Hindawi
Wireless Communications and Mobile Computing
Volume 2020, Article ID 8887933, 10 pages
https://doi.org/10.1155/2020/8887933

*Research Article*

# An Intelligent Planning-Based Modeling Method for Diagnosis and Repair

**Chuang Li,[1,2,3] Dantong Ouyang,[1,3] Xiaoyu Wang,[2] and Wei Wei[4]**

[1] *College of Computer Science and Technology, Jilin University, Changchun 130012, China*
[2] *College of Computer, Jilin Normal University, Siping 136000, China*
[3] *Key Laboratory of Symbolic Computation and Knowledge Engineer of Ministry of Education, Changchun 130012, China*
[4] *Public Computer Education and Research Center, Jilin University, Changchun 130012, China*

Correspondence should be addressed to Wei Wei; wei_wei@jlu.edu.cn

Received 24 April 2020; Revised 26 May 2020; Accepted 22 June 2020; Published 24 July 2020

Academic Editor: Wei Wang

Planning and model-based diagnosis are both branches of artificial intelligence. In model-based diagnosis, multiresults may be gotten which lead to an uncertain diagnosis. We use the landmark method from planning to designing an event sequence to get a reaction. A method that uses planning to repair in local results of incremental diagnosis is proposed. Firstly, a model is established on model-based diagnosis and planning. Incremental diagnosis results are used as the initial state of planning, and the heuristic search method is used to find the solution to an unfaulty state. Two algorithms with different strategies are designed for diagnosis and repair: one is to repair all possible faults and use controllable events to repair them, and the other is to test through the feedback of controllable events and observable events to get the only solution and repair them. At the same time, the efficiency of the incremental diagnosis method is improved based on heuristics.

## 1. Introduction

AI planning and model-based diagnosis are developing at the forefront of AI research. Diagnostic problems originate from medicine and then extend to the detection of equipment failure or behavior deviation. As machinery equipment gradually replaces manual work, the significance of safety is gradually highlighted, so the broad diagnosis problems are also paid attention to in the field of control and artificial intelligence [1]. Model-based diagnosis is a diagnostic method of common reasoning based on the equipment model and behavior. Because of its low equipment dependence and high efficiency, it has developed rapidly in theory and application since it was put forward.

The model-based diagnosis method was originally used in static systems [2]; then, the conflict-based method was used in other problems such as knowledge representing [3] and planning [4]. After the system shutdown, the fault of the current system was inferred based on the online observations. However, as the scale of equipment increases, the behavior becomes more and more complex and the loss caused by shutdown increases. The dynamic diagnosis that can be monitored online is put forward [5, 6] and has been extended to various industries.

The goal of dynamic diagnosis is to discover the possible faults online and present the faults and their causes as the results; the results are useful to the industrial system [7, 8]. Furthermore, when the system failure is identified, we hope the system can be effectively intervened and repaired online [9]. Intelligent repair of the system is the follow-up work of fault diagnosis, which further improves the security of the system.

Dynamic model-based diagnosis methods are usually based on discrete event system models, which decompose the behavior of the system into persistent states and abrupt events [10]. Faults are defined as special events. When running online, only part of the system's behavior can be detected, which is called an observable event. The diagnostic system infers whether there is a fault in the current state of the system according to the discrete event system model and the sequence of observable events. If there is a fault in the system, the process of the system's behavior given by the diagnostic results is a path in which states are triggered by events. Diagnosis usually assumes that the fault is a persistent fault, that is, when the fault occurs, without interference, the fault will not disappear by itself. Then, the fault is repaired from the fault state to the nonfault state through controllable events in the device.

When the system failed, the diagnostic system tries to use controllable events to repair it [11], and we hope these faults can be attached. Path planning uses state and action to describe the current environment and system and accomplish the expected tasks. The path planning method is combined with the diagnosis model. The planning method is used to design a path from a faulty state to unfaulty, and by the landmark, incremental diagnosis is provided a heuristic value. On the result of the dynamic diagnosis, the fault is repaired by using path planning through controllable events and known models of the system. The merging strategy in the field of gesture recognition is used for reference; the gestures are collected by different equipment but used by the same framework [12–16].

The article is organized as follows: Section 2 defines the model framework; Section 3 gives the process of diagnosis and planning; Section 4 gives the corresponding algorithm; Section 5 is the experimental results.

## 2. Model

A discrete event system is used to model dynamic diagnosis and map from real world to logical system. Firstly, the basic model is established.

*Definition 1* (model). Model is defined as a five-tuple $G = (Q, E, T, I, F)$, in which $Q$ is a finite state set, $E$ is a finite event set, $T$ is a transition set, $T \subseteq Q \times E \times Q$, $I$ is an initial state set, $I \subseteq Q$, and $F$ is a final state set, $F \subseteq Q$.

State is usually in the process of system stability. Event represents the behavior of the system, which is instantaneous relative to the state. Transfer represents the process in which the system is triggered by the behavior from one continuous state to another state. Event triggering may be passive or active.

In classical diagnostic methods, event sets are usually divided into three subsets according to their properties: observable event set $E_o$, normal unobservable event set $E_n$, and faulty event set $E_f$, and they satisfy $E = E_o \cup E_n \cup E_f$, $E_o \cup E_n = \varnothing, E_o \cup E_f = \varnothing, E_n \cup E_f = \varnothing$. If the system can directly change the state of the system by controlling some events, the event set also includes such special events.

*Definition 2* (controllable events). Controllable events are events that can be controlled and executed by the diagnostic system, marked as $E_c$, $E_c \subseteq E$.

