INTERNATIONAL JOURNAL OF CIRCUITS, SYSTEMS AND SIGNAL PROCESSING
DOI: 10.46300/9106.2020.14.21 Volume 14, 2020

Discovery of Incomplete Diagnostic Model based on Learning

Wang Xiaoyu, Li Chuang and Ye Liang

***Abstract*—**The model-based diagnosis uses the common reasoning of offline model and online observation to obtain whether and why faults occur. However, the diagnosis is based on the premise of complete model. Once there are unknown behaviors in the diagnosis process, the diagnosis results will not be obtained. In this paper, a method of incomplete model discovery based on online diagnosis process is proposed: In the online diagnosis process, the data of the complete model are learned and the model is trained and adjusted. When the incomplete behavior is found, the nature of the incomplete behavior is determined according to the historical diagnostic data and online observation data, and the corresponding transition/state/event is generated and added to the model to further obtain the definite diagnosis results.

***Keywords*—**mode based diagnosis; incomplete model; model discovery

I. INTRODUCTION

Model-based diagnosis method is a cross branch of both artificial intelligence and control field[1-3]. Through the common reasoning of model and observation, the conclusion of whether the system is faulty or not is obtained. Model-based diagnosis is initially a static method, which can obtain the diagnoses when the system is down[4-5]. As the modern system is becoming larger and larger, the behavior is more and more complex, and the cost of system downtime is higher and higher, a dynamic model-based diagnosis method is produced. In the process of system operation, system faults are judged by off-line models and on-line observations. Dynamic diagnosis method can not only get faults, but also get the way of system gets bad[6-7]. However, the above diagnosis process is very dependent on the model. Once the model can not fully correspond to the actual behavior, it will produce blind expansion, or even can not get the diagnoses. At the same time, the premise of traditional model-based diagnosis is that the model established by abstraction or approximation can completely correspond to the actual behavior. The advantage of off-line complete model is that it can improve the diagnostic efficiency by pre-compiling the model, while the disadvantage is that if there are incomplete behaviors, the diagnostic results cannot be obtained.

Incomplete model refers to undefined behavior in the process of mapping from real world to logical model. In the diagnosis based on complete model, undefined behavior will not be able to get results[8]. And in another case, the failure cannot be completely eliminated[9]. In the diagnosis of incomplete model, the basic complete diagnoses is obtained by synchronization through the existing complete model and online observations, and the incomplete diagnosis problem is defined by temporal and causality[10]. Although this diagnosis breaks the premise of the traditional diagnosis method which assumes that the model is complete, it has some drawbacks: When the event is incomplete, it cannot accurately distinguish the location of incomplete events, and when defining the nature of incomplete events, it is not accurate to rely only on the judgment of the trajectory before and after.

In order to solve this kind of problem, the concept of incomplete model is proposed[8,11] to solve the situation that the expected behavior is different from the model. On this basis, more researchers pay attention to this problem and propose a series of methods[12-14]. Context of model is used to deduce the faults[15], or knowledge is compiled to dianogsis[16]. In this paper, decision tree is used to infer online incomplete events, further characterize the attributes of events, and more accurately locate the location and context of events in the model. At the same time, it defines a corresponding diagnostic model that can use learning method, and carries out complete or incomplete online diagnoses simultaneously.

II. MODEL

A. A Framework of Diagnosis

Complete system diagnostic methods for discrete event are usually based on automata models.

Definition 1 (complete model)[17]: The complete model is represented by a five tuple automata $G = (Q, E, T, I, F)$

And Q is a finite state set; E is a finite event set; T is a finite transition set: $T \subseteq Q \times E \times Q$, $I \subseteq Q$ is an initial state set; $F \subseteq Q$ is a final state set. According to the nature, event set can be divided into three subsets, observable event set E_o ,normal non-observable event set E_n ,and fault non-observable event set E_f , and satisfy $E = E_o \cup E_n \cup E_f, E_o \cap E_n = \emptyset, E_o \cap E_f = \emptyset, E_n \cap E_f = \emptyset$.

In a complete model, the process of system behavior is defined as a path.

Definition 2 (path)[18]: Path s is the transition sequence of states triggered by events $s =< q_0, e_0, q_1 \ldots q_n >$, and $\forall t, t = q_i \times$

This work was supported by Science and Technology Project of the 13th Five-Year Plan of Jilin Provincial Department of Education (JJKH20180763KJ); Science and Technology Development Plan Project of Siping (2017093).

