L(G_j)$, where $G_i \neq G_j$, they are communication compatible if the following conditions are satisfied:

- $\forall \sigma \in s, \sigma \in \Sigma_{ic}$, if $\sigma \in \Sigma_{jc}$, then $\sigma \in s'$;
- $\forall \sigma \in s', \sigma \in \Sigma_{jc}$, if $\sigma \in \Sigma_{ic}$, then $\sigma \in s$;
- $\forall (\sigma, \sigma')$, where $\sigma \in s, \sigma' \in s, \sigma \in s', \sigma' \in s'$ and $(\sigma, \sigma')$ in $\Sigma_{ic} \times \Sigma_{jc}$, the whole sequence of $\sigma$ and $\sigma'$ in $s$ is identical to its analog in $s'$.
Cooperative Diagnosability [YE et al. ECAI-2010]

**Cooperative diagnosability**

A fault $F$ is cooperatively diagnosable in a system $G$, with a set of autonomous components \( \{G_1, \ldots G_n\} \), iff

\[
\exists k \in \mathbb{N}, \forall s^F \in L(G), \forall t \in L(G) \setminus s^F, \left( \forall i \in \{1, \ldots, n\}, |P_i(t)| \geq k \Rightarrow (\forall p \in L(G)
(\forall i \in \{1, \ldots, n\}, P_i(p) = P_i(s^F.t)) \Rightarrow F \in p) \right).
\]

**Undecidable pair**

A pair of trajectories $p \in L(G), p' \in L(G)$ satisfying the following conditions is an undecidable pair:

- $p$ contains $F$ and $p'$ does not
- $p$ has arbitrarily long local observations in all components after the occurrence of $F$
- $\forall i \in \{1, \ldots, n\}, P_i(p) = P_i(p')$

**Theorem**

A fault $F$ is cooperatively diagnosable in a system $G$ iff there is no undecidable pair in $G$. 
The existence of undecidable pair violates cooperative diagnosability property.
Cooperative Diagnosability

The existence of undecidable pair violates cooperative diagnosability property
Cooperative Diagnosability

The existence of undecidable pair violates cooperative diagnosability property
Local twin plant

Local pre-diagnoser

Local twin plant

Local pre-diagnoser

Local twin plant
Communication compatible path [YE et al. ECAI-2010]

Definition

A path $\varphi_i$ in the local twin plant or in the local twin checker of $G_i$ is communication compatible with a path $\varphi_j$ in the local twin plant or in the local twin checker of $G_j$ if the corresponding left infinite trajectory of $\varphi_i$ in $G_i$ is communication compatible with the corresponding left infinite trajectory of $\varphi_j$ in $G_j$, and right with right.

Example
Definition

A path $\varrho_i$ in the local twin plant or in the local twin checker of $G_i$ is communication compatible with a path $\varrho_j$ in the local twin plant or in the local twin checker of $G_j$ if the corresponding left infinite trajectory of $\varrho_i$ in $G_i$ is communication compatible with the corresponding left infinite trajectory of $\varrho_j$ in $G_j$, and right with right.

Example
Diagnosability condition [YE et al. ECAI-2010]

**Local critical path compatibility**

In a system $G$ with autonomous components, a local critical path $\varrho_f$ in the local twin plant of $G_f$ is (communication) compatible in a subsystem $G_s$, where $G_f \subseteq G_s$, if $\forall G_i \in G_s \setminus \{G_f\}$, there exists a path with enough local observations in the local twin checker of $G_i$, such that this set of paths satisfy the following conditions:

1. $\forall G_i \in G_s \setminus \{G_f\}$, $\varrho_i$ is communication compatible with $\varrho_f$;
2. $\forall (i, j), i \neq j, G_i \in G_s \setminus \{G_f\}, G_j \in G_s \setminus \{G_f\}$, $\varrho_i$ is communication compatible with $\varrho_j$.

**Lemma**

In a system $G$, there exists a local critical path that is globally compatible iff there exists an undecidable pair.

**Theorem**

A fault $F$ is cooperatively diagnosable in system $G$ iff there is no local critical path that is globally compatible.
Diagnosability condition [YE et al. ECAI-2010]

Local critical path compatibility

In a system $G$ with autonomous components, a local critical path $\rho_f$ in the local twin plant of $G_f$ is (communication) compatible in a subsystem $G_S$, where $G_f \subseteq G_S$, if $\forall G_i \in G_S \backslash \{G_f\}$, $\exists \rho_i$, a path with enough local observations in the local twin checker of $G_i$, such that this set of paths satisfy the following conditions:

1. $\forall G_i \in G_S \backslash \{G_f\}$, $\rho_i$ is communication compatible with $\rho_f$;
2. $\forall (i, j), i \neq j, G_i \in G_S \backslash \{G_f\}, G_j \in G_S \backslash \{G_f\}$, $\rho_i$ is communication compatible with $\rho_j$.

Lemma

In a system $G$, there exists a local critical path that is globally compatible iff there exists an undecidable pair.