Through controllable events and models, the faults are repaired. A path is formed in the model.

*Definition 3* (trajectory). Trajectory $s$ is the transfer sequence of state triggered by events $s = <q_0, e_0, q_1 \cdots q_n>$, and $\forall t$, $t = q_i \times e_{i+1} \times q_{i+1}, t \in T$, $S(G)$ is defined as the set of all possible trajectories on model $G$.

In actual operation, the system is carried out along a certain trajectory. Some of the events in the trajectory are visible, collected by the diagnostic system, and searched in the predefined model to obtain the determined trajectory. If there is a fault event in the trajectory, the system is considered to be in a fault state. The incremental search path planning is used to repair and restore the system to a normal state.

*Definition 4* (repair path). Repair path is an extension of the model, defined as a three tuple $R = (G_p, Q_{\text{start}}, Q_{\text{goal}})$, among which, $G_p$ is a local system model, $G_p = (Q_p, E_p, T_p)$, and $Q_p \subseteq Q, E_p \subseteq E, T_p \subseteq T$; $Q_{\text{start}}$ is the finite state set with fault label, $Q_{\text{start}} \subseteq Q, \forall q \in Q_{\text{start}}, q = q \times \{e_{f0}, e_{f1} \cdots e_{fi}\}$; $Q_{\text{goal}}$ is the finite state set with fault label, $Q_{\text{goal}} \subseteq Q, \forall q \in Q_{\text{goal}}$, $q = q \times \varnothing$.

The repair path is generated according to the diagnoses, and the initial state is obtained from $Q_{\text{start}}$. According to the model using the incremental method in reverse, the heuristic search is conducted. The path composed of controllable events is obtained, and the local system model is generated. Finally, the set $Q_{\text{goal}}$ is obtained, the system is restarted, and the diagnosis is made.

The system is event-driven, and the possible sequence of events is defined as the language of the system.

*Definition 5* (language). Language on system $G$ is a sequence of possible events $L(G) = <e_i^* >, e_i^* \in 2^E$.

In practice, only part of the behavior in the system can be observed, which is a sequence of observable events corresponding to the model.

*Definition 6* (observable mapping). Observable mapping is defined between event sets and observable event sets.

$$P_o(e) = \begin{cases} e & e \in E_o, \\ \varepsilon & e \in E \setminus E_o. \end{cases} \quad (1)$$

Observable mappings are defined linguistically.

$$P_o(l) = P_o\left(l' e\right) = P_o\left(l'\right) P_o(e). \quad (2)$$

And further, observable mappings are defined on trajectories.

$$P_o(s) = P_o\left(s' e_i q_i\right) = P_o\left(s'\right) P_o(e_i). \quad (3)$$

The sequence of observation events actually obtained during the system operation is recorded as Obs. The same mapping can also be defined between the event set and the fault event set. Conversely, the inverse mapping of an observable mapping is defined as a set of languages.

*Definition 7* (observable inverse mapping). $P_o^{-1}(l') = \{l \mid P_o (l) = l'\}$.

The relationship between observable mapping and inverse mapping is not a one-to-one correspondence, and there may be more than one path to repair faults through controllable events, but a fault-free state can be reached, which is acceptable to the diagnosis system.

*Definition 8* (fault label). Fault label is a set of fault events, $l = \{e_i \mid e_i \in E_f\}$.

The state carries one or more fault labels, which are normal if and only if all fault labels are empty sets. Otherwise, it will be a failure state. When the diagnosis system gets the conclusion of the fault, the fault path will be given, which is actually the state-event sequence with the fault label.

Controllable events confirm repair by modifying the fault label in the state, so adding and deleting effects are set for controllable events. In the state $q$, if triggered by controllable events, the adding effect of controllable events is set to reduce or increase the fault label of the state and transfer the state. The deletion effect of controllable events is set to withdraw from the target state to the current state and add or subtract the fault label of the current state.

*Definition 9* (diagnosis). Diagnosis is defined as a compatible path between model and observation $\Delta(G, \mathrm{Obs}) = \{s \mid s \in S (G), P_o(s) = \mathrm{Obs}\}$.

The diagnosis is definitive, but in the actual system, sensors are limited, leading to more than one diagnosis. If only one result is needed, the most probable diagnoses will be chosen from the candidate set. But when the choice is wrong, the diagnosis will backtrack and choose another. The backtrack will depress the efficiency, so we use a landmark as a heuristic function to improve the chosen procedure.

*Definition 10* (landmark). A landmark $q_m$ in diagnosis is a state with a value, $q_m = q \times n, q \in Q, n \in \mathbb{N}^+$.

The landmark is used to mark the state which leads to more than one trajectories; the value in the state expresses how many times the state is passed.

A path is obtained by the diagnosis. When the path $s$ contains a fault event, the subsequent fault status label must contain the event. Searching forward from path $s$, when there is a path that can get a nonfault state through controllable events, the fault is considered to be repaired. The repair behavior is defined as in the model. An example is given to explain the definitions.