Wang Xiaoyu is with the College of Computer, Jilin Normal University, Siping, Jilin, China (e-mail: wxyjldx@163.com)

Li Chuang is with the College of Computer, Jilin Normal University, Siping, Jilin, China (e-mail: lichuang12@mails.jlu.edu.cn)

Ye Liang is with the College of Computer, Jilin Normal University, Siping, Jilin, China and the Siping Highway Administration Office, Siping, Jilin, China (e-mail:caocaovb@163.com)

$e_{i+1} \times q_{i+1}, t \in T$. $S(G)$ is defined as the set of all possible paths on model *G*.

State is triggered by events, and system operation is driven by event sequence. The sequence of events is called the language of the system. The sequence of events on the path constitutes the language of the system.

Definition 3 (language)[1]: The language on system *G* is a sequence of possible events $L(G) =< e_i^* >, e_i^* \in 2^E$.

When the actual system is running, only partial events are visible, and the sequence of all events can be deduced from the visible event sequence, and the corresponding transformation mapping between languages can be defined:

Definition 4 (observable mapping)[1]: The observable mapping is defined between the event set and observable event set.

$$P_o(e) = \begin{cases} e & e \in E_o \\ \varepsilon & e \notin E_o \end{cases}$$

Furthermore, observable mapping of language is defined: $P_o(l) = P_o(l'e_i) = P_o(l')P_o(e_i)$ The same mapping can be defined between the event set and the failure event set. Conversely, the inverse mapping of the observable mapping is the set of languages.

Definition 5 (observable inverse mapping)[1]: $P_o^{-1}(l') = \{l | P_o(l) = l'\}$

The observable event sequence is obtained during the online process, denoted by Obs. Obs is divided into several subsequences according to time: $Obs =< w_0 w_1 \dots w_n >, w_i =< e_i >, e_i \in E_o$. Diagnosis is a set of trajectories synchronized from online observations and offline models.

Definition 6 (diagnosis) :a diagnosis $\Delta(G, Obs)$ is defined as: $\Delta(G, Obs) = \{s | s \in S(G), P_o(s) = Obs\}$

If the system can be diagnosed, the path is unique when $|Obs|$ is greater than a finite integer value. A complete diagnostic model is defined above, and a compatible trajectory is found in the model for diagnosis based on online observations. Figure 1 is an example of a diagnostic automata model.

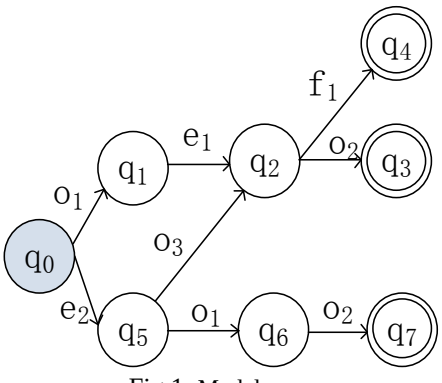

Fig.1. Model

In Fig.1, q_0 is an initial state, q_3, q_4, q_7 are final states, the e_i, o_i, f_i are events, and a pair of circles connected by an edge is a transition. There are four paths in Fig.1: from q_0 to the final states, and languages as $o_1 e_1 o_2$ etc. The observable map of language $o_1 e_1 o_2$ is $o_1 o_2$, and observable inverse mapping of $o_1 o_2$ are $o_1 e_1 o_2$ and $e_2 o_1 o_2$. If we get observation o_1 online, the diagnosis would be the path from q_0 to q_4 with fault f_1.

However, when the premise of complete model is canceled, there may be events or states that do not exist in the model, which can not get the diagnosis results in the complete model. Therefore, the diagnosis model is redefined, and the partial methods of machine learning are used to discover the model online and define the incomplete events accurately.

Definition 7 (incomplete model): The incomplete model is represented by a five-tuple automata $G^I = (Q^I, E^I, T^I, I^I, F^I)$.

