A New Protocol for Distributed Diagnosis

Wenbin Qiu (qiuw@corning.com), Corning Incorporated, Corning, NY 14831*
Ratnesh Kumar (rkumar@iastate.edu), Iowa State University, Ames, IA 50011

Abstract—In this paper, we propose a new protocol for distributed diagnosis, where distributed diagnosers share their diagnosis information through bounded-delay channels. In our previous work [13], distributed diagnosis was studied based on an immediate observation passing (iop) protocol, where each local site transmits its observations to other sites immediately after each observation through bounded-delay channels. It was shown that the complexity of constructing local diagnosers is exponential in the number of local sites, which limits the applicability of iop-based distributed diagnosis to large distributed networks. A synchronized diagnosis estimate truncation strategy is proposed to further reduce the space requirement. It is shown that the new diagnosis protocol has the same diagnosis capability as the iop-protocol. An example is provided to illustrate the distributed diagnosis procedure under the new protocol.

Index Terms—Discrete event systems, failure diagnosis, distributed systems, communication delay.

I. INTRODUCTION

Failure diagnosis is an active area of research, and has received considerable attention in the literature. A failure is a deviation from an expected or desired behavior. Various approaches have been proposed for failure diagnosis, including fault-trees, expert systems, neural networks, fuzzy logic, bayesian networks, and analytical redundancy [11]. These are broadly categorized into non-model based (where observed behavior is matched to known failures), and model based (where observed behavior is compared against model predictions for any abnormality).

For discrete event systems (DES) the task of diagnosis of a system requires detecting the occurrence of a failure by observing the system behavior, whereas the diagnosability property requires that the occurrence of a failure be detected within a bounded delay. For untimed discrete-event systems diagnosability has been examined in [16], [23], [6], [22], and a stronger notion of state-observability was examined in [10]. Extensions to decentralized setting can be found in [5], [12], [20] and to distributed setting in [5], [17], [15], [2], [18], [13], [14]. Extensions to diagnosis of repeatable/intermittent-failures can be found in [9], [21], [7], [3], [24], to the temporal logic setting in [8], [7], and to the probabilistic setting in [19].

The research was supported in part by the National Science Foundation under the grants NSF-ECS-0219207, NSF-ECS-0247132, NSF-EPNET-0323379, and NSF-ECS-0424048.

*This work was done when the author was with Iowa State University, Ames, IA 50010.
II. Notions and Preliminaries

Given an event set $\Sigma$, $\Sigma^*$ denotes the set of all finite length event sequences over $\Sigma$, including the zero length event sequence $\epsilon$. A member of $\Sigma^*$ is a trace and a subset of $\Sigma^*$ is a language. A generated language $L \subseteq \Sigma^*$, it is said to be prefix-closed if $L = pr(L)$, where $pr(L) := \{ s \in \Sigma^* | \exists t \in \Sigma^* \text{ s.t. } st \in L \}$. Given two traces $s$ and $t$, $s \leq t$ represents that $s$ is a prefix of $t$ ($s \in pr(t)$), and $s < t$ represents that $s$ is a strict prefix of $t$ ($s \in pr(t), s \neq t$).

A DES is modeled as a finite automaton $G = (X, \Sigma, \alpha, x_0)$, where $X$ is the set of states, $\Sigma$ is the finite set of events, $x_0 \in X$ is the initial state, and $\alpha : X \times \Sigma \rightarrow 2^X$ is the transition function with $\Sigma := \Sigma \cup \{\epsilon\}$. $G$ is said to be deterministic if $|\alpha(\cdot, \cdot)| \leq 1$ and $|\alpha(\cdot, \epsilon)| = 0$; otherwise, it is called nondeterministic. The generated language of $G$ is given by, $L(G) := \{ s \in \Sigma^* | \alpha(x_0, s) \neq \emptyset \}$. A path in $G$ is a sequence of transitions $x_1 \cdot \sigma_1 \cdot x_2 \cdot \sigma_2 \cdots \sigma_{n-1} \cdot x_n$, where $x_i \in X$, $\sigma_i \in \Sigma$ and $x_{i+1} \in \alpha(x_i, \sigma_i)$ for all $i \in \{1, \ldots, n-1\}$. The set of all paths in $G$ is defined as $\Pi(G) := X \times (\Sigma \cdot X)^*$. For a path $\pi = x_1 \cdot \sigma_1 \cdot x_2 \cdot \sigma_2 \cdots \sigma_{n-1} \cdot x_n \in \Pi(G)$, $tr(\pi) = \sigma_1 \sigma_2 \cdots \sigma_{n-1}$ defines the event trace associated with path $\pi$.