The digraph in Figure 1 is a model with $Q = \{q_0 \cdots q_7\}$, $E = \{e_0 \cdots e_5\}$, $T = \{q_0 \times e_0 \times q_2, \cdots q_5 \times e_5 \times q_6\}$, $I = \{q_0\}$, and $F = \{q_6, q_7\}$. The controllable events can be some special events that can change some states. In the example, it can be $e_3$. $s = q_0 e_0 q_2 e_2 q_7$ is a trajectory. If $q_5$ is a faulty state, $q_1$ is a normal state; a trajectory which contains $t = q_5 e_4 q_1$ may be a repair path. Events in a trajectory can be a language, $e_1 e_3 e_5$ is a language, but a prefix $e_1 e_3$ is a language. In event set $E$, if $e_3$ is observable, the observable map of language $e_1 e_3 e_5$ is $e_3$, and the observable inverse mapping of $e_3$ is $e_1 e_3 e_5$ and $e_1 e_2 e_4 e_3$. If the event $e_5$ is faulty, the state $q_6$ has a label of fault $e_5$, $q_6\{e_5\}$. The diagnoses of $\mathrm{Obs} = <e_3 >$ will be $q_0 q_1 q_4 q_6$ and $q_0 q_1 q_3 q_5 q_1$. Some of the states at the key position is defined as a benchmark, such as $q_1$.

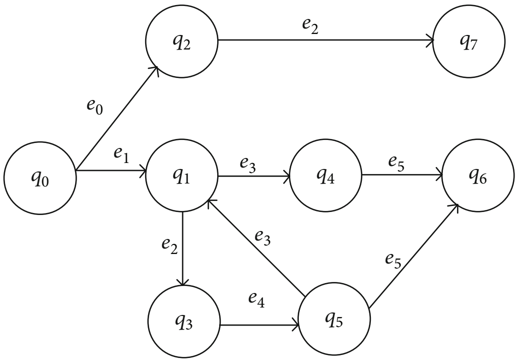

FIGURE 1: An example of a model.

## 3. Method

Discuss the diagnosis and repair of the system using the planning method according to the situation. The incremental method is used to diagnose the system. With the arrival of the observation, one or more diagnostic results may be produced at a time, and the diagnostic results may or may not contain faults. There are four situations as follows:

(1) There is only one diagnosis result, which does not include fault. Keep the current path and wait for the next observation

(2) There is only one diagnosis result, which includes the fault. Path planning based on the heuristic search is carried out from the final state of diagnosis results until a repair path is found

(3) There are many diagnostic results, which do not include faults. The heuristic method is used to select a path as the diagnosis result and continue to wait for the next observation

(4) There are many diagnostic results, including faults. Combined with the multiobjective programming method, the paths that may cover the most diagnostic results are tested. After determining the only fault path, it is repaired

The first case and the third case have been solved in incremental diagnosis, so they are not considered in this paper. Next, we will discuss the situation (2) and (4). They both occur one or some faults that should be repaired, so a planning path should be deduced. The controllable events will carry other behaviors to change the path. So when the planning path is found, the system can be repaired or done some different behaviors.

*3.1. Single Candidate Trajectory.* When the diagnosis only obtains one trajectory and there is a fault event in the trajectory, go backward from the current model and search for possible repair trajectories. The heuristic search method for multiple target paths is used.

The initial state $Q_{\text{start}}$ of the repair path is determined. Only one path in the single candidate path is compatible with the observation. Then, the final state $q$ of the current diagnosis trajectory is selected into the set $Q_{\text{start}}$. The goal is to obtain the local model $G_p = (Q_p, E_p, T_p)$, to make the repair path $R = (G_p, Q_{\text{start}}, Q_{\text{goal}})$, and $q \in Q_{\text{start}}$.

With elements in the set $Q_{\text{start}}$ as the initial state, the controllable events are searched for an extension. The controllable event defines its heuristic value according to the difficulty of its control, adds or deletes the fault label of the state according to the adding or deleting effect of controllable events, and generates the trajectory. In the process of searching, the model $G$ is regarded as the global environment, and the local model $G_p$ that has been obtained is regarded as the local environment. Control events may change the local environment and transfer the state to a specific follow-up state. Incremental methods are used to deal with the search problem in a variable environment.

The search begins in the open table. Controllable events are added according to heuristic values, and the impact on current status fault labels is judged. When a state without a faulty label is obtained, it is added to $Q_{\text{goal}}$. As a final state, all controllable event sequences from $Q_{\text{start}}$ to $Q_{\text{goal}}$ constitute a path, which is a repair sequence.

*3.2. Multiple Candidate Trajectories.* Multiple candidate trajectories have the same observation sequence, but they contain different faults. Therefore, before repairing, a test is carried out as far as possible through controllable events, and the set of candidate trajectories is reduced. A unified or compatible repair path can be obtained by multiobjective programming on the reduced set of paths.

If the current set of candidate diagnostic trajectories is $S_i$ and the observation sequence is $\text{Obs}_i$, then $\forall s_i \in S_i, P_o(s_i) = \text{Obs}_i$. The initial states of the trajectories are the same and the fault labels of the states are different. The repair path $R_i = (G_p, Q_{\text{start}}, Q_{\text{goal}})$. The final states of all diagnostic trajectories are added to the initial state set $Q_{\text{start}}$ of the repair path. Controllable events are used to test the trajectories, and the candidate diagnostic trajectories are further distinguished according to the observations returned again. There may be more than one state in $Q_{\text{start}}$ and more controllable events that can be selected. The controllable events where adding effects can reduce the maximum number of failure events are selected.

That is, in $Q_{\text{start}}$, there are multiple states and their fault labels $q_0 \times l_{00}, l_{01}, \cdots q_n \times l_{n0}, l_{n1}$. Among all controllable event sets that can trigger states, the controllable events that can delete the most faults should be selected. At this time, two kinds of methods are considered for the properties of the system: (1) All repairs are made directly through controllable events until they are fully restored to a normal state. (2) Candidate diagnostic sets are tested by controllable events, and the candidate diagnostic sets are reduced according to the observations obtained from the interaction. The problem is reduced to the repair problem of a single candidate trajectory. Next, we will discuss the situation further.