Similar to the complete model, Q^I is a finite set of incomplete states with state vectors, $\mathrm{q^i} \in \mathrm{Q^I}, \mathrm{q^i} = \mathrm{q} \times (\mathrm{a_1}, \mathrm{a_2} \dots \mathrm{a_m})$; E^I is a finite set of incomplete events with state triggered vector function, $e^i \in E^I, e^i = e \times (f(a_1), f(a_2) \dots f(a_m))$; T^I is a finite transition set, $T^I \subseteq Q^I \times E^I \times Q^I$, and if $\exists q_x^i, q_z^i \in Q^I, \exists e_y^i \in E^I, q_x^i = q \times (a_{x1}, a_{x2} \dots a_{xm}), q_z^i = q \times (a_{z1}, a_{z2} \dots a_{zm}), e_y^i = e \times (f(a_{y1}), f(a_{y2}) .. f(a_{ym}))$. For $\forall t^i \in T^I$,if $t^i = q_x^i \times e_y^i \times q_z^i$, there is $f(a_{x1}) = a_{z1}, f(a_{x2}) = a_{z2} \dots f(a_{xm}) = a_{zm}$. I^I and F^I are initial state and termination state, respectively.

The states will be labeled vectors and the events will be labeled functions. If we use this model to represent a three-tanks, the state q_0 describes a high level of the first tank, the vector $a_1 = 0.5, a_2 = 3, a_3 = 24$ may be the depth of water is 0.5m, the pressure is 3kPa and the temperature is 24℃, and the event o_1 indicates "open the outlet of the first tank" which leads to lower the level, and has the function $f(a_1) = a_1 - 0.01 \times time, f(a_2) = a_2 - 0.05 \times time, f(a_3) = a_3$

In the process of modeling, the historical data can be used to adjust $f(a_i)$ to get the relationship of the state, event and transition. The initial incomplete model is obtained from the complete model. Different types of states and events can be distinguished by vectors and vector functions. The distance between states is defined to distinguish different types of states.

Definition 8 (trigger): The trigger of state q^i is defined as a finite set of events $C(q^i)$, $C(q^i) = \{e^i | q^i \times e^i \times q_y^i \in T^I\}$. That is, the set of events that can trigger this state, for instance, $C(q_0) = \{o_1, e_2\}$

Definition 9 (state distance): The distance between states is defined as the weighted average of the difference between two state labels.

$$D(q_x^i, q_y^i) = \left| \sum_{i=1}^{m} \mu_i (a_{xi} - a_{yi}) \right|$$

This is a simplified distance method. Without considering the relationship between state labels, all the weights $\mu_i = 1$.

The online diagnosis firstly considers that the model is complete, and the state is synchronized with the offline complete model. When $\Delta(G, Obs) = \emptyset$, if the current observation length is larger than the fixed value n (obtained from the diagnosability decision), it is considered that the incomplete state or event is generated currently. Then the complete hypothesis of the model is canceled and the diagnosis of the incomplete model is carried out.

Definition 10 (critical state): Observation $Obs =< w_0 w_1 \dots w_n >$. If $\Delta(G, < w_0 w_1 \dots w_{n-1} >) = s, s =< q_0 e_0 \dots q >, \Delta(G, < w_0 w_1 \dots w_n >) = \emptyset$, q is called the critical state of the current diagnosis, which is recorded as ${}^V q$, , and w_n is defined as the current critical window.

For instance, if the windows are $w_0 = o_1, w_1 = o_3$, the first diagnosis are paths ends at q_2, q_4 and q_6, but when a new window arrives, all the paths cannot explain what happens, an incomplete behavior appears, and leads to an accidental observation, the critical states are q_2, q_4 and q_6.

In the critical state set, the nature of incomplete events is judged, and incomplete events and states are classified according to states and other events. Next, the method is described in detail.

III. DISCOVERY OF INCOMPLETE BEHAVIOR

A. Incomplete cases

After the off-line model is built, the system enters the online diagnosis process. Considering that the model may be incomplete, the complete model is replaced by the incomplete model in the online process, and the state vector is assigned online to adjust the vector function of the event. However, in order to be efficient, these values and functions are not used to diagnose. Until it is considered that incomplete behavior is generated, the behavior discovery in incomplete model begins.

In the offline process, the value of state and the function of event are initialized. In the online process, according to the observation and diagnoses, the value of state is determined and the parameters of function are adjusted. When the model is basically complete, the parameters and values will not change much.