Given two automata $G_1 = (X_1, \Sigma_1, \alpha_1, x_{0,1})$ and $G_2 = (X_2, \Sigma_2, \alpha_2, x_{0,2})$, the synchronous composition of $G_1$ and $G_2$ is defined as, $G_1 \parallel G_2 = (X_1 \times X_2, \Sigma_1 \cup \Sigma_2, \alpha(x_{1,1}, x_{1,2}))$, where $\alpha$ is defined as follows: $\forall (x_1, x_2) \in X_1 \times X_2, \sigma \in \Sigma_1 \cup \Sigma_2, \alpha((x_1, x_2), \sigma) :=$

$$\begin{cases} 
 \alpha_1(x_1, \sigma) \times \alpha_2(x_2, \sigma) & \text{if } \sigma \in \Sigma_1 \cap \Sigma_2; \\
 \alpha_1(x_1, \sigma) \times \{x_2\} & \text{if } \sigma \in \Sigma_1 - \Sigma_2; \\
 \{x_1\} \times \alpha_2(x_2, \sigma) & \text{if } \sigma \in \Sigma_2 - \Sigma_1,
\end{cases}$$

and $\alpha((x_1, x_2), \epsilon) := \alpha_1(x_1, \epsilon) \times \{x_2\} \cup \{x_1\} \times \alpha_2(x_2, \epsilon)$.

When the system execution is observed by a global observer, we can define a global observation mask, $M : \Sigma \rightarrow \mathbb{X}$ with $M(\epsilon) = \epsilon$, where $\mathbb{X} := \Lambda \cup \{\epsilon\}$ and $\Lambda$ is the set of observed symbols. The definition of $M$ can be extended from events to event sequences inductively as follows: $M(\sigma) = \{x \in \Sigma^* | x' \in \epsilon_\sigma(x) \Rightarrow \alpha(x', \epsilon) \subseteq \epsilon_\sigma(x)\}$. The domain of the state transition function $\alpha$ can be extended from $X \times \Sigma$ to $X \times \Sigma^*$ recursively as follows: $\forall x \in X, s \in \Sigma^*, \sigma \in \Sigma, \alpha(x, s) = \epsilon_\sigma(x, \epsilon) \cup \alpha(x, \sigma)$.

Given a plant $G$ and a deterministic specification model $R = (Y, \Sigma, \beta, y_0)$, the specification language $K = L(R)$ specifies the desired behavior in the plant. Execution of any trace outside of $K$ is viewed as the occurrence of a fault. The completed specification model $\overline{R}$ is defined as $\overline{R} := (\overline{Y}, \Sigma, \overline{\beta}, y_0)$, where $\overline{Y} := Y \cup \{F\}$, and $\overline{\beta}$ is defined as:

$$\forall \overline{y} \in \overline{Y}, \sigma \in \Sigma,$$

$$\overline{\beta}(\overline{y}, \sigma) := \begin{cases} 
 \beta(\overline{y}, \sigma), & \text{if } [\overline{y} \in Y] \land [\beta(\overline{y}, \sigma) \neq \emptyset]; \\
 F, & \text{if } [\overline{y} = F] \lor [\beta(\overline{y}, \sigma) = \emptyset].
\end{cases}$$

Using the completed specification model $\overline{R}$, the refined plant model $\overline{G}$ can be defined as $\overline{G} := G|\overline{R}$, where $L(\overline{G}) = L(G)$ and all states reached by failure traces have second coordinate “$F$” (also labeled by $F$).

The failure diagnosis problem is to detect and diagnose any failure behavior in $L(G) - K$ within a bounded delay of its execution. When there does not exist any communication among the local diagnoser sites, it is called a decentralized failure diagnosis problem; otherwise, it is called a distributed failure diagnosis problem.

III. DISTRIBUTED DIAGNOSIS PROTOCOL WITH SYNCHRONIZED TRUNCATION

In the proposed distributed diagnosis protocol with synchronized truncation, each local site maintains a diagnosis estimate, which tracks the evolution history of the system and is updated when a local observation or a communicated diagnosis information from another site is received. Using the diagnosis estimate, the set of states that the plant may have currently reached, referred to as the reachability set, can be computed. Once all states in the reachability set are labeled by $F$, the local diagnoser reports that a failure has been detected. To reduce the space complexity of maintaining diagnosis estimates, a distributed protocol for the synchronized truncation of the estimates at each of the local sites is proposed. A truncation operation removes an initial part of the history from a diagnosis estimate. A key challenge is to do the truncation in a distributed yet synchronous manner so that all local diagnosers remove a same part of the initial history and there by maintain a “synchrony” of their diagnosis estimates.

A. Diagnosis Information

Given a plant $G$ and a specification model $R$, we first construct the refined plant model $\overline{G} = G|\overline{R} = (Z, \Sigma, \gamma, z_0)$. Then the diagnosis information $\phi_i = (E_i, T_i, C_{f_i}, T_{f_i})$ at site $i$ consists of the following:

- $E_i \subseteq \Pi(\overline{G}) \subseteq Z \cdot (\Sigma \cdot Z)^*$ is a diagnosis estimate at site $i$, which keeps tracks of all possible paths the refined plant $\overline{G}$ may have executed since the last time the diagnosis estimate was truncated;
- $T_i \subseteq \Pi(\overline{G}) \subseteq Z \cdot (\Sigma \cdot Z)^*$ is an estimate truncation, which is a certain set of prefixes of the diagnosis estimate that is either proposed or being used for truncating the local diagnosis estimate $E_i$ at site $i$;
- $C_{f_i}$ is the flag of truncation candidate, which equals 1 if and only if a truncation candidate is proposed at site $i$;
- $T_{f_i}$ is the flag of truncation, which equals to 1 if and only if $T_i$ has been truncated from $E_i$.