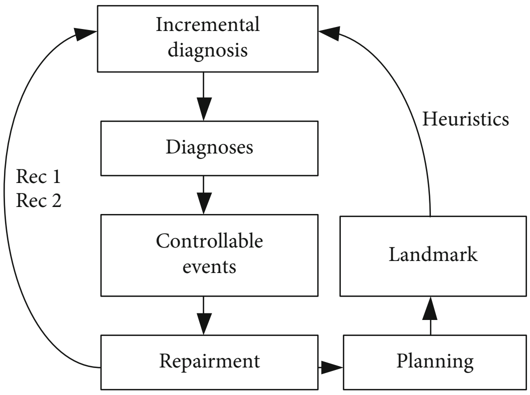

Figure 2: The relationship of the methods.

*3.2.1. Multiple Candidate Repair.* If the current set of candidate diagnostic paths is $S_i$ and the observation sequence is $\text{Obs}_i$, the initial state set $Q_{\text{start}}$ of repair is the final state set of all paths, and it has a fault label. For $\forall q \in Q_{\text{start}}$, all controllable events and adding effects are generated, and events that can produce maximum additive effects are selected. The corresponding state in $Q_{\text{start}}$ is replaced, and the replacement process is added to model $G_p$. The above process is repeated until all the states in $Q_{\text{start}}$ are normal. The relationship between each candidate diagnostic trajectories is "and." The final state set generates $Q_{\text{goal}}$.

*3.2.2. Test and Repair.* In the current final state of candidate trajectories, all controllable events and adding effects are generated. If $\exists e_c \in E_c$ enables at least two paths to produce different observable events, then the event is controlled. In the effective time, if the determined observation events are obtained, the corresponding paths are retained and other paths are deleted. The process is repeated until the only diagnosis result can be selected, and then, the repair of a single candidate path is dealt with.

*3.3. Landmark and Heuristic Incremental Diagnosis.* When the model becomes more complex, diagnosis based on an overall model will be more difficult. Then, the incremental method was proposed by [17]; the diagnosis is gotten incrementally. Incremental diagnosis is gotten from a part of the model and corresponding observations and expresses as the newest result, when another new observations are gotten, the diagnosis will be updated upon the last diagnosis.

But when the multiple trajectories appear, some periodical diagnoses are wrongly chosen, some incremental method backtrack to initial state, and some method backtrack to a

```
Input: System model $G = (Q, E, T, I, F)$, Observation sequence Obs = <$\text{obs}_0$, $\text{obs}_1$ ⋯ .>.
Output: Diagnostic results $\Delta_0, \Delta_1$
$G_f = \text{Comp}(G)$            //Add fault labels to the model
while($\text{Obs} \neq \varnothing$)
    $\Delta_\text{i} = \text{Dia}(G_f, \text{obs}_i)$            //Diagnose the model
if($\exists e_f \in \Delta_i$)
        $\text{Rec}(G_f, \Delta_i)$            //If there is a fault, repair it
Return $\Delta = \Delta_0\Delta_1 \cdots$            //Return all diagnostic results
```

Algorithm 1: Dia and Rec.

former state. If we want to improve the efficiency, we need to reduce the number of backtracks.

To any path $s$, if there is a set $S(G)$, such that $\forall s \in S(G)$, $S(G) = \{s \mid P_o(s) = P_o(s_i)\}$, it gives the basic heuristic value $a$ to the end states in paths, and the heuristic value of the set is $a^2$. When the diagnostic system runs, the heuristic value $a$ is changed by the chosen times and backtrack times.

We use the heuristic value to get a more possible trajectory: at the beginning of diagnosing, the value depends on the number of the trajectories with the same observations, and when the diagnostic system begins to run, the value depends on the number of trajectories was passed. Once backtrack happened in one chosen trajectory, the value will be reduced, and when the recovery planning passes the trajectory, the value will be increased.

The heuristic value is used in diagnosis and planning, which accelerates the diagnostic process. The experiments will be given in part 5.

## 4. Algorithm and Explanation

Online diagnosis and repair mainly include three parts: compilation of offline models, diagnosis based on online observation, and repair based on the diagnosis. Offline model compilation generates fault labels based on models. Online diagnosis uses incremental methods. Diagnosis and repair include two strategies: repair all possible paths and verify the repair of a single path by testing. The relationship of the following methods is described in Figure 2.

Firstly, the whole algorithm is given.

Next, the key algorithms are described in detail. Firstly, offline compilation is carried out according to the model to generate fault labels. Based on the computational diagnosability method, the fault label is generated. On the basis of the computation diagnosability method, fault labels are generated.

The algorithm Comp generates a model with fault label in the offline state. In the online process, if the state with fault label is obtained from the observation, the current system is in a fault condition. The algorithm of online incremental diagnosis is as follows:

In fact, multiple candidate diagnostic trajectories may be obtained for each diagnosis. According to heuristics or other principles, the only path is selected as the result of this diagnosis. The initial state of the next diagnosis result is set to the final state of this diagnosis result. When there is a fault state in the diagnosis result, it is repaired. The repair algorithm Rec is given. According to the strategy of repairing all possible faults or the unique path obtained by a test, two algorithms are given: Rec1 (repair all possible faults) and Rec2 (repair the unique path obtained by a test).

```
Input: System model $G = (Q, E, T, I, F)$
Output: Model with fault label $G_f = (Q \times L, E, T, I, F \times L)$
Initialization: $BI = I$, $Q_f = \{q \mid q = q \times l \times v\}$, $l = \varnothing$, $v = 0$
While($BI \neq \varnothing \| BI \nsubseteq F$)
    $\forall q \in BI, \exists t = q \times e \times q'$
    if($e \in E_f$)
        if($v' == 0$)
        $l' = l' \cup \{e\}$
        else
        $l' = l' \times \{e\}$
$BI = BI - \{q\} \cup \{q'\}$
Return $\text{G}_\text{f}$
```

Algorithm 2: Comp.