The incremental method is used for diagnosis. Observations are coming one after another, and compatible paths are searched in each local model. If the compatible path can not be found at present, the incremental backtracking is considered first. When the diagnostic solution can't be found by tracing back to the initial state, the current model is considered to be incomplete and the process of incomplete behavior discovery and diagnosis is entered. According to the characteristics of incomplete behavior of the model, it is discussed in three cases.

1. The event is complete, the state is complete and the transition is incomplete, that is, the current event and state are in the complete model, but the correlation(transition) between event states does not belong to the complete model.
2. The event is complete and the state is incomplete. The current event belongs to the complete model, and the result after transition does not belong to the complete model.
3. The event is incomplete, and the state is incomplete. The current event and state do not belong to the complete model.

B. Incomplete Transition

In the current situation, the relationship between events and states is incomplete. It is necessary to determine the transition relationship between events and states, that is, a transition t needs to be found $t = q_i \times e_i \times q_k$, among which $t \notin T, q_i, q_k \in Q, e_i \in E$.

First, an event should be found and redefined. This event triggers incomplete behavior. The current critical state is ${}^V q_m$, the current observation $Obs =< w_1, w_2 .. w_{n-1} >$, the critical window is w_n, and the incomplete model is G^I.

The possible arrival state set Q_1 consists of the trigger states of the observed events in the critical window, that is $\forall q \in Q_1, \exists o \in E_o \cap w_n, \exists t \in T^I, t = q \times o \times q'$. The set E_1 of possible events consists of events that trigger critical states and events in observation windows, that is $\forall e \in E_1, (\exists t \in T^I, t = {}^V q \times e \times q) \vee (e \in w_n))$.

For any state in $Q_1 \cup \{{}^V q\}$, any event in E_1 is used to trigger and a state set Q_2 with values is obtained. The state distance d of any two states in Q_2 and Q^I is calculated, $d = D(q_i, q_j), q_i \in Q_2, q_j \in Q^I$. If $\exists d < \varepsilon$, the transition $q_i \times e \times q_j$ is generated, among which, e is the event that moves the state q_i to q_j. The values of q_i, q_j are not changed. The parameter of the event function is adjusted to make it fit the current transition.

The incomplete transition is often caused by the neglecting in modeling, the events and states have been considered in the model, but not been connected. For instance, the three-tanks include two states "high level" and "medium level", one event "turn on the outlet", but not a transition among them. We compute the distance between the two states, and find similar other state distances, add the event with right function to connect the states.

C. Incomplete State

If there is no state in Q_2 that is close enough to the state set of the complete model that can't be found in the complete model, and $w_n \neq \emptyset, \forall o \in w_n, o \in E$, it is considered that there exists a complete event, which makes the state deviate. At this time, the state is generated, assigned and transferred.

According to the state distance in the class, Q_1 is classified and the state set $C(Q_1)$ of the classification is generated. ${}^V q$ is considered as the event e of the condition state to generate a set E_2, e is defined as $\exists s \in S(G), s =< {}^V q \ldots . e \ldots >$. The event in the event set E_2 to trigger ${}^V q$ to obtain the state set Q_3. For $\forall q \in Q_3$, classification is carried out in $C(Q_1)$, and incomplete states are generated in the classification results.

If $\exists q \in Q_3$ and q belongs to arbitrary classification in $C(Q_1)$, it is believed that there is a complete event that leads the system to an incomplete state. If q can't belong to any classification in $C(Q_1)$, it is considered that there are incomplete events and incomplete states, which will be discussed in the next section.

If there are only incomplete states, the establishment of incomplete states and corresponding transitions are discussed. For $\forall q \in Q_3$, the distribution of q in $C(Q_1)$ is calculated, and q is in different classifications. The classification $C_i(Q_1)$ with the most projections is selected. The center point q_c of $\{q\} \cup C_i(Q_1)$ is calculated and the state q_k closest to q_c in $\{q\}$ is selected. If ${}^V q \times e_m \times q_k$, e_m is the complete event, and q_c is the generated incomplete state. The transition ${}^V q \times e_m \times q_c$ is generated and $Q^I = Q^I \cup \{q_c\}, T^I = T^I \cup \{{}^V q \times e_m \times q_c\}$. The function of event e_m is adjusted step by step according to the values obtained many times during diagnosis.