A diagnoser $D_i$ at site $i$ is responsible for maintaining and updating the diagnosis information, and sharing and
synchronizing that information with other diagnosers. Besides the diagnosis information \( \phi_i \), an index set \( \Omega_i \subseteq I \) is also maintained at site \( i \). In the proposed protocol, there are three phases within an entire cycle of truncation. \( \Omega_i \) contains indices of all diagnosers whose diagnosis information has been collected at site \( i \) in the current diagnosis phase. Initially, \( \Omega_i := \{ i \} \).

**B. Basic Idea**

In the following, we discuss the method to (i) update the local diagnosis estimate when a new observation from the plant or a communicated diagnosis information from another site arrives, and (ii) perform synchronized truncation distributively.

When diagnoser \( D_i \) receives a new observation \( \lambda_i \in \Lambda_i \) from the plant, it updates its diagnosis estimate \( E_i \), and sends the diagnosis information \( \phi_i = \{ E_i, T_i, C_{f_i}, T_{f_i} \} \) to other diagnosers with \( T_i, C_{f_i}, \) and \( T_{f_i} \) unchanged.

When diagnoser \( D_i \) receives a communicated diagnosis information \( \phi_j = \{ E_j, T_j, C_{f_j}, T_{f_j} \} \) from another diagnoser \( D_j \) \( (j \neq i) \), it uses that information to get a more accurate estimate by fusing \( E_j \) with its own estimate \( E_i \). From the information-fusion point of view, such prefixes of paths in a diagnosis estimate that are common among all diagnosers are not useful in obtaining a further accurate estimate. So one can reduce the space requirement and communication burden by truncating that redundant part of the diagnosis estimate.

For the purposes of truncating the diagnosis estimates synchronously among all distributed sites, a three-phase truncation protocol is proposed. Each diagnoser may be in one of these three phases at any time. The operations performed in each phase at site \( i \) are described as follows.

**Phase 1 – Local truncation computation \((C_{f_i} = 0, T_{f_i} = 0)\):** In this phase, a diagnoser computes a local truncation candidate based on the diagnosis information received from other diagnosers, and sends that truncation candidate to other diagnosers so as to arrive at a global truncation candidate. When diagnoser \( D_i \) receives a diagnosis information from diagnoser \( D_j \), then depending on the phase of diagnoser \( D_j \) it performs a different set of operations as described below:

- **\( D_j \) in Phase 1 \((C_{f_j} = 0, T_{f_j} = 0)\):** First, diagnoser \( D_i \) combines estimate \( E_j \) received from diagnoser \( D_j \) with its own estimate \( E_i \). Then, it computes a local truncation candidate \( T_i \), and adds the index \( j \) into the set \( \Omega_i \). Once \( \Omega_i \) covers all diagnosers, i.e., \( \Omega_i := I \), diagnoser \( D_i \) broadcasts information \( \phi_i = \{ E_i, T_i, C_{f_i}, T_{f_i} \} \) to other diagnosers with \( C_{f_i} = 1 \), indicating that \( D_i \) has proposed a truncation candidate. At this point diagnoser \( D_i \) transits from phase 1 to phase 2, and resets the set of indices, i.e., \( \Omega_i := \{ i \} \).

- **\( D_j \) in Phase 2 \((C_{f_j} = 1, T_{f_j} = 0)\):** In this case, diagnoser \( D_i \) receives a diagnosis information from diagnoser \( D_j \) which is in phase 2, i.e., it has proposed a truncation candidate \( T_j \). This implies that diagnoser \( D_j \) has collected diagnosis estimates from all local diagnosers. So diagnoser \( D_i \) simply assigns \( T_i := T_j \), broadcasts its diagnosis information \( \phi_i \), resets \( \Omega_i := \{ i \} \), and transits to phase 2.

- **\( D_j \) in Phase 3 \((C_{f_j} = 0, T_{f_j} = 1)\):** This case can exist only in the scenario that diagnoser \( D_j \) has completed a cycle of truncation and returned to phase 1 while diagnoser \( D_j \) is still in phase 3, and before \( D_j \) has been able to collect all truncation acknowledgments, it has received a new observation and sent out the updated diagnosis estimate to \( D_i \). In this case, diagnoser \( D_i \) only updates its own diagnosis estimate \( E_i \) with respect to the estimate \( E_j \) received from \( D_j \), and does not initiate any further communication.

**Phase 2 – Global truncation computation \((C_{f_i} = 1, T_{f_i} = 0)\):** In this phase, diagnoser \( D_i \) has already proposed a local truncation candidate. Once it collects truncation candidates proposed by all other diagnosers, it computes a global truncation candidate, performs the actual truncation, and transits to phase 3. In phase 2, when diagnoser \( D_i \) receives a diagnosis information from diagnoser \( D_j \), then depending on the phase of diagnoser \( D_j \) it performs a different set of operations as described below:

- **\( D_j \) in Phase 1 \((C_{f_j} = 0, T_{f_j} = 0)\):** In this case, diagnoser \( D_i \) only updates its estimate \( E_i \) with respect to \( E_j \), and keeps truncation candidate \( T_i \) unchanged. Also, \( D_i \) does not send out any diagnosis information to other diagnosers. The reason being that is diagnoser \( D_i \) has already sent its truncation candidate \( T_i \) to all diagnosers including \( D_j \), and when \( D_j \) eventually receives that information it will send an updated diagnosis information based on the local truncation candidate \( T_i \). In order to achieve synchrony, the truncation candidate \( T_i \) is not modified in this case.

