Abstract—We propose a multiple-fault diagnosis method with high diagnosability, resolution, first-hit and short run time. The method has no assumption on fault models, thus can diagnose arbitrary faults. To cope with the multiple-fault mask and reinforcement effect, two key techniques of construction and scoring of fault-tuple equivalence trees are introduced to choose and rank the final candidate locations. Experimental results show that, when the circuits have 2 arbitrary faults, the average diagnosability and resolution are 98% and 0.95, respectively, with the best case 100% and 1.00. Moreover, in average, even when 21 arbitrary faults exist, our method can still identify 93% of them with the resolution 0.78, increased by 41% and 39% in comparison with the latest work where the diagnosability and resolution are 66% and 0.56. Finally, 96% of our top-ranked candidate locations are actual fault locations.

Keywords—multiple arbitrary faults; diagnosis; mask and reinforcement effect; fault-tuple equivalence tree

I. INTRODUCTION

Rapid yield learning is the key to the success of the electronics industry, where short life-cycles of integrated circuits mandate quick time-to-market and volume [1]. From the time of its conception in the mid 1960s, fault diagnosis plays an important role in the yield learning process [2]. To get a quick time-to-market and low total product cost, fault diagnosis methods need to provide a few but most likely candidate locations of faults to help physical failure analysis (PFA). Single-fault diagnosis is a well-studied problem with various linear-time complexity solutions [3]. However, the single-fault model may not be adequate for diagnosing defects in modern devices where multiple defects tend to cluster and affect several metal wires [4]. Experiments in [5] show that more than 41% of defects cannot be diagnosed using the single stuck-at fault model. Therefore, methods for multiple-fault diagnosis [2, 5-12] and compound-fault diagnosis [13, 14] have been proposed in recent years. Among the works of diagnosing multiple-fault, some of them make restricting assumptions about the fault models [6, 7], while others are independent of them so that they can diagnose arbitrary faults [2, 5, 8-12]. In our work, we focus on the multiple arbitrary faults diagnosis.

A major issue of diagnosing multiple-fault is the multiple-fault mask and reinforcement effect (MFMRE). MFMRE occurs when the failing responses of a single fault are interfered by other faults. According to the patterns used for diagnosis, prior work can be classified into two categories: DDPG (Deterministic Diagnostic Pattern Generation)-based diagnosis and ATPG (Automatic Test Pattern Generation)-based diagnosis. In the first category, dedicated diagnostic patterns are generated to prevent the MFMRE. Moreover, since diagnostic patterns can distinguish some faults that cannot be identified by test patterns, DDPG-based diagnosis can achieve sufficiently high diagnosability and resolution [6]. However, in the situation of volume diagnosis, manufacturers may only use the test patterns to conduct fault diagnosis. In the second category, where only test patterns are used [2, 5, 7-12], MFMRE is mainly handled by three kinds of methodologies. First is executing time-consuming fault simulations with three-valued logic (0, 1, X) [7, 8]. Though this method can solve MFMRE precisely, it costs up to 19 minutes for diagnosing only 3 open faults in one of the ISCAS’89 benchmarks, so it is impractical when many faults exist in the large scale circuit. Second is using SLAT patterns [5, 9]. A SLAT pattern is a failing pattern which can be explained by some single stuck-at fault. Since MFMRE may result in no SLAT patterns existing, and the experiments conducted in [5] have demonstrated the possibility, the SLAT-pattern based method is not effective when confronting MFMRE. The last method is evaluating each potential fault location to choose the final candidates [2, 11, 12]. For example in [12], the faults with the highest propagation possibility to failing observable points are chosen as the final fault candidates. However, this evaluation is not sufficient because only the possibility of a fault been masked or reinforced is considered, but the capability of this fault to mask or reinforce other faults is not evaluated. So the diagnosability decreases fast as the number of faults increases.

Our method belongs to the ATPG-based diagnosis category. In this paper, we define the fault-tuple as a set containing of no less than one faulty line with its faulty value under a certain pattern. To cope with MFMRE, we firstly build a fault-tuple equivalence tree (FTET) for each failing test pattern to store the relation among potential faults to consider the MFMRE. Then we use a scoring mechanism based on the FTETs for sufficiently evaluating the capability of each potential fault to explain each failing pattern so that the final candidate locations can be chosen according to the scores. During the entire diagnosis flow, no characteristics of any fault models are used, and no assumptions on the cause of a faulty value at a line are needed, so arbitrary faults can be diagnosed.
The rest of the paper is organized as follows. Section 2 gives an overview of our diagnosis methodology; Section 3 specifies the diagnosis techniques; Experimental results are given in Section 4. Finally, Section 5 presents the conclusion and future work.