Continue the three-tank instance, if the state "medium level" is not included in model, the trigger of critical state "high level" are found, and calculate the right event with function. By

the function and critical state, the vector will be calculated, then a state and transition will be built in model.

D. Incomplete Events and States

When a complete event triggers a complete state and the result is not close to the existing state, the incomplete event and its corresponding state are considered. In the critical state, undefined event triggers state and produces unknown result state. New events and new states are generated according to critical states and observations.

If the incomplete event is observable, it appears in the current observation window, starts with a state, and moves to an unknown state. In this case, according to the position of observable incomplete events in the observation window, the critical state is updated, the classification is carried out in the subsequent state, and the generated event function is trained and the incomplete event is generated. Then, the incomplete state is generated according to the critical state and event.

If the incomplete event is not observable, according to the offset of the current observation window, starting with the critical state, all subsequent states that may be triggered by the current observation window are classified, and then the generated events are trained and then the incomplete states are generated according to the critical states and events.

The incomplete events and states are usually accident in system. In the instance of three-tanks, if one tank is punctured a hole suddenly, a completely unexpected situation happens. The critical state "high level" will nominate a state "lower level" by learning from "medium level" and build an event "open a bigger outlet" to describe the rapidly lowered level by learning from event "open an outlet".

There is a premise here: there is only one incomplete event in the current system. The incomplete event and state generated are given priority in the subsequent increment of the current diagnosis. When the next initial diagnosis begins, the newly generated incomplete model is the same as the original model, and the complete or incomplete model is trained and adjusted online.

IV. ALGORITHMS

A. Find an incomplete model

The proposed incomplete diagnosis is a whole process: In the offline process, a model compatible with incomplete behavior is established. In the online monitoring process, the observation is obtained and while diagnosing, the incomplete model established offline is supplemented. If the incomplete behavior beyond expectation is not found, the diagnosis path is obtained according to the complete model diagnosis. Otherwise, according to the complete model generated in the online process, the incomplete behavior is defined and the diagnosis on the incomplete model is obtained. The overall algorithm is as follows:

Algorithm 1: Diagnostic process of incomplete model
Input: Incomplete model $G^I = (Q^I, E^I, T^I, I^I, F^I)$, Observation sequence $Obs =< w_1, w_2 .. w_n \dots >$
Output: Diagnosis Δ, or modified incomplete model $G^I = (Q^I, E^I, T^I, I^I, F^I)$

1. Initialization: diagnosis $\Delta_i = \emptyset$, the current window $i = 0$
2. While(Complete)
3. $\{ \Delta_i = Dia(G^I, w_n)$ //Incremental diagnosis
4. $Set(G^I, \Delta_i)$
//The model is trained according to the diagnostic results
5. Return$(\sum_{t=0}^{i} \Delta_t)\}$
6. ${}^V q = End(\Delta_t),\ t = max(\Delta_i \neq \emptyset)$
//The critical state is the termination state of the current diagnosis result.
7. if$(\exists t_i \notin T^I)$
8. $G^I = FindT({}^V q, w_i, G^I)$
//If only the transition is incomplete, a transition is generated and the model is updated
9. if$(\exists q_i \notin Q^I \wedge \nexists e \notin E^I)$
10. $G^I = FindQ({}^V q, w_i, G^I)$
//If the event is complete and only the state is incomplete, the state is generated
11. else
12. $G^I = FindE({}^V q, w_i, G^I)$
//The state and event are generated

Diagnosis is initially operated under complete assumptions. When it is found that the diagnosis results can't be obtained, the incomplete model discovery is considered. The incomplete cases are judged and dealt with separately. Next, the algorithms for finding incomplete models are given.