- **\( D_j \) in Phase 2 \((C_{f_j} = 1, T_{f_j} = 0)\):** First, diagnoser \( D_i \) updates its estimate \( E_i \) with respect to \( E_j \). Then, it computes a new truncation candidate by fusing its own truncation candidate \( T_i \) with the communicated truncation candidate \( T_j \). The index \( j \) is added into \( \Omega_i \). When \( \Omega_i = I \), diagnoser \( D_i \) has fused local truncation candidates from all sites. Then, it performs the actual truncation, resets the value of \( T_i \), i.e., \( T_i := \epsilon \), and also \( C_{f_i} := 0 \) (indicating that the candidate has been consumed), sets \( T_{f_i} := 1 \) (indicating that \( D_i \) has performed the actual truncation), and resets \( \Omega_i := \{ i \} \). Then, diagnoser \( D_i \) broadcasts diagnosis information \( \phi_i = \{ E_i, T_i, C_{f_i}, T_{f_i} \} \) to other diagnosers and transits from phase 2 to phase 3.

- **\( D_j \) in Phase 3 \((C_{f_j} = 0, T_{f_j} = 1)\):** Since diagnoser \( D_j \) is in phase 3, it has performed the actual truncation. It implies that \( T_j \) is the global truncation candidate computed by fusing local truncation candidates from all local sites. Thus, diagnoser \( D_i \) simply sets \( T_i := T_j \), performs the actual truncation and resets \( T_i := \epsilon \). Then, diagnoser \( D_i \) updates its estimate \( E_i \) with respect to \( E_j \), broadcasts \( \phi_i = \{ E_i, T_i = \epsilon, C_{f_i} = 0, T_{f_i} = 1 \} \), resets \( \Omega_i := \{ i \} \) and transits to phase 3.

**Phase 3 – Truncation synchronization \((C_{f_i} = 0, T_{f_i} = 0)\):** In this phase, diagnoser \( D_i \) resets its diagnosis estimate to \( E_i = 0 \).
1): When a diagnoser is in phase 3, it has performed the actual truncation, and is waiting for truncation acknowledgments from other diagnosers. When acknowledgments from all diagnosers are collected, the diagnoser resets flags $C_{f_i} = 0$ and $T_{f_i} = 0$, completes a cycle of truncation, and returns back to phase 1. When diagnoser $D_i$ receives a diagnosis information from diagnoser $D_i$, then depending on the phase of diagnoser $D_i$, it performs a different set of operations as described below:

- **$D_i$ in Phase 1** ($C_{f_i} = 0, T_{f_i} = 0$): This case exists only in the scenario where diagnoser $D_i$ has completed a cycle of truncation and returned to phase 1 while diagnoser $D_i$ is still in phase 3, and at the same time $D_j$ has received a new observation from the plant and sent the updated diagnosis information $e_j$ to diagnoser $D_i$. When $D_i$ updates its own estimate $E_i$ with respect to $e_j$, and then transits to phase 1 directly since $D_j$ is in phase 1 which implies that all diagnosers have completed the actual truncation.

- **$D_i$ in Phase 2** ($C_{f_i} = 1, T_{f_i} = 0$): Since diagnoser $D_i$ has performed the actual truncation ($D_i = 1$ in phase 3) whereas the communicated estimate $E_j$ has not been truncated yet ($D_j = 2$ in phase 2), in order to update $E_i$ with respect to $E_j$, $D_i$ first untruncates $E_i$ with respect to $T_i$, and then fuses it with $E_j$ to get an updated one. After updating $E_i$, diagnoser $D_i$ truncates $E_i$ using the same $T_i$.

- **$D_i$ in Phase 3** ($C_{f_i} = 0, T_{f_i} = 1$): This scenario corresponds to diagnoser $D_i$ receiving a truncation acknowledgment from $D_j$ (since $T_{f_j} = 1$). Then, $D_i$ adds the index $j$ to $O_{f_i}$, and when $O_{f_i}$ contains indices of all diagnosers, i.e., $O_{f_i} = I$, $D_i$ completes a cycle of truncation. Thus, it reverts its diagnosis information $e_i := \{E_i, T_i = \epsilon, C_{f_j} = 0, T_{f_j} = 0\}$ and $O_{f_i} := \{j\}$, and returns to phase 1. In this case, diagnoser $D_i$ does not need to send any further information to other diagnosers (since it already notified other diagnosers about its truncation when transiting from phase 2 to phase 3, and that notification is the information which other diagnosers need from site $i$ for the purpose of synchronizing truncations).

### C. Properties of Synchronized Truncation

The three-phase truncation described above has the following properties:

- At any time, there are at most two phases existing among all distributed diagnosers.
- When a diagnoser receives diagnosis information from another diagnoser in the next lower phase, it only updates its estimate without initiating further communication.
- When a diagnoser is in phase 1, phase 3 is the next lower phase.
- When a diagnoser receives diagnosis information from another diagnoser in the same phase, it adds the index of the sender to the set of indices. Once that set covers all diagnosers, the diagnoser transits from the current phase to the next higher phase (for phase 3, it transits to phase 1).
- When a diagnoser receives diagnosis information from another diagnoser in the next higher phase, it directly transits to the next higher phase without waiting for information from other diagnosers. When a diagnoser is in phase 3, phase 1 is the next higher phase.