II. DIAGNOSIS METHODOLOGY OVERVIEW

A line may get different faulty values under different patterns, which is determined by the real fault behavior. For example, a stuck-at 0 fault at line \( l \) gives \( l \) the faulty value 0 under all the patterns, but a dominant bridging fault at line \( l \) may give \( l \) either the faulty value 0 or the faulty value 1. Since our work is independent of the fault models, we use \( l/v \) to represent a fault which causes the line \( l \) to get a faulty value \( v \) under a certain pattern without knowing the fault model.

MFMRE is the major issue encountered when diagnosing multiple-fault. If the fault effect can be propagated to at least one observable point in the single-fault situation, but be blocked in the multiple-fault situation, then the multiple-fault mask effect occurs. On the other hand, if the fault effect can not be propagated to an observable point in the single-fault situation, but can be observed in the multiple-fault situation, then the multiple-fault reinforcement effect occurs. An example of MFMRE is shown in Fig. 1. Under the pattern \( abc(001) \), if only \( b/1 \) exists, a failing response at line \( h \) can be observed as shown in Fig. 1-(1); if other two faults of \( a/1 \) and \( c/0 \) occur, the propagation of \( b/1 \) to \( h \) is masked by \( c/0 \) and the propagation of \( b/1 \) to \( g \) is reinforced by \( a/1 \) as Fig. 1-(2) shows. Different from the failing response at \( h \) in Fig. 1-(1), a failing response at \( g \) is observed in Fig.1-(2) if multiple-fault exists.

Assuming there are \( n \) failing patterns, an overview of our methodology is shown in Fig. 2. We will describe each step briefly in this section.

Since it is impractical to traversal all of the fault-tuples to construct equivalent relations among them, we only consider the fault-tuples which can explain the failing patterns. Therefore in the first step, we try to find a first fault-tuple (FFT) which can explain the failing pattern. For the simplicity, we define the FFT as the fault-tuple consisting all of the failing observable points as well as their faulty values. For example, in Fig. 3-(1), the circuit under pattern \( abc(110) \) has two failing observable points \( k \) and \( l \), so the FFT is \( (k/1, l/1) \).

In the next step, for each failing pattern \( p \), we trace from the failing observable points to the circuit inputs to further find the fault-tuples which are equivalent to the FFT. In the sequel, the relations between the potential fault-tuples which can explain the failing pattern \( p \) are built in the FTET. In this paper, if the failing responses of two fault-tuples are totally the same under a certain pattern \( p \), we consider the two fault-tuples are equivalent under this pattern. The MFMRE is considered while building the FTET, and we will describe it in the next section.

After finishing FTET construction, we score all of the potential faults in the FTETs. Then we will choose some potential fault locations as the final candidate locations according to the score. In the next step, we prune these chosen faults and their equivalent fault-tuples in FTETs. If all faults in a FTET are pruned, then the failing pattern of this FTET is explained. When all failing patterns are explained, the diagnosis finishes, otherwise, the process will enter the next iteration. In the next section, we will elaborate these steps.

III. DIAGNOSIS METHODOLOGY USING FAULT-TUPLE EQUIVALENCE TREE

A. Fault-tuple equivalence tree construction

Path-tracing, as the first step to mark the potential fault locations for further analysis, is performed in most of prior work [2, 5-12]. In our method, we not only use path-tracing to find the potential fault locations, but also to build the equivalent relation among them. When constructing the FTET, we start from tracing the gates whose outputs are in the FFT to build the equivalent relation between the faults at inputs and outputs of these gates, and then we continue to build the FTET by tracing the inputs of the gate until arriving at the circuit inputs. Since we know the FFT can explain the failing pattern, once we find its equivalent relation with other fault-tuples, the fault-tuples which can explain the failing pattern are also found.