Algorithm 2: $FindT$
Input: The current critical state ${}^V q$, the current observation Obs $=< w_0 \dots w_i >$, and the current incomplete model G^I
Output: Updated incomplete model G^I or the result set that can't generate incomplete transition

1. Initialization: $Q_1 = \emptyset, E_1 = \emptyset$
2. $\forall o \in w_i, \forall t \in T^I, t = q \times o \times q'$
3. $Q_1 = Q_1 \cup \{q\}, E_1 = E_1 \cup \{o\}$
4. ${}^V q, \forall t \in T^I, t = {}^V q \times e \times q'$
5. $Q_1 = Q_1 \cup \{{}^V q\}, E_1 = E_1 \cup \{e\}$
6. $\forall q \in Q_1, \forall e \in E_1, q = q \times (a_{q1} \dots a_{qm})$
7. $e = e \times (f_e(a_1) \dots f_e(a_m))$
8. $q' = q' \times (a_1' \dots a_i' \dots a_m'), a_i' = f_e(a_{qi})$
9. $Q_2 = Q_2 \cup \{q'\}$
10. $\forall q \in Q_2, \forall q' \in \mathrm{Q^I}$
11. if$(D(q, q') < \varepsilon)$
12. $t^i = q_1 \times e_1 \times q', q_1 \in Q_1, e_1 \in E_1, q_1 \times e_1 = q$
13. $T^I = T^I \cup \{t^i\}$
14. Else
15. Return Q_1

Incomplete transition is an incomplete condition in which only the relationship between events and states is missing. Therefore, in the current critical state and the state related to observation, it can be judged whether it can be triggered by the current event. If the final state is close to the existing model state, it will be considered as an incomplete transition result. At this time, the state and event of this state are obtained in reverse, the transition is established, and the incomplete model is updated.

When the algorithm $FindT$ can't find the incomplete transition, we can further judge whether the incomplete state can be found, so that the complete event can trigger the

complete state and get the incomplete state. Algorithm 3 is a method of generating incomplete states.

Algorithm 3: $FindQ$

Input: Critical state ${}^{V}q$, the current observation $Obs =< w_0 \dots w_i >$, the current incomplete model G^I, and the set Q_1 returned by $FindT$

Output: Updated incomplete model G^I, or the result set that can't generate the incomplete state

1. Initialization: $E_2 = \emptyset, Q_3 = \emptyset, C = \emptyset$
2. ${}^{V}q, \forall t, t = {}^{V}q \times e \times q', E_2 = E_2 \cup \{e\}, Q_3 = Q_3 \cup \{q'\}$
3. while$(C_i(Q_1) \neq C_{i+1}(Q_1))$
4. MS(Q_1) //Compute classification by Meanshift
5. $\forall q \in Q_3$
6. if$\big(D\big(q, Center(C_i)\big) < r\big)$

//If the distance from the state to a central point is less than the radius of classification

7. $Record(d(q, q'), q, C_i)$

//Record state, distance and classification

8. $C = Max(C_i)$

//Choose the classification with the most states

9. $q = Min(Record)$

//Calculate the state with the smallest distance

10. $Q^I = Q^I \cup \{q\}, T^I = T^I \cup \{{}^{V}q \times e \times q\}$

//e is the event that triggers ^{V}q to transition to q

11. else
12. Return Q_3

When there is no intersection between the critical state and the observation state, the state and event are generated and they are incomplete at the same time. Preliminary events and states are generated according to observation windows and incomplete models with historical data, and incomplete models are updated. Furthermore, the function parameters of events are trained according to the online running data, and the state attributes are adjusted.

Algorithm 4: $FindE$

Input: Critical sate ${}^{V}q$, the current observation Obs, the current incomplete model G^I, and local results of the aforementioned algorithms Q_1, Q_2, Q_3

Output: Updated incomplete model G^I, or unrecognizable incomplete results

1. Initialization:
2. if$(\exists e \in w_i, e \notin E_o)$
3. $\forall q, \exists s \in S(G^I), s = {}^{V}q \dots q..$
4. $Q_4 = \{q\} \cup Q_4, Q_1 = Q_1 \backslash \{{}^{V}q\}$
5. $Q_5 = Q_4 \cap Q_1$
6. $\forall q \in Q_5, \exists t = q \times e \times q'$
7. $Q_6 = \{q'\} \cup Q_6$
8. $MS(Q_6)$

//Compute classification by Meanshift method

9. $q^i = Center(Max(C_i))$

//Choose the center point with the most classifications as the temporary state

10. $e^i \times= Train(q, q^i)$ //Train event
11. $q' = F(q, e^i)$

//Generate state according to state and event

12. $t = q \times e^i \times q'$ // Generate transition
13. if$(e \notin w_i)$
14. $Q_7 = Q_4 \cup Q_1$
15. $MS(Q_7)$
16. $q^i = Center(Max(C_i))$