### D. A Formal Presentation

For the convenience of presentation, we define the following operations used in the protocol:

- Update $E_i$ with respect to an observation $\lambda_i$:
  \[ E_{i}^{\text{old}} = E_i; \quad E_{i}^{\text{new}} = \{\pi | \pi_1 \in \Pi(G) \mid \pi_1 \in E_{i}^{\text{old}}, \pi_2 \in (\Sigma \cdot Z)^{\ast}, M_{i}(tr(\pi_2)) = \lambda_i\}; \]
- Update $E_i$ with respect to $E_j$:
  \[ E_{i}^{\text{old}} = E_i; \quad E_{i}^{\text{new}} = \{\pi | \pi_1 \in E_{i}^{\text{old}}, \pi_2 \in (\Sigma \cdot Z)^{\ast}\}; \]
- Compute $T_i$ with respect to $E_i$ and $T_j$:
  \[ T_i = \{\pi | \pi \in pr(E_i) \cap pr(E_j) \mid \pi \in \mathbb{N}, E_j \cap \mathbb{N} \}, \]
- Truncate $E_i$ with respect to $T_i$:
  \[ E_{i}^{\text{old}} = E_i; \quad E_{i}^{\text{new}} = \{\pi | \pi \in E_{i}^{\text{old}}, \pi \in \mathbb{N} \}; \]
- Untruncate $E_i$ with respect to $T_i$:
  \[ E_{i}^{\text{old}} = E_i; \quad E_{i}^{\text{new}} = \{\pi | \pi \in E_{i}^{\text{old}}, \pi \in \mathbb{N} \}; \]

Protocol 1 (Distributed Diagnosis Protocol with Synchronized truncation):

Given a plant $G = (X, \Sigma, \alpha, x_0)$ and a specification model $\widehat{G} = (Y, \Sigma, \beta, y_0)$, suppose there are $n$ distributed diagnosers ($I = \{1, \ldots, n\}$) communicating with each other through lossless, order-preserving, and bounded-delay channels. Each diagnoser $D_i$ ($i \in I$) performs the following operations to diagnose failure behaviors in the plant:

1. Construct a refined plant model: $\overline{G} = G||\overline{R} = (Z, \Sigma, \gamma, z_0)$.
2. Perform online diagnosis based on $\overline{G}$: Each diagnoser $D_i$ maintains and updates a diagnosis information $\phi_i = (E_i, T_i, C_{f_i}, T_{f_i})$, and a reachability set $\text{Reach}_i = \{z \in Z \mid \exists \pi \in E_i \in \mathbb{N}, z = \text{last}(\pi)\}$, where $\text{last}(\pi)$ represents the last state in the path $\pi$. When all states in $\text{Reach}_i$ are labeled by “F”, diagnoser $D_i$ reports that a failure has been detected.
3. The diagnosis information $\phi_i$ is updated, communicated, and synchronized at site $i$ as follows:

   **Initial Condition:** $E_0 := \{\pi | \pi \in \Pi(G) \mid \text{first}(\pi) = z_0, M_{i}(\text{first}(\pi)) = \epsilon\}, T_0 := \epsilon, C_{f_0} := 0, T_{f_0} := 0$, and $O_0 := \{i\}$. (Here $\text{first}(\pi) \in Z$ is the first state in the path $\pi$.)

   **Case 1:** Diagnostic $D_i$ receives local observation $\lambda_i \in \Lambda_i$ from the plant:
   - Update $E_i$ with respect to $\lambda_i$.
   - Broadcast $\phi_i = (E_i, T_i, C_{f_i}, T_{f_i})$.

   **Case 2:** Diagnostic $D_j$ receives $\phi_j = (E_j, T_j, C_{f_j}, T_{f_j})$ from diagnoser $D_j$ ($j \neq i$):

   - Phase 1 – Local truncation computation ($C_{f_i} = 0, T_{f_i} = 0$).
   - T.T Diagnoser $D_j$ in phase 1 ($C_{f_j} = 0, T_{f_j} = 0$).
• Update $E_i$ with respect to $E_j$.
• Compute local truncation candidate $T_i$ with respect to $E_i$ and $E_j$.
• $\Omega_i = \Omega_i \cup \{j\}$.
• If $\Omega_i = I$, then $C_{f_i} = 1$, $\Omega_i = \{i\}$, and broadcast $\phi_i$.

1.2 Diagnoser $D_j$ in phase 2 ($C_{f_j} = 1, T_{f_j} = 0$)
• Update $E_i$ with respect to $E_j$.
• $T_i = T_j$.
• Set $C_{f_i} = 1$, $\Omega_i = \{i\}$, and broadcast $\phi_i$.