With the example in Fig. 3, we will show the flow of FTET construction. The fault-free values (FFVs) of all lines are indicated in Fig. 3-(1). FFVs are the logic values of the lines when no faults exist in the circuit. During the process of path-tracing, we may meet two situations as follows:

1. The line under trace is an output of a gate and not a fanout branch of a stem. In this case, we build the equivalent relation between the fault at the output of the gate and the faults at the inputs of the gate, and then continue to trace the potential faulty inputs. Take the line \( k \) and line \( l \) in Fig. 3-(1) as an example, under the pattern \( abc(110) \), because \( FFV_{g/0} = 1 \) and \( FFV_{c/0} = 0 \), \( k/1 \) is equivalent to \( g/1 \), and because \( FFV_{b/0} = 0 \) and \( FFV_{d/0} = 0 \), \( l/1 \) is equivalent to the fault-tuple \((h/1, i/1)\). Then we will continue to trace \( g, h \), and \( i \). In Fig. 3-(3), we draw the FTET graph with each fault in the vertices. If two faults are equivalent, for example, \( k/1 \) and \( g/1 \), we connect their vertices using the symbol “\( = \)”. If a single fault is equivalent to a fault-tuple, for example, \( l/1 \) and the fault-tuple \((h/1, i/1)\), we first connect the vertices of the faults in the fault-tuple using the symbol “\( + \)”, and then use the symbol “\( = \)” to connect “\( + \)” with the vertices of the single fault. In this example, we can see that the fault reinforcement effect between \( h/1 \) and \( i/1 \) is already considered in the FTET.

Figure 1. An example of multiple-fault mask and reinforcement effect

Figure 2. Diagnosis methodology overview
(2) The line under trace is a fanout branch of a stem. In this case, we need to make following two determinations.

(i) Firstly, we need to determine whether all the other fanout branches of this stem will be traced. For example, when our trace arrives at $g$, we find $g$ is a fanout branch of the stem $f$, and then we determine whether the other branch $h$ will be traced. Because $h$ will indeed be traced from $l$, we can build the equivalent relation between the fault at the stem $f$ and the fault-tuple consisting of the faults at all its fanout branches, that is $(g^1, h^1)$.

(ii) If some of the fanout branches will not be traced, we need to determine whether the fault at the stem is equivalent to the fault-tuple consisting of the faults at the fanout branches under trace. For example, while tracing $i$, we find the other fanout branch $j$ is not traced, so we need to determine whether $i^1$ is equivalent to $c^1$. A fault simulation of $j^1$ will be conducted as shown in Fig. 3-(2). We find the logic value of the passing observable point $m$ is changed to 1 because of $j^1$. Hence $i^1$ is not equivalent to $c^1$. In this case, to continue the FTET construction, we must find other faults to mask the fault effect of $j^1$ so that the equivalent relation between the faults at the fanout branches and the fault at the stem is held. These faults can be found by tracing the fault propagation paths. In the example of Fig. 3-(1), we trace the fault propagation path of $j^1$ which is from $j$ to $m$, and find either of the faults $m^0$, $e^0$, and $j^0$ along this path can block the fault propagation of $j^1$, so we can build the equivalent relation between $i^1$ and the fault-tuple $(c^1, m^0)$. Then the equivalent relations between $m^0$ and $e^0$, $m^0$ and $j^0$ are also built. Later $e$ can be further traced.

With these two decisions made above, we can continue to trace the fault at the stem. The FTET construction will stop when the trace arrives at the circuit inputs. Fig. 3-(3) shows the complete FTET of the failing pattern in Fig. 3-(1). From the process of FTET construction, we can see that the MFMRE is considered in the FTET. Besides the FTET ($k^1$, $l^1$), we can find other fault-tuples such as ($k^1$, $h^1$, $i^1$) and ($j^1$, $i^1$) which can also explain the failing pattern according to the equivalent relation. In the next steps, we will evaluate their capabilities of explaining the failing patterns to choose the final candidate locations.

B. Fault-tuple equivalence tree scoring

After the FTET of a failing pattern is built, we score the potential faults in the FTET to evaluate the capability of each fault to explain the failing pattern. The procedure also starts from scoring the FFT and then the fault-tuple which is equivalent to it. Since the faults in FFT have the same capability of explaining the failing pattern, if there are $n$ failing observable points, we give them the same score $1/n$. For example in Fig. 3-(3) where we write the score below the fault in the vertices, since both $k$ and $l$ are needed for explaining the failing pattern, they get the same score 0.5.

Because two equivalent fault-tuples have the same fault effect under a certain pattern, we obey the rule that two equivalent fault-tuples should have the same capability to explain the failing pattern while scoring the other faults. For example in Fig. 3-(3), since $l^1$ is equivalent to the fault-tuple ($h^1$, $i^1$), and $l^1$ has been scored 0.5, we give $h^1$ and $i^1$ the same value 0.25.