Theorem

A fault $F$ is cooperatively diagnosable in system $G$ iff there is no local critical path that is globally compatible.
Communication compatibility checking (reuse ALTP and ALTC)

**Left communication compatibility checking**

- Rename the right communication events by prefixing component ID in ALTP and connected ALTC
- Synchronize ALTP and connected ALTC based on common left communication event
- Keep the paths with cycles containing observations of all involved components

**Right communication compatibility checking**

- From the resulted FSM obtained for now, perform delay closure w.r.t right communication events and observable events for components
- Rename right communication events by removing the prefix of component ID
- Synchronize them based on common right communication events
- Keep the paths with cycles containing observations of all involved components, which correspond to globally compatible local critical path, whose existence verifies non cooperative diagnosability.
Example

Left communication compatibility checking

Right communication compatibility checking
Example

**Left communication compatibility checking**

$$
\begin{align*}
X_0 & \xrightarrow{L:C_1} X_1 \\
Z_0 & \xrightarrow{L:C_3} Z_3
\end{align*}
$$

**Right communication compatibility checking**

$$
\begin{align*}
S_0 & \xrightarrow{obs_3} S_6 \\
S_0 & \xrightarrow{obs_1} S_4 \\
S_0 & \xrightarrow{obs_1} S_9
\end{align*}
$$
Example

Left communication compatibility checking

Right communication compatibility checking
Example

Left communication compatibility checking

Right communication compatibility checking
Example

Left communication compatibility checking

Right communication compatibility checking
Outline

1 Introduction

2 *Diag* _Abstraction*
   - Preliminaries
   - Original diagnosability information
   - Global consistency checking

3 *Diag* _Pattern*
   - Preliminaries
   - Pattern recognition
   - Pattern diagnosability checking
   - Preliminary Implementation

4 *Diag* _Cooperation*
   - Preliminaries
   - Cooperative diagnosability

5 Conclusion
Propose an optimized algorithm for distributed diagnosability analysis
- Define regional diagnosability and diagnosable subsystem
- Obtain original diagnosability information from ALTP of $G_f$
- Check global consistency through synchronizing with all connected ALTCs

Extend distributed diagnosability from fault event to fault pattern
- Incrementally recognize pattern by extending subsystem but avoid global model
- Construct abstracted pattern verifier to obtain original diagnosability information
- Check global consistency through synchronizing with connected ALTCs
- Easy extension from simple pattern to general pattern

Propose a new cooperative diagnosability algorithm
- Propose cooperative diagnosis architecture and define cooperative diagnosability (distributed observations)
- Provide the necessary and sufficient condition to verify cooperative diagnosability (the existence of undecidable pairs)
- Check cooperative diagnosability by analyzing communication compatibility between ALTP and connected ALTCs through synchronization.
Future work

- Optimization of our algorithm for cooperative diagnosability verification
- Validation on real distributed systems
- For a diagnosable subsystem, how to reduce observations but still maintain its diagnosability; for a system that is not diagnosable, how to upgrade its diagnosability level
- Extension of our approach to analyze predictability of distributed systems (stronger)
- Research more profound from distributed diagnosability to distributed diagnosis
Publication

- L. Ye and P. Dague, Diagnosability Analysis of Discrete Events Systems with Autonomous Components, 19th European Conference on Artificial Intelligence (ECAI-2010), Lisbon, Portugal, August 2010
- L. Ye and P. Dague, Diagnosability of patterns in distributed discrete event systems, 7th Symposium on Fault Detection, Supervision and Safety for Technical Processes, (SAFEPROCESS-09), Barcelona, Spain, June-July 2009
- L. Ye, P. Dague et Y. Yan, An incremental approach for pattern diagnosability in distributed discrete event systems, 21st International Conference on Tools with Artificial Intelligence (ICTAI-09), Newark, NJ, USA, november 2009
- Y. Li, T. Melliti, L. Ye and P. Dague, A decentralized model-based diagnosis for BPEL services, 21st International Conference on Tools with Artificial Intelligence (ICTAI-09), Newark, NJ, USA, November 2009
- L. Ye and P. Dague, Decentralized Diagnosis for BPEL Web Services, 4th International Conference on Web Information Systems and Technologies (WEBIST-08), Funchal, Portugal, May 2008
Thanks!
Appendix

Cooperative diagnosis
Figure: Cooperative diagnosis architecture
Diagnosis without cooperation

<table>
<thead>
<tr>
<th></th>
<th>( P_1 )</th>
<th>( D_1 )</th>
<th>( P_2 )</th>
<th>( D_2 )</th>
<th>( P_3 )</th>
<th>( D_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( O1O2^* )</td>
<td>( F(C1C3^<em>) ) ( \emptyset(C2C3^</em>) )</td>
<td>( O3O4^* )</td>
<td>( (C1C3^<em>) ) ( (C3^</em>) )</td>
<td>( O6^* )</td>
<td>( C3^* )</td>
</tr>
</tbody>
</table>
Example

Diagnosis with cooperation

<table>
<thead>
<tr>
<th>$P_1$</th>
<th>$D_1$</th>
<th>$P_2$</th>
<th>$D_2$ update with the inference of $D_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O1O2^*$</td>
<td>$F(C1C3^<em>)$ in $G_1$ compatible with $C1C3^</em>$ in $G_2$</td>
<td>$O3O4^*$</td>
<td>$F(C1C3^<em>$ in $G_1$ compatible with $C1C3^</em>$ in $G_2$)</td>
</tr>
<tr>
<td></td>
<td>$\emptyset(C2C3^*)$</td>
<td></td>
<td>$\emptyset(C2C3^<em>$ in $G_1$ compatible with $C3^</em>$ in $G_2$)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$P_3$</th>
<th>$D_3$ update with the inference of $D_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O6^*$</td>
<td>$F(C1C3^<em>$ in $G_1$ compatible with $C1C3^</em>$ in $G_2$ compatible with $C3^*$ in $G_3$)</td>
</tr>
<tr>
<td></td>
<td>$\emptyset(C2C3^<em>$ in $G_1$ compatible with $C3^</em>$ in $G_2$ not compatible with $C3^*$ in $G_3$)</td>
</tr>
</tbody>
</table>
Incremental approach

Component G1

Component G2

Pattern

Pattern recognizer for G1