1.3 Diagnoser $D_j$ in phase 3 ($C_{f_j} = 0, T_{f_j} = 1$)
• Update $E_i$ with respect to $E_j$.

Phase 2 – Global truncation computation ($C_{f_i} = 1, T_{f_i} = 0$)

2.1 Diagnoser $D_j$ in phase 1 ($C_{f_j} = 0, T_{f_j} = 0$)
• Update $E_i$ with respect to $E_j$.

2.2 Diagnoser $D_j$ in phase 2 ($C_{f_j} = 1, T_{f_j} = 0$)
• Update $E_i$ with respect to $E_j$.
• $T_i = T_i \cap T_j$.
• $\Omega_i = \Omega_i \cup \{j\}$.
• If $\Omega_i = I$, then
  i) Truncate $E_i$ with respect to $T_i$.
  ii) Set $C_{f_i} = 0$, $T_{f_i} = 1$, $T_i = \epsilon$, $\Omega_i = \{i\}$, and broadcast $\phi_i$.

2.3 Diagnoser $D_j$ in phase 3 ($C_{f_j} = 0, T_{f_j} = 1$)
• $T_i = T_j$.
• Truncate $E_i$ with respect to $T_i$.
• Update $E_i$ with respect to $E_j$.
• Set $C_{f_i} = 0$, $T_{f_i} = 1$, $T_i = \epsilon$, $\Omega_i = \{i\}$, and broadcast $\phi_i$.

Phase 3 – Truncation synchronization ($C_{f_i} = 0, T_{f_i} = 1$)

3.1 Diagnoser $D_j$ in phase 1 ($C_{f_j} = 0, T_{f_j} = 0$)
• Update $E_i$ with respect to $E_j$.
• Set $T_{f_i} = 0$ and $\Omega_i = \{i\}$.

3.2 Diagnoser $D_j$ in phase 2 ($C_{f_j} = 1, T_{f_j} = 0$)
• Untruncate $E_i$ with respect to $T_i$.
• Update $E_i$ with respect to $E_j$.
• Truncate $E_i$ with respect to $T_i$.

3.3 Diagnoser $D_j$ in phase 3 ($C_{f_j} = 0, T_{f_j} = 1$)
• Update $E_i$ with respect to $E_j$.
• $\Omega_i = \Omega_i \cup \{j\}$.
• If $\Omega_i = I$, then set $T_{f_i} = 0$ and $\Omega_i = \{i\}$.

Remark 1: In the above, we discuss the protocol for a general case with multiple distributed diagnosers. If there are only two diagnosers, the protocol can be simplified when diagnoser $D_i$ in phase 1 receives diagnosis information from another diagnoser $D_j$ in phase 2. As described in Case 2 – Phase 1.2, when diagnoser $D_i$ receives a truncation candidate $T_j$ from diagnoser $D_j$, it assigns $T_i = T_j$. Since there are only two diagnosers, and they have the same truncation candidate, diagnoser $D_i$ can perform truncation immediately using the same operations listed in Case 2 – Phase 2.2, and transit to phase 3 directly.

Comparing Protocol 1 presented above and the iop-based distributed diagnosis protocol introduced in [13], we have the following result that states that Protocol 1 has the same diagnosis capability as the iop-based protocol introduced in [13].

**Theorem 1:** A system is diagnosable under Protocol 1 if and only if it is diagnosable under the iop-based protocol.

**Proof:** A proof sketch is presented here. The proof consists of two parts. First, we claim that if there was no truncation performed in Protocol 1, then it would have the same diagnosis capability as the iop-based protocol. Second, we claim that the three-phase truncation process as given by Protocol 1 is synchronized among all distributed diagnosers. Since the part of the diagnosis estimate history that is common to all diagnosers is not useful in providing a further accurate estimate, the correctness of Theorem 1 follows from the above two assertions.

First note that the second assertion follows by the synchronization steps in Protocol 1, where a diagnoser $D_i$ does not return to its the initial status ($C_{f_i} = T_{f_i} = 0$) without having received the truncation acknowledgments from all other diagnosers. Also the truncation is performed using a common global candidate, which is obtained by first proposing a local truncation candidate and then intersecting them. (All of this is done completely distributively.) It can be concluded that the truncation is synchronized among all distributed diagnosers.

Now to establish the first assertion, we first show that the same diagnosis capability would exist under the following two scenarios (in the first scenario a diagnosis estimate is constructed by an observer itself, whereas in the second one the diagnosis estimate is constructed by a receiver of the observations communicated by an actual observer):

1) Protocol 1 without truncation;
2) Diagnoser $D_i$ sends out its local observation to diagnoser $D_j$ immediately after obtaining that observation, and then diagnoser $D_j$ constructs estimate $E_i$ based on its knowledge of observation mask $M_i$. Then diagnoser $D_j$ performs the same update procedure by fusing $E_j$ and $E_i$ as in Protocol 1.

Since communication delay is same in two situations, it does not matter whether estimate $E_i$ is constructed at the sender side or at the receiver side. Thus, the two scenarios would have the same diagnosis capability.