To show how the scoring mechanism handles MFMRE, we analyze the score of $b^0$ in Fig. 3-(3). We find that the capability of $b^0$ to explain the failing pattern involves three aspects. Firstly, it can explain the failing observable point $k$. Secondly, it has a reinforce effect for propagating $i^1$ or $c^1$ to the failing observable point $l$, and it can also be reinforced by $i^1$ or $c^1$ to propagate its fault effect to $l$. Finally, it has a mask effect for preventing $c^1$ from propagating to the passing observable point $m$. In Fig. 3-(3), it is scored 0.875 which is the sum score of $d^0$ and $e^0$. We can see that, based on the FTET, not only the possibility of a fault be masked or reinforced is considered, but also the capability of this fault to mask or reinforce other faults is evaluated, thus the scoring mechanism is sufficient.

C. Candidate Locations Choosing and Ranking

When all FTETs have been scored, we firstly choose some faults with the highest score in the FTETs. Since we do not know the model of the fault, a line may get different faulty value in the FTETs. So among the chosen faults, the fault location which appears most in the FTETs is chosen as one of the final candidate locations. Because less and less unexplained patterns are left as the number of iterations increases, the later a candidate location is chosen, the less likely the fault at this location can explain all of the failing patterns, so the lower rank it gets. For example in Fig. 3-(3), since only one failing pattern exits, and $b^0$ has the highest score, we choose $b$ as one of the final candidate locations, which also has the top rank.

D. Fault-tuple equivalence tree pruning

Once some locations are identified, the equivalent fault-tuples of all the faults at them will be pruned from the FTETs. If all faults in a FTET are pruned, that means the fault-tuple consisting of all chosen candidates is equivalent to the FFT, so the failing pattern is explained. After pruning the FTETs, each fault in FTETs will be scored again. This step is necessary because when a candidate is identified, the information can be used to guide more accurate evaluation of other fault’s capability to explain the failing patterns. For example in Fig. 3-(3), when we choose $b$ as a candidate, the equivalent fault-tuples of $b^0$ in the FTET which are shown in Fig. 3-(4) with dashed lines are pruned, and the rest FTET is shown in Fig. 3-(5). The capabilities of $l^1$, $i^1$ and $c^1$ are the same this time.

Figure 3. Example of FTET-based diagnosis methodology
IV. EXPERIMENTAL RESULTS

Experiments are conducted using the full-scan combinational versions of the larger ISCAS’89 benchmarks and ITC’99 benchmarks. The test patterns are generated by a commercial ATPG tool with 100% single stuck-at fault coverage using a 5-detect option. If the test patterns can test faults for at least $N$ times, we call them $N$-detect patterns. According to [8], because of the tester storage space limitation, only at most 100 failing patterns can be stored. Therefore, in our experiment, we use at most 100 failing patterns for diagnosis.

To show the capability of our method to diagnose faults with various fault models, in the experiment, we inject multiple permanent stuck-at faults, transition faults, and dominant bridging nonfeedback faults. Arbitrary faults consisting of permanent stuck-at faults, transition faults, and dominant bridging nonfeedback faults. Arbitrary faults consisting of these three fault types are also injected randomly. For each number of injected faults, each fault type, and each circuit, we conduct experiments 100 times for a total of 19600 injections.

The diagnosis quality is evaluated using four metrics: diagnosability, resolution, first-hit and run time. We also conduct experiments 100 times for a total of 19600 injections. For each number of injected faults, each fault type, and each circuit, we conduct experiments 100 times for a total of 19600 injections.

The diagnosis quality is evaluated using four metrics: diagnosability, resolution, first-hit and run time. We also conduct experiments 100 times for a total of 19600 injections. For each number of injected faults, each fault type, and each circuit, we conduct experiments 100 times for a total of 19600 injections.

A. Diagnosability, Resolution and Run Time

Diagnosability and resolution are two commonly used metrics to evaluate the quality of diagnosis. The diagnosability is defined here the same as the definition used in [12], that is $S_d/S_t$, where $S_d$ is the number of actual fault locations identified by diagnosis and $S_t$ is the number of fault locations injected.

While calculating the resolution with $S_d/S_t$ as used in [12], where $S_t$ is the total number of final candidate locations, we observed an unreasonable phenomena. In the multiple-fault scenario, because the number of candidate locations may be less than the number of fault locations injected, the resolution may be greater than 1. In the worst case, if the candidate locations reported by a diagnosis method match none of the actual fault locations, but the number of candidate locations is the same as the number of actual fault locations, we can still get a high resolution 1 which is obviously not suitable. Since the purpose of calculating resolution is to evaluate the average number of candidate locations in which one actual fault location can be found, we define the resolution as $S_d/S_t$. Thereafter, if none of actual fault locations are identified, the resolution is 0. On the other hand, since some faults can not be distinguished by the failing patterns used for diagnosis, to evaluate the quality of a diagnosis method more reasonable, $S_t$ is defined here as the total number of final candidate locations the faults at which can be distinguished by the failing patterns used for diagnosis.