//Choose the center point with the most classifications as the temporary state

17. $e^i \times= Train(q, q^i)$ //Train event
18. $q' = F(q, e^i)$

//Generate state based on state and event

19. $t = q \times e^i \times q'$ // Generate transition
20. $T^I = \{t\} \cup T^I, E^i = \{e^i\} \cup E^i, Q^I = \{q'\} \cup Q^I$

This algorithm processes undefined behaviors. When a diagnosis cannot be found in model, the closest state is calculated from the cluster. If the closest state is near to a state of model, the event and transition will be generated based on this neighborhood; otherwise, find a most possible state then train the event and transition. Actually, if an undefined behavior happens, we guess whether it like some behaviors in model first, otherwise, we follow the average.

After the incomplete model is obtained, the diagnosis will start incremental diagnosis from the critical state, and the incomplete model currently obtained is selected as a candidate diagnosis path. The diagnosis result will return a path with incomplete events and states. The current incomplete model can be used as a complete model and parameters can be adjusted online in the next online diagnosis from the initial state. The corresponding experimental results will be given next.

V. EXPERIMENTAL RESULTS AND ANALYSIS

The experimental machine configuration is as follows: processor: Inter i7-6700, 3.4GHz, memory 8GB RAM, Window7 home version 64-bit operating system

The simulation model is used to test the algorithm, and the automata models of different scales are established. Some transitions, states or events are deleted from the model to simulate incomplete situations. In the simulation operation process, a fixed size observation window with time mark is generated periodically, which enables the diagnostic system to make incremental diagnosis on the model according to the observation window. Deleted transitions, states and events are added randomly, and different observations will be obtained by the system.

In the experiment, the test will be carried out in the following two directions:

Under different model scales, the efficiency of discovering incomplete behaviors and adding to models is tested. The efficiency of using algorithms *FindT FindQ*, and *FindE* is tested.

Using complete model defined in definition 1, the scale of model is computed by states. When online windows are fixed, observations sequences in one trajectory are given by order, the model diagnoses online. When model is complete, the efficiency of incremental diagnosis is tested. According to different complete degree, a part of states, events, transitions are removed, the remainder of model are diagnosed in the same way, the efficiency of both complete and incomplete are tested. The algorithms *FindT FindQ*, and *FindE* are tested. Incomplete events/states/transitions each account for 1/3.

TABLE I contrasts time efficiency of diagnosing complete model, diagnosing incomplete model with complete hypothesis and diagnosing incomplete model without complete hypothesis. *a* in **TABLE I** means time of diagnosing complete model, table cell with complete degree means time of diagnosing incomplete model by using algorithms $FindT$、$FindQ$ and $FindE$, The units are expressed in milliseconds

TABLE I. Efficiency of diagnoses

scale	Complete degree	99%	98%	95%	90%	80%	*a*
20		1.1	1.2	1.23	1.24	1.3	1
100		47.7	47.9	48	48.3	48.7	47
500		219	224.5	234.2	250.3	276	202
1000		489	545.	623	773	1070	423
5000		2439	3204.2	5092	8927	16338	783
10000		7027	10164	18581	36350	82549	812

In the same model, the difference between the transition/state/event discovered and that of the actual transition/state/event is tested, so as to test the accuracy of discovering incomplete behavior.

In complete model, attributes and functions are generated. According to different complete degree, states, events or transitions are removed, and then the incomplete model is got. The differences between actual value and value from incomplete algorithm are recorded. **TABLE II** means the accuracy rate of algorithms $FindT$、$FindQ$ and $FindE$.

TABLE II. Efficiency of diagnoses

scale	$FindT$	$FindQ$	$FindE$
20	48%	47.3%	42.6%
100	50.1%	49.6%	44.9%
500	65.3%	63.5%	60.6%
1000	76.7%	75.1%	73.4%
5000	90.6%	90.3%	88.2%
10000	95.2%	95.1%	93.6%

When the scale of model increases, the efficiency reduces, but due to get more information from model, the accuracy rate of algorithm increases.