Finally we claim that the diagnosis capability in the second scenario above would be the same as that in the setting of iop-based protocol. Since online diagnosis is based on the computation of reachability set, it suffices to show that both protocols would give the same reachability set. Now, suppose the plant executes trace $s$. Then any path $\pi$ that is present in the estimate of diagnoser $D_j$ which is receiving event observations forwarded from $D_i$ would have the following properties:

1) $M_j(s) = M_j(tr(\pi))$;
2) $\exists t \in pr(s), |t| \geq (|s| - K_d) \text{ s.t. } M_i(t) \in pr(M_i(tr(\pi)))$,

where diagnoser $D_i$ is the sender, diagnoser $D_j$ is the receiver, and $K_d$ is the communication delay bound. Let
last(\pi) be the last state of path \pi. Then we need to show that last(\pi) is also in the site j reachability set constructed in the iop-based protocol. Conversely, we need to show that if x is in the site j reachability set constructed in the iop-based protocol, then there exists a path \pi satisfying condition 1 and 2 such that x = last(\pi). This can be proved by induction on length of s. i.e., for the base step, let s = \epsilon, and for the induction step, let s = t\sigma, where |t| = k, |s| = k + 1, and \sigma \in \Sigma.

Remark 2: Due to the differences in the way the reachability set Reach_1 is computed, Protocol 1 presented in this paper has a different complexity than the iop-based protocol (even though they have the same diagnosis capability). In the iop-based distributed diagnosis, the reachability set Reach_1 is computed using an extended local diagnoser at site-i, whose complexity is polynomial in the number of states and events of the plant as well as the specification, and exponential in the number of distributed diagnosers. In contrast, computation of estimate E_i does not depend on the number of diagnosers (since only two sets of diagnosis information is fused at any given time), but since E_i needs to maintain a list of all possible paths the system has executed, its complexity is exponential in the number of states and events of the plant as well as the specification. Clearly, there is a complexity trade-off. The new protocol is preferable when there are large number of distributed sites, whereas the iop-based protocol is preferable when the plant/specification sizes are large.

IV. ILLUSTRATING EXAMPLE

The following example illustrates how to use Protocol 1 to perform distributed diagnosis. The same example was used in [13] to illustrate the iop based diagnosis as well.

Example 1: Figure 1(a) and Figure 1(b) show a plant model G and a specification model R, respectively, with L(G) = pr(abab* + baac*) and L(R) = pr(abac*). The refined plant model \overline{G} = G||R is shown in Figure 1(c). Suppose there are two local sites (I = \{1, 2\}), with observation masks defined as follows:

- M_1(a) = a, M_1(b) = M_1(c) = \epsilon, and
- M_2(b) = b, M_2(a) = M_2(c) = \epsilon.

For the convenience of presentation, the path contained in estimate E_i (i \in I) is represented in the format of “(state)event(state)···(state)”. Suppose a faulty trace “baac” is executed in the plant. We consider the following online diagnosis procedure:

1) Initialization (#1: phase-1; #2: phase-1):
- E_1 = {(00), (00)b(4F)}, T_1 = \epsilon, C_f_1 = 0, T_f_1 = 0,
- E_2 = {(00), (00)a(11), (00)a(11)a(22)}, T_2 = \epsilon, C_f_2 = 0, T_f_2 = 0;
- Reach_1 = {(00), (4F)}, Reach_2 = {(00), (11), (22)}.

2) #2 observes “\&b”:
- Apply “Case 1” of Protocol 1:
  - E_2 = {(00)a(11)a(22)b(33)(c(33))*, (00)b(4F), (00)b(4F)a(5F), (00)b(4F)a(5F)a(6F)(c(6F))}, T_2 = \epsilon, C_f_2 = 0, T_f_2 = 0;
  - #2 sends \phi_2 = \{E_2, T_2, C_f_2, T_f_2\} to #1;
- Reach_2 = \{(33), (4F), (5F), (6F)\}.

3) #1 observes “\&a”:
- Apply “Case 1” of Protocol 1:
  - E_1 = {(00)a(11), (00)b(4F)a(5F)}, T_1 = \epsilon, C_f_1 = 0, T_f_1 = 0;
  - #1 sends \phi_1 = \{E_1, T_1, C_f_1, T_f_1\} to #2;
- Reach_1 = \{(11), (5F)\}.

4) #1 receives \phi_2 (in Step 2) from #2: (#1: phase-1 → phase-2)
- Apply “Case 2 – Phase 1.1” of Protocol 1:
  - E_1 = {(00)b(4F)a(5F)}, T_1 = {(00)b(4F)}, C_f_1 = 1, T_f_1 = 0;
  - #1 sends \phi_1 = \{E_1, T_1, C_f_1, T_f_1\} to #2;
- Reach_1 = \{(5F)\}.

At this point, since Reach_1 only contains a failure state (5F), a failure is detected at site 1. From the diagnosis point of view, we do not need to pursue further diagnosis steps. However, to illustrate how the protocol works, especially how to perform synchronized truncation, we continue to explain the following procedures.

5) #2 receives \phi_1 (in Step 3) from #1: (#2: phase-1 → phase-2)
- Apply “Case 2 – Phase 1.1” of Protocol 1:
  - E_2 = {(00)a(11)a(22)b(33)(c(33))*, (00)b(4F)a(5F), (00)b(4F)a(5F)a(6F)(c(6F))}, T_2 = {(00)a(11), (00)b(4F)a(5F)}, C_f_2 = 1, T_f_2 = 0;
  - #2 sends \phi_2 = \{E_2, T_2, C_f_2, T_f_2\} to #1;
- Reach_2 = \{(33), (4F), (5F)\}.