If it is a single candidate fault trajectory, a trajectory composed of control events is returned. If the control system triggers controllable events in turn, the fault can be repaired. If the control system triggers controllable events in turn, the fault can be repaired. If there are multiple candidate fault paths, multiple paths may cross or be independent are returned. If the control system triggers controllable events in turn, all possible faults can be repaired. If no control event can be found to repair all candidate diagnostic trajectories, the trajectory with the highest heuristic value is selected and the single candidate fault path is processed instead.

Another method uses heuristic values to select candidate diagnostic trajectories and reduces the set of candidate diagnosis according to the feedback observations until the only diagnostic result is obtained. On this result, the fault is repaired.

According to the continuous feedback observation, the candidate diagnosis set is reduced, and the only diagnosis trajectory is finally obtained, and then, the repair is carried out. Compared with the algorithm Rec1, Rec2 takes a long time to interact continuously, and controlling controllable events for testing may consume more time or system workload. However, when Rec2 has more candidate diagnostic paths, the

```
Input: Model with fault label $G_f = (Q \times L, E, T, I, F \times L)$ Real-time observation sequence Obs = <$\text{obs}_0$, $\text{obs}_1$ $\cdots$ $\text{obs}_i$ $\cdots$ >
Output: Candidate diagnostic results $\Delta_i$
Initialization: $QI = I, QF = \varnothing, S = \varnothing$
$\forall q \in Q, s = <q, e_0, \cdots q' >$
if$(P_o(s) == \text{obs}_i)$
        $S = S \cup \{s\}$
    $s' = \text{Choose}(S)$            //choose the most possible path
    $QF = {q_n}'$
    $QI_{i+1} = {q_0}'$
$\Delta_i = S$
```

Algorithm 3: Dia.

```
Input: Model with fault label $G_f = (Q \times L, E, T, I, F \times L)$
Real-time observation sequence Obs = <$\text{obs}_0$, $\text{obs}_1$ $\cdots$ $\text{obs}_i$ $\cdots$ >
Current diagnostic results $\Delta_i$
Output: Repair path $S_c$
Initialization: $S_c = \varnothing, N_q = \varnothing$
if$(|\Delta_i| > 1)$
    $\forall s \in \Delta_i, N_q = \{q \mid s = <q_0 e_0 q_1 \cdots q >\}$            //Get the final state of the trajectory
    while$(\forall q \in N_q, l_q \neq \varnothing)$
        $e_c = \text{Choose}(q)$            //Select the control event with maximum adding effect
        $G_c = \text{Change}(N_q, e_c)$            //Update status and delete faults
        $S_c = S_c e_c$            //Record a sequence of controllable events
        if$(\text{Dead}(N_q))$            //If it cannot be expanded
            $\text{Back}(N_q, e_c)$            //Backtrack and recover failures
            $S_c = S_c - e_c$
          if$(\nexists S_c)$            //If there is no path to full recovery
          $S_c = \max\,(\Delta_i)$, break            //Choose the diagnosis with the greatest heuristic value and quit multi-fault repair
else
    $s = <q_0 \cdots q >, N_q = \{q\}$            //Get the final state of the path
    while$(l_q \neq \varnothing)$
        $S_c = \text{Find}(N_q, e_c)$            //Get the repair status of the path
        if$(\nexists S_c)$
            Return Fail            //If the repair path cannot be found, fail to exit
```

Algorithm 4: Rec1.

size of the path set is reduced by testing, and the amount of computation is less. Rec1 is prone to generate more branches, which eventually leads to calculation failure.

Rec1 and Rec2 select in the current set of candidate diagnostic trajectories and select controllable events for the next step. The heuristic method is chosen to evaluate the current trajectory. In Rec1, the principle of choosing controllable events is to cover as many paths as possible and repair more faults. In Rec2, the principle of choosing controllable events is to generate as many different observations as possible, so as to distinguish candidate diagnosis trajectories and reduce the set of candidate diagnosis trajectories.

Next, the above methods are tested to get the time and space complexity of the algorithm. The efficiency of repair results of the Rec1 method and the Rec2 method under different conditions (such as the controllable number of events and controllable event efficiency) is compared.

## 5. Experimental Results and Analysis

In this paper, the experimental machine configuration is as follows: processor: Inter i7-6700, 3.4 GHz, memory: 8 GB RAM, Window7 Home 64-bit operating system.