Run time is the total time used for diagnosis. The average results of diagnosability, resolution and run time are shown in Table 1.

In Table 1, the first column is the circuit name, and below each name is the number of gates in the circuit. The second column is the type of faults injected, and the rest columns are the average diagnosability, resolution, and run time of different

\[\text{TABLE I. RESULTS OF DIAGNOSABILITY, RESOLUTION AND RUN TIME}\]

<table>
<thead>
<tr>
<th>Circuit</th>
<th>2 faults</th>
<th>4 faults</th>
<th>7 faults</th>
<th>10 faults</th>
<th>13 faults</th>
<th>16 faults</th>
<th>21 faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>s3207</td>
<td>1.00</td>
<td>1.01</td>
<td>0.98</td>
<td>0.90</td>
<td>0.93</td>
<td>0.88</td>
<td>0.83</td>
</tr>
<tr>
<td>(2573)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T.</td>
<td>0.99</td>
<td>0.99</td>
<td>0.98</td>
<td>0.98</td>
<td>0.97</td>
<td>0.95</td>
<td>0.93</td>
</tr>
<tr>
<td>B.</td>
<td>0.96</td>
<td>0.96</td>
<td>0.95</td>
<td>0.95</td>
<td>0.94</td>
<td>0.92</td>
<td>0.90</td>
</tr>
<tr>
<td>S.</td>
<td>0.97</td>
<td>0.97</td>
<td>0.96</td>
<td>0.96</td>
<td>0.95</td>
<td>0.94</td>
<td>0.92</td>
</tr>
<tr>
<td>(3448)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A.</td>
<td>0.99</td>
<td>0.99</td>
<td>0.98</td>
<td>0.97</td>
<td>0.96</td>
<td>0.94</td>
<td>0.92</td>
</tr>
<tr>
<td>s35932</td>
<td>1.00</td>
<td>1.00</td>
<td>0.99</td>
<td>0.99</td>
<td>0.98</td>
<td>0.97</td>
<td>0.95</td>
</tr>
<tr>
<td>(12204)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T.</td>
<td>0.99</td>
<td>0.99</td>
<td>0.98</td>
<td>0.97</td>
<td>0.96</td>
<td>0.95</td>
<td>0.93</td>
</tr>
<tr>
<td>B.</td>
<td>0.96</td>
<td>0.96</td>
<td>0.95</td>
<td>0.95</td>
<td>0.94</td>
<td>0.92</td>
<td>0.90</td>
</tr>
<tr>
<td>S.</td>
<td>0.97</td>
<td>0.97</td>
<td>0.96</td>
<td>0.96</td>
<td>0.95</td>
<td>0.94</td>
<td>0.92</td>
</tr>
<tr>
<td>(11448)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A.</td>
<td>0.99</td>
<td>0.99</td>
<td>0.98</td>
<td>0.97</td>
<td>0.96</td>
<td>0.94</td>
<td>0.92</td>
</tr>
<tr>
<td>b17</td>
<td>0.96</td>
<td>0.95</td>
<td>0.94</td>
<td>0.93</td>
<td>0.92</td>
<td>0.91</td>
<td>0.90</td>
</tr>
<tr>
<td>(22645)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T.</td>
<td>0.98</td>
<td>0.97</td>
<td>0.96</td>
<td>0.95</td>
<td>0.94</td>
<td>0.93</td>
<td>0.91</td>
</tr>
<tr>
<td>B.</td>
<td>0.95</td>
<td>0.94</td>
<td>0.93</td>
<td>0.92</td>
<td>0.91</td>
<td>0.90</td>
<td>0.89</td>
</tr>
<tr>
<td>S.</td>
<td>0.97</td>
<td>0.96</td>
<td>0.95</td>
<td>0.94</td>
<td>0.93</td>
<td>0.92</td>
<td>0.91</td>
</tr>
<tr>
<td>(14282)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A.</td>
<td>0.99</td>
<td>0.98</td>
<td>0.97</td>
<td>0.96</td>
<td>0.95</td>
<td>0.94</td>
<td>0.92</td>
</tr>
</tbody>
</table>