6) #1 observes “\&a”:
- Apply “Case 1” of Protocol 1:
  - E_1 = {(00)b(4F)a(5F)a(6F)(c(6F))}, T_1 = {(00)b(4F)}, C_f_1 = 1, T_f_1 = 0;
  - #1 sends \phi_1 = \{E_1, T_1, C_f_1, T_f_1\} to #2;
- Reach_1 = \{(6F)\}.

7) #2 receives \phi_1 (in Step 4) from #1: (#2: phase-2 → phase-3)
- Apply “Case 2 – Phase 2.2” of Protocol 1:
  - Perform truncation: T_2 = T_1 \cap T_2 = \{(00)b(4F)\}, E_2 = {(00)a(11)a(22)b(33)(c(33))*, (4F)a(5F), (4F)a(5F)a(6F)(c(6F))} ;
  - T_2 = \epsilon, C_f_2 = 0, T_f_2 = 1;
  - sends \phi_2 = \{E_2, T_2, C_f_2, T_f_2\} to #1;
- Reach_2 = \{(33), (5F), (6F)\}.

8) #1 receives \phi_2 (in Step 5) from #2: (#1: phase-2 → phase-3)
- Apply “Case 2 – Phase 2.2” of Protocol 1:
  - Perform truncation: T_1 = T_1 \cap T_2 = \{(00)b(4F)\}, E_1 = \{(4F)a(5F)a(6F)(c(6F))\};
  - T_2 = \epsilon, C_f_1 = 0, T_f_1 = 1;
  - sends \phi_1 = \{E_1, T_1, C_f_1, T_f_1\} to #2;
9) \#2 receives $\varphi_1$ (in Step 6) from \#1: (#2: phase-3 → phase-1)
   - Apply “Case 2 – Phase 3.2” of Protocol 1:
     - Untruncate $E_2$ with $T_2$:
       $$E_2 = \{(00)a(11)a(22)b(33)(c(33))^*,
       (00)b(4F)a(5F), (00)b(4F)a(5F)a(6F)(c(6F))^*)\}$$
     - Update $E_2$:
       $$E_2 = \{(00)b(4F)a(5F)a(6F)(c(6F))^*)\}$$
     - truncate $E_2$: $E_2 = \{(4F)a(5F)a(6F)(c(6F))^*)\}$
   - $Reach_2 = \{(6F)\}$.

10) \#1 receives $\varphi_2$ (in Step 7) from \#2: (#1: phase-3 → phase-1)
    - Apply “Case 2 – Phase 3.3” of Protocol 1:
      - $E_1 = \{(4F)a(5F)a(6F)(c(6F))^*\}$, $T_1 = \epsilon, C_{f_1} = 0, T_{f_1} = 0$
    - $Reach_1 = \{(6F)\}$.

11) \#2 receives $\varphi_1$ (in Step 8) from \#1: (#2: phase-3 → phase-1)
    - Apply “Case 2 – Phase 3.3” of Protocol 1:
      - $E_2 = \{(4F)a(5F)a(6F)(c(6F))^*\}$, $T_2 = \epsilon, C_{f_2} = 0, T_{f_2} = 0$
    - $Reach_2 = \{(6F)\}$.

After Step 10 and 11, both sides complete a cycle of truncation, and return to the initial status ($C_{f_j} = T_{f_j} = 0$). I.e., truncated diagnosis estimates at both sides are synchronized.

Remark 3: In the above example, we consider a scenario where communication delay is less than or equals to 1. I.e., between the transmission and reception of a message, there is at most one event executed in the plant. However, if the communication delay is increased to 2, we can verify that the system does not remain diagnosable. The same result also holds for the iop protocol, as discussed in [13].

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed a new distributed diagnosis protocol to address the problem of failure diagnosis in large distributed systems. In our previous work involving iop-protocol based distributed diagnosis [13], the complexity of constructing local diagnosers is exponential in the number of local sites, which limits its applicability in large distributed systems with many local sites (such as sensor networks).

In the iop-based protocol, all communication models, which include information about observation masks and communication delays, are fused together to extend original system models. The extended models have exponential complexity in the number of local sites and as a result, the local diagnosers constructed using them also have the same exponential complexity in the number of local sites. To overcome this complexity problem, we presented a new distributed diagnosis protocol that avoids extending any of the system models. Instead, each diagnoser transmits diagnosis information and fuses such information with its own when it receives such information from other diagnosers. In order to reduce the space requirement and communication burden, redundant information is truncated synchronously and distributively among all diagnosers. This is a key feature of the proposed distributed diagnosis protocol.

We argued that the new protocol has the same diagnosis capability as the iop-based protocol. Since the computation of reachability set at one local site is not dependent on the number of local diagnosers, the new protocol has a linear complexity in the total number of local sites. An example is provided to illustrate the distributed diagnosis procedures under the new protocol.

As discussed above, while the new protocol reduces the complexity in the number of local sites, it has an exponential complexity in the size of plant and specification. In contrast, the iop-based diagnosis protocol has a polynomial complexity in the size of the plant and the specification. Therefore, the choice of a distributed diagnosis protocol will depend on a specific application. Also, a trade-off between complexity and communication exists in the new protocol, where more communication is invoked to save on space complexity. Discussion about this trade-off also appeared in the literature [1], [4]. Future work on these topics is important to distributed diagnosis/control, and other distributed decision-making problems.
REFERENCES