*5.1. Test Problem Description.* The frameworks of three tanks were used in the data set, but the behavior set was expanded, and the controllable event set was added. Using the simulation model to test, the current system has $M$ states and $N$ events, among which the number of normal events, observable events, and fault events is $N_1$, $N_2$, and $N_3$, respectively, the number of controllable events is $N_c$, the execution time is $t_c$, and the time of generating and acquiring observable events is $t_o$. Generating model $G$ is complete and diagnosable, real-time observation timing is normal, and controllable event control is normal.

```
Input: Model with fault label $G_f = (Q \times L, E, T, I, F \times L)$
Real-time observation sequence Obs = $<\mathrm{obs}_0, \mathrm{obs}_1 \cdots \mathrm{obs}_i \cdots>$
Current diagnostic results $\Delta_i$
Output: Repair results $S_c$
Initialization: $S_c = \mathrm{null}, N_q = \varnothing, w = \varnothing, \Delta' = \varnothing$
while($|\Delta_i| > 1$)
$\forall s \in \Delta_i, N_q = \{q \mid s = <q_0 e_0 q_1 \cdots q>\}$
    $e_c = \mathrm{Choose}(q)$            //Choose the path with the most different observations
    $\mathrm{Do}(e_c)$
    $w = \mathrm{Next}(\mathrm{obs}_i)$
    $\Delta' = \mathrm{Dia}(\Delta_i, w)$            //New diagnosis is obtained according to new observations
    $\Delta_i = \mathrm{Delete}(\Delta_i)$            //Update candidate diagnostic results
while($l_q \neq \varnothing$)
    $S_c = \mathrm{Find}(N_q, e_c)$            //Get the repair status of the path
    if($\nexists S_c$)
        Return Fail
```

Algorithm 5: Rec2.

Table 1: Time efficiency of diagnosis and planning methods.

| Time (s) | Rec1 |  | Rec2 |
|---|---|---|---|
|  | Calculation | Control |  |
| $M = 10$ | <0.001 | 1.06 | 1.24 |
| $M = 50$ | 0.02 | 3.31 | 3.68 |
| $M = 100$ | 0.49 | 6.11 | 7.02 |
| $M = 1000$ | 1.98 | 13.76 | 15.99 |
| $M = 5000$ | 4.39 | 39.20 | 44.11 |
| $M = 10000$ | 9.83 | 72.15 | 108.65 |

In the process of running, the model randomly chooses the trajectory to advance and generates the corresponding observation as output. After receiving the observation, the diagnosis system diagnoses the model. After obtaining the fault, it is repaired according to algorithm Rec1 and Rec2, respectively. The efficiency of the Rec1 algorithm is divided into two parts: calculating the time and space efficiency of repairing path, the time efficiency of control event execution, and heuristic selection of controllable events with the most repair states. The efficiency of the Rec2 algorithm includes the overall test path, the time and space efficiency of the calculation and repair, and heuristic selection of controllable events that cause the most different observations.

*5.2. Analysis of Experimental Results.* The test is divided into two parts. Table 1 shows the efficiency of using algorithms Rec1 and Rec2, respectively, when $M : N_1 : N_2 : N_3 : N_c = 10 : 5 : 3 : 2 : 3$, $t_c = 1s$, $t_o = 0.2s$. The time efficiency of Rec1 is divided into two parts. The generated models may be different, so the time is the average value of multiple models.

The time of Rec1 is mainly spent on controllable events. If the time of controllable events is small, the choosing Rec1 is more reasonable. The time of Rec2 is mainly spent on controllable events and in waiting for further observation events. If the critical value $n$ of system diagnosability is small, the choice of Rec2 is more reasonable.

Table 2 shows the ratio of state to event when $M = 1000$ and the time efficiency of diagnosis and repair under the same scale model when controllable events and observation time change. The former value is the time efficiency of using Rec1, and the latter value is the time efficiency of using Rec2.

Similarly, under the same proportion of event states, the time complexity of using Rec1 and Rec2 remains unchanged. When the ratio of $M : (N_1 + N_2 + N_3)$ is larger, the path branches are usually less and the diagnosis efficiency is higher. When the proportion of observable events is higher, the diagnostic efficiency is relatively high because there is relatively little backtracking. When the proportion of controllable events is higher, there are more candidate events. And there is less backtracking when using the Rec1 method, and the efficiency is higher. When the ratio of $M : (N_1 + N_2 + N_3)$ is small, there are more branches and the same diagnosis efficiency is relatively low.

*5.3. Contrast of Heuristic Diagnosis and Incremental Diagnosis.* We contrast the heuristic diagnosis with the landmark and the normal incremental diagnosis; three kinds of model are used: the normal model, the long chain with short local model, and the short chain with long local model. The results are shown in Figures 3–5. The blue line is the normal incremental method, and the red dotted line is the landmark method.

It can be seen from the comparison of Figures 3–5, the heuristic algorithm of using landmark improves the efficiency of diagnosis in different types of models. In incremental diagnosis, the new diagnoses begin from the former result. But when the former result was wrong, the real faults will not be obtained and backtrack will occur. The landmark is used to define the heuristic value to the diagnostic trajectories and reduce the time complexity.

Since the landmark is used to choose a trajectory, the backtrack caused by misdiagnosis is reduced. In the model according to Figure 4, more backtrack is avoided, so the method of the landmark is more efficient.

Table 2: Time efficiency with different percentages of different kinds of events.

| $M : N_1 : N_2 : N_3 : N_c$ | $t_c = 1$<br>$t_o = 0.2$ | $t_c = 1$<br>$t_o = 0.5$ | $t_c = 1$<br>$t_o = 1$ | $t_c = 0.2$<br>$t_o = 0.2$ | $t_c = 0.5$<br>$t_o = 0.2$ |
|---|---|---|---|---|---|
| 10 : 5 : 3 : 2 : 3 | 15.74/15.99 | 15.74/26.45 | 15.74/28.23 | 6.56/9.62 | 8.28/10.13 |
| 10 : 10 : 3 : 1 : 3 | 16.01/16.63 | 16.01/20.06 | 16.01/21.54 | 7.01/8.96 | 8.83/11.04 |
| 10 : 10 : 5 : 2 : 1 | 12.06/12.34 | 12.06/22.35 | 12.06/24.91 | 6.89/8.77 | 7.96/9.29 |