(Dig.: Diagnosability; Res.: Resolution; RT: Run time; F.T.: Fault type; S.: Permanent stuck-at faults; T.: Transition faults; B.: Dominant bridging nonfeedback faults; A.: Arbitrary faults)
number of injected faults. From this table, we can see that when the circuit has 2 faults, for all combinations of 7 circuits and 4 fault types, most of the average diagnosabilities are no less than 98%, and most of the average resolutions are above 0.90. Moreover, even when the circuit has 21 faults, the best diagnosability and resolution still achieve 97% and 0.96, respectively. These results demonstrate our diagnosis method is effective when confronting the MFMRE. However the diagnosability still can not be 100%, that is because when multiple-fault exists, under certain failing patterns, some single fault may always be equivalent to no less than one actual faults, so this single fault will have higher capability to explain the failing patterns than the actual faults, and the single fault location will be chosen as the candidate location because of the higher score. Moreover, as the size of circuit and the number of faults increase, the possibility this situation occurs also increases. Therefore, when 21 stuck-at faults are injected into the circuit b17, only 80% of the actual faults are identified. Finally, the run time of the proposed diagnosis method is very short, usually a few of or tens of seconds, even the longest diagnosis time is only 113.81s. That is because each time we prune all fault-tuples related to the faults at the candidate locations, the number of iterations is no more than the number of faults.

B. Comparison with the Prior Work

In this subsection, we compare our results with the earlier work proposed in [12]. The comparison of diagnosability and resolution are shown in Fig. 4 and Fig. 5, respectively. Since the definition of resolution in [12] is different from us, we transform their results for comparison. In these two figures, the x-axis represents the number of injected faults, and the y-axis represents the diagnosability and resolution, respectively. Each column represents one circuit. In average, when 21 arbitrary faults exist, our method can identify 93% of them with the resolution 0.78, increased by 41% and 39% in comparison with the latest work where the diagnosability and resolution are 66% and 0.56. Because the diagnosability of [12] decreases fast along with increasing the number of faults, the more faults exist, the more improvement of diagnosability our method makes.

C. First-Hit

Since it is possible that the chip could contain multiple defects, but it is impractical for PFA to analyze all candidate locations [15], the accuracy of the high ranked candidate locations is very important. If we use the first-hit to evaluate the top-ranked candidate location. If the top-ranked candidate location is an actual fault location, we define the first-hit as 1, or else the first-hit is 0. The average results of first-hit are shown in Table 2. From Table 2, we can see that, for all 19600 times of diagnosis, the overall average first-hit is 96%, which means 18816 top ranked candidate locations are actual fault locations.

D. Diagnosis Quality and Test Patterns

At last, we analyze the relationship between the diagnosis quality and the test patterns used. Two characters of test patterns are considered. One is the number of times a fault can be tested by the test patterns, and the other is which kind of the methods is used for test generation.
Firstly, we compare the diagnosability of the 1-detect, 3-detect and 5-detect test patterns generated by a commercial ATPG tool. We inject 21 arbitrary faults for each circuit for 100 times. The relationship between $N$ and the diagnosability is shown in Fig. 6. Because the more number of times a fault can be tested, the more information can be given for the diagnosis to get the higher diagnosability. Since some faults can still be tested for multiple times by 1-detect patterns, the diagnosability using 1-detect patterns does not decrease too much in comparison with that using 5-detect patterns.

Secondly, we randomly generate 200 patterns for each circuit so that these patterns are independent of the test generation technique. Meanwhile, we use the 5-detect test patterns generated by the commercial ATPG tool for comparison. In the experiment, for all 100 times of fault injections, each time we randomly inject 21 arbitrary faults which can also be tested by the random patterns for 5 times. The comparison of diagnosability and resolution between these two kinds of test patterns are shown in Fig. 7. We can see that the results are similar to each other, which demonstrates our methodology be used for diagnosing various kinds of failing test patterns. Our future work includes using passing test patterns to distinguish the faults which cannot be identified by the failing test patterns.

V. CONCLUSION AND FUTURE WORK

In this paper, we propose a multiple arbitrary faults diagnosis method using FTET to address the MFMRE issue. By iteratively scoring and pruning of the FTETs, we can get the most likely candidate locations, so our method achieves a high diagnosability, resolution and first-hit in short run time even when 21 faults are injected. The independence of the test generation technique enables our methodology be used for diagnosing various kinds of failing test patterns. Our future work includes using passing test patterns to distinguish the faults which cannot be identified by the failing test patterns.

REFERENCES