VI. CONCLUSIONS AND FUTURE WORK

This paper designs a model to solve the incomplete diagnoses online, and proposes an algorithm to process these incomplete behaviors. By the learning results, these incomplete behaviors will be modeled, and the new model will be merged with the former model. By the model and algorithm, the online diagnosis can process rougher behaviors set, and can be more suitable for alterable environment. Based on the methods, some experiments are designed. The results proved that when the incomplete behavior occurred, although the efficiency of diagnosing will reduce, but diagnoses will be gotten.

Discrete event systems in model based diagnosis can be generated by learning methods. In future work, we will increase the degree of incomplete degree and eventually generate models directly from the data, thus effectively reducing the cost in the modeling process.

ACKNOWLEDGMENT

This work was supported by Science and Technology Project of the 13th Five-Year Plan of Jilin Provincial Department of Education: JJKH20180763KJ And supported by Science and Technology Development Plan Project of Siping: 2017093.

REFERENCES

[1] Sampath M, Sengupta R, Lafortune S, et al. Diagnosability of discrete-event systems. *IEEE Transactions on automatic control*, **1995**, 40(9): 1555-1575.

[2] Sampath M, Sengupta R, Lafortune S, et al. Failure diagnosis using discrete-event models. *IEEE transactions on control systems technology*, **1996**, 4(2): 105-124.

[3] Cassar J P , Staroswiecki M , Declerck P . Structural decomposition of large scale systems for the design of failure detection and identification procedures[C]// *International Symposium on Distributed Computing & Applications to Business*. IEEE, **2012**.

[4] De Kleer J. Problem solving with the ATMS. *Artificial Intelligence*, **1986**, 28(2): 197-224.

[5] Reiter R. A theory of diagnosis from first principles. *Artificial Intelligence*，**1987**，32 (1):57-95.

[6] Lamperti G, Zhao X. Diagnosis of active systems by semantic patterns. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, **2014**, 44(8): 1028-1043.

[7] Cassandras C G , Lafortune S . Introduction to Discrete Event Systems[M]. Springer US, **2008**.

[8] Zhao X , Ouyang D . Model-Based Diagnosis of Discrete Event Systems with an Incomplete System Model[C]// *Conference on Ecai: European Conference on Artificial Intelligence*. IOS Press, **2008**.

[9] White A, Karimoddini, A. Event-based diagnosis of flight maneuvers of a fixed-wing aircraft[J]. Reliability Engineering & System Safety, **2020**, 193: 106609.

[10] Wang X. Discrete-Event System Diagnosis upon Incomplete Model[J]. *Journal of Software*, **2012**,23(3):465-475.

[11] Kwong R H , Yongemallo D L . Fault Diagnosis in Discrete-Event Systems: Incomplete Models and Learning[J]. *IEEE Transactions on Systems Man & Cybernetics Part B*, **2011**, 41(1):118-130.

[12] Lamperti G , Zhao X . Diagnosis of Active Systems by Semantic Patterns.[J]. *IEEE Transactions on Systems Man & Cybernetics Systems*, **2017**, 44(8):1028-1043.

[13] Lamperti G , Zhao X . Viable diagnosis of complex active systems[C]// *IEEE International Conference on Systems*. IEEE, **2017**.

[14] Geng X , Ouyang D , Zhang Y . Model-based diagnosis of incomplete discrete-event system with rough set theory[J]. *Science China Information Sciences*, **2017**, 60(1):012205.

[15] Xiang Y , Li Z L. Decentralized fault prognosis of discrete-event systems using state-estimate-based protocols[J]. IEEE transactions on cybernetics,**2018**,49(4): 1302-1313.

[16] Bertoglio N., Lamperti G., Zanella M. Temporal Diagnosis of Discrete-Event Systems with Dual Knowledge Compilation[M]. In: Holzinger A., Kieseberg P., Tjoa A., Weippl E. (eds) Machine Learning and Knowledge Extraction. CD-MAKE **2019**. Lecture Notes in Computer Science, vol 11713. Springer, Cham

[17] Yannick Pencolé, Cordier M O . A formal framework for the decentralised diagnosis of large scale discrete event systems and its application to telecommunication networks[J]. *Artificial Intelligence*, **2005**, 164(1-2):121-170.

[18] Grastien A , Cordier M O. Exploiting independence in a decentralised and incremental approach of diagnosis[C]// *International Joint Conference on Artirficial Intelligence*. DBLP, **2007**