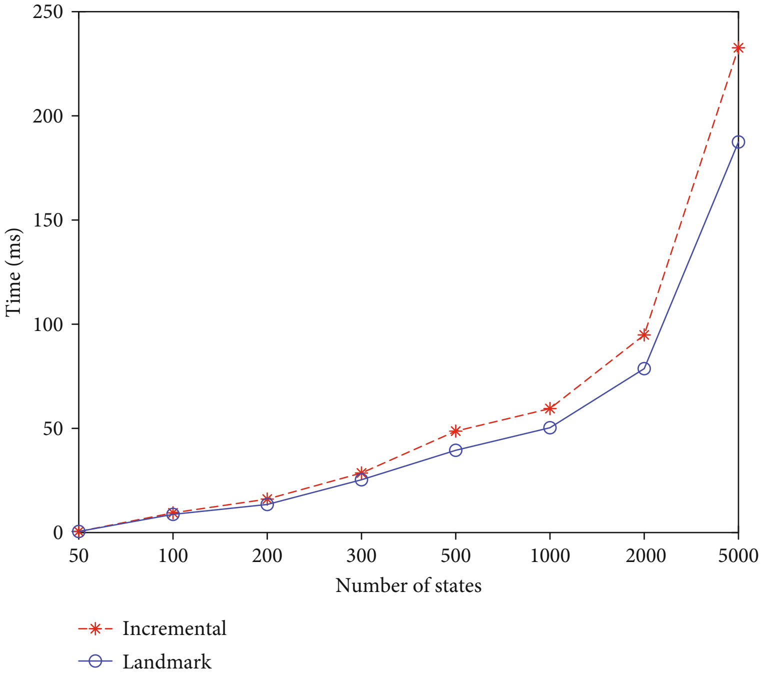

Figure 3: Contrast of basic model.

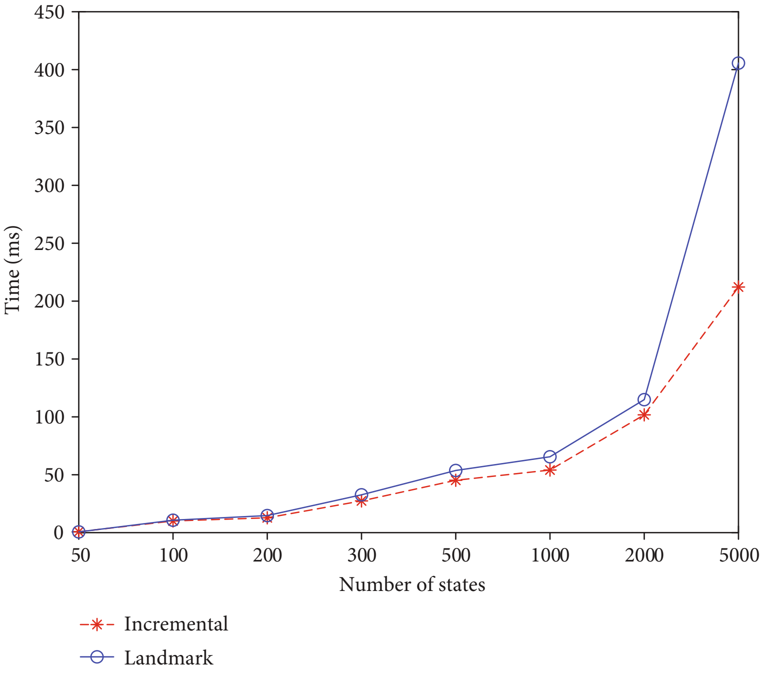

Figure 4: Contrast of long chain with short model.

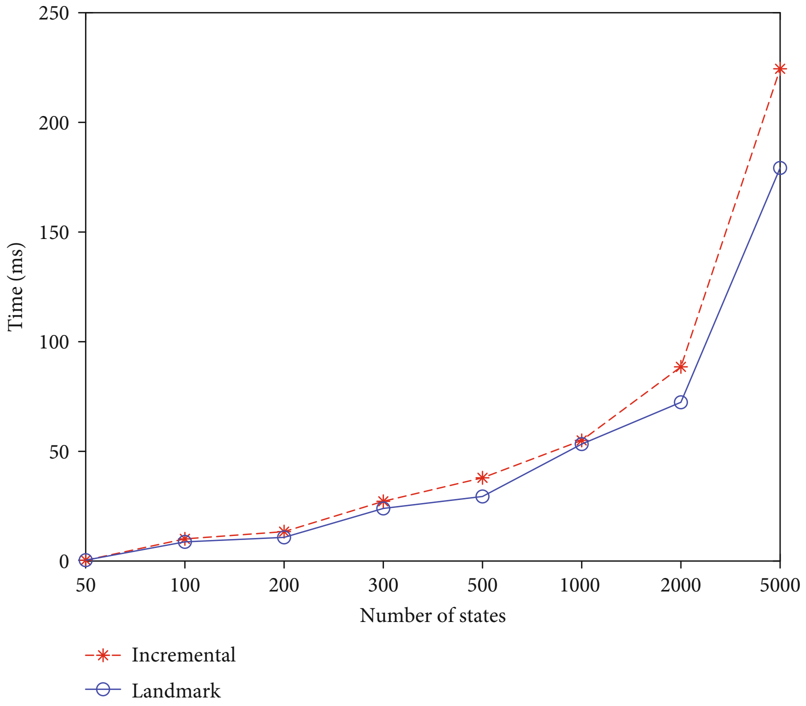

Figure 5: Contrast of short chain with long model.

## 6. Conclusion

Planning and model diagnosis are both symbolic branches of artificial intelligence. A method is proposed to merge the planning model and diagnosis model which is used to solve the fault testing and diagnosis acceleration. Based on the merged model, when an uncertain fault occurs, the landmark is used to generate a heuristic which leads to a better choice, and the controllable events are used to judge whether an event occurs or not. Two kinds of testing methods using controllable events are proposed to cater different cases. If the system has more expensive observation and less expensive controllable events, that means the system is easy to control and hard to observe; then, we use a controllable event to get a reaction. And another method is used to opposite case. A concept of a landmark in planning is used to generate a heuristic value, which accelerates diagnosis. When multiple diagnoses occur, the value will guide a better choice. If the testing method cannot get more observations or controllable events, the landmark can find a more precise result.

In the process of online diagnosis, the most possible path is marked with landmarks to improve diagnosis efficiency. After obtaining the diagnosis results, the landmarks are used to control the planning process and select the possible repair path or test points. A diagnosis-repair framework is proposed. However, if there are few controllable events, even there are no controllable events, the planning will not be able to backtrack, repair, and detect. This method cannot be implemented, which is a disadvantage.

## Data Availability

We will not publish our data for open resource.

## Conflicts of Interest

The authors declare that they have no conflict of interest.

## Acknowledgments

This work was supported by a project grant from the National Natural Science Foundation of China (Grant Nos. 61502197, 61502199, and 61672261), the Science and Technology Project of “13th Five-Year” supported by the Jilin Provincial Education Department (Grant Nos. JJKH20191000KJ and JJKH20180763KJ), Science and Technology Development Project supported by Siping City of Jilin Province (Grant No. 2017093), and the Graduate Innovative Research Project of Jilin Normal University (Grant No. 201949).

## References

[1] L. Travé-Massuyès, “Bridging control and artificial intelligence theories for diagnosis: a survey,” *Engineering Applications of Artificial Intelligence*, vol. 27, pp. 1–16, 2014.

[2] R. Reiter, “A theory of diagnosis from first principles,” *Artificial Intelligence*, vol. 32, no. 1, pp. 57–95, 1987.

[3] D. Benavides, S. Segura, and A. Ruiz-Cortés, “Automated analysis of feature models 20 years later: a literature review,” *Information Systems*, vol. 35, no. 6, pp. 615–636, 2010.

[4] J. Y. Halpern, *Reasoning about uncertainty*, MIT Press, Cambridge, Massachusetts, USA, 2017.

[5] M. Sampath, R. Sengupta, S. C. Lafortune, K. Sinnamohideen, and D. Teneketzis, “Diagnosability of discrete-event systems,” *IEEE Transactions on Automatic Control*, vol. 40, no. 9, pp. 1555–1575, 1995.

[6] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. C. Teneketzis, “Failure diagnosis using discrete-event models,” *IEEE Transactions on Control Systems Technology*, vol. 4, no. 2, pp. 105–124, 1996.

[7] L. K. Carvalho, Y.-C. Wu, R. Kwong, and S. Lafortune, “Detection and mitigation of classes of attacks in supervisory control systems,” *Automatica*, vol. 97, pp. 121–133, 2018.

[8] A. G. C. Gonzalez, M. V. S. Alves, G. S. Viana, L. K. Carvalho, and J. C. Basilio, “Supervisory control-based navigation architecture: a new framework for autonomous robots in industry 4.0 environments,” *IEEE Transactions on Industrial Informatics*, vol. 14, no. 4, pp. 1732–1743, 2018.

[9] C. Edwards, S. K. Spurgeon, and R. J. Patton, “Sliding mode observers for fault detection and isolation,” *Automatica*, vol. 36, no. 4, pp. 541–553, 2000.

[10] C. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, vol. 11 of Kluwer International Series in Discrete Event Dynamic Systems, Kluwer Academic Publisher, Boston, MA, USA, 1999.

[11] H. Ibrahim, P. Dague, A. Grastien, L. Ye, and L. Simon, “Diagnosability planning for controllable discrete event systems,” in *31st National Conference on Artificial Intelligence AAAI-17*, San Francisco, USA, 2017.

[12] W. Cheng, Y. Sun, G. Li, G. Jiang, and H. Liu, “Jointly network: a network based on CNN and RBM for gesture recognition,” *Neural Computing and Applications*, vol. 31, no. S1, Supplement 1, pp. 309–323, 2019.

[13] B. Li, Y. Sun, G. Li et al., “Gesture recognition based on modified adaptive orthogonal matching pursuit algorithm,” *Cluster Computing*, vol. 22, no. S1, Supplement 1, pp. 503–512, 2019.

[14] G. Li, D. Jiang, Y. Zhou, G. Jiang, J. Kong, and G. Manogaran, “Human lesion detection method based on image information and brain signal,” *IEEE Access*, vol. 7, pp. 11533–11542, 2019.

[15] Y. He, G. Li, Y. Liao et al., “Gesture recognition based on an improved local sparse representation classification algorithm,” *Cluster Computing*, vol. 22, no. S5, Supplement 5, pp. 10935–10946, 2019.

[16] D. Jiang, G. Li, Y. Sun, J. Kong, and B. Tao, “Gesture recognition based on skeletonization algorithm and CNN with ASL database,” *Multimedia Tools and Applications*, vol. 78, no. 21, pp. 29953–29970, 2019.

[17] A. Grastien, M. O. Cordier, and C. Largouet, “Incremental diagnosis of discrete-event systems[C],” in *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pp. 1564–1665, Edinburgh, Scotland, 